Центральноукраїнський національний технічний університет
Механіко-технологічний факультет
Кафедра кібербезпеки та програмного забезпечення

”Допущено до захисту”
Завідувач кафедри кібербезпеки
та програмного забезпечення
д.т.н., професор
_____ Олексій СМІРНОВ
“____” _____ 2022 р.

ВИПУСКНА КВАЛІФІКАЦІЙНА РОБОТА
за другим (магістерським) рівнем вищої освіти
на тему
“Дослідження та програмна реалізація системи даних карти тахографа”

Виконав здобувач вищої освіти
ІІ курсу, групи КІ-21М-1,4
ОПП «Комп’ютерна інженерія»
спеціальності 123 «Комп’ютерна інженерія»
_____ Шевчук І.І.
«____» _____ 2022 р.

Керівник проекту
доктор технічних наук, професор
_____ Смірнов О.А.
«____» _____ 2022 р.
Рецензент _____
_____

Кафедра_КбПЗ_2022рік

м. Кропивницький

Центральноукраїнський національний технічний університет

Факультет *Механіко-технологічний*

Кафедра *Кібербезпеки та програмного забезпечення*

Рівень вищої освіти *магістр*

Галузь знань _12 _____*"Інформаційні технології"*_____

Спеціальність 123 *"Комп'ютерна інженерія"*

Освітньо-професійна (освітньо-наукова) програма _*"Комп'ютерна інженерія"*_

ЗАТВЕРДЖУЮ
Завідувач кафедри
д.т.н., проф.
_____ Олексій СМІРНОВ
« 6 » вересня 2022 року

ЗАВДАННЯ НА ВИПУСКНУ КВАЛІФІКАЦІЙНУ РОБОТУ ЗА ДРУГИМ (МАГІСТЕРСЬКИМ) РІВНЕМ ВИЩОЇ ОСВІТИ ЗДОБУВАЧА ВИЩОЇ ОСВІТИ

Шевчуку Іллі Івановичу
(прізвище, ім'я, по батькові)

1. Тема роботи *Дослідження та програмна реалізація системи даних карти тахографа*

2. Керівник роботи *Смірнов Олексій Анатолійович, докт. техн. наук, професор*
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом вищого навчального закладу № 19-13 від 17.08.2022 року

3. Строк подання студентом роботи до захисту *10.12.2022 р.*

4. Мета та завдання випускної кваліфікаційної роботи*: Метою розробки є дослідження та програмна реалізація системи даних карти тахографа*

5. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

1. Призначення та область використання.	*6. Наукова новизна.*
2.Перегляд аналогічних існуючих систем.	*7. Економічна ефективність розробленої програми.*
3. Опис і обґрунтування проектних рішень.	*8. Заходи з охорони праці та техніки безпеки.*
4. Етапи програмування системи.	*9. Висновки.*
5. Впровадження системи в промислову експлуатацію	

6. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)

Наукова новизна	*1 аркуш*
Структурна схема системи	*1 аркуш*
Функціональна схема системи	*1 аркуш*
Діаграма процесів	*1 аркуш*
Блок-схема алгоритму роботи додатку	*2 аркуша*
Показники економічної ефективності	*1 аркуш*

Кафедра_КБПЗ_2022рік

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Економічний	Савеленко Г.В.	05.10.2022	14.11.2022
Охорона праці	Оришака О.В.	06.10.2022	16.11.2022

7. Дата видачі завдання « 6 » вересня 2022 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти	Строк виконання етапів випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти	Примітка
1.	Аналіз існуючих систем	10.10.2022 р.	
2.	Постановка задачі, оформлення ТЗ	15.10.2022 р.	
3.	Розробка моделі компонента	20.10.2022 р.	
4.	Розробка структур даних	25.10.2022 р.	
5.	Розробка алгоритмів зв’язку та відображення	30.10.2022 р.	
6.	Програмування алгоритмів	10.11.2022 р.	
7.	Розрахунок економічної ефективності	13.11.2022 р.	
8.	Розрахунки з охорони праці та техніки безпеки	15.11.2022 р.	
9.	Оформлення ПЗ	17.11.2022 р.	
10.	Попередній захист роботи	10.12.2022 р.	

Дата видачі завдання
« 6 » вересня 2022 р.

Підпис керівника
_____ Смірнов О.А.
(прізвище та ініціали)

Завдання прийнято до виконання
« 6 » вересня 2022 р.

Підпис здобувача
_____ Шевчук І.І.
(прізвище та ініціали)

Кафедра КБПЗ 2022рік

АНОТАЦІЯ

Шевчук І.І. Дослідження та програмна реалізація системи даних карти тахографа. 123 Комп'ютерна інженерія. Центральноукраїнський національний технічний університет. Кропивницький. 2022.

В даній випускній кваліфікаційній роботі за другим (магістерським) рівнем вищої освіти розроблено програмне забезпечення, яке призначено для системи даних карти тахографа.

Метою розробки є дослідження та програмна реалізація системи даних карти тахографа.

Об'єктом дослідження є процес даних карти тахографа.

Предметом дослідження є методи даних карти тахографа.

Методи дослідження базуються на методах захисту інформації, методах математичної статистики, методах розробки програмного забезпечення.

Результат роботи – програмна реалізація системи даних карти тахографа.

В процесі роботи над програмною моделлю виконано аналіз існуючих апаратних та програмних засобів. В повній мірі описані всі компоненти розробленого програмного забезпечення.

Розроблено зручний інтерфейс користувача. Наведені інструкції по роботі з програмними засобами.

Програма може використовуватися на ПЕОМ архітектури IBM PC з ОС Windows 10/11.

Програму розроблено в середовищі Embarcadero Delphi.

Ключові слова: комп'ютерна інженерія, захисту доступу, тахограф

Кафедра_КБПЗ_2022рік

ABSTRACT

Shevchuk I.I. Research and software implementation of the tachograph map data system. 123 Computer engineering. Central Ukrainian National Technical University. Kropyvnytskyi. 2022.

In this final qualification work for the second (master's) level of higher education, software is developed, which is intended for the data system of the tachograph card.

The purpose of the development is research and software implementation of the tachograph card data system.

The object of the study is the data process of the tachograph card.

The subject of the study is the data methods of the tachograph card.

Research methods are based on information protection methods, mathematical statistics methods, and software development methods.

The result of the work is the software implementation of the tachograph card data system.

In the process of working on the software model, an analysis of existing hardware and software was performed. All components of the developed software are fully described.

A convenient user interface has been developed. Instructions for working with software tools are provided.

The program can be used on PCs of IBM PC architecture with Windows 10/11 OS.

The program was developed in the Embarcadero Delphi environment.

Keywords: computer engineering, access protection, tachograph

КафеДра_КбПЗ_2022рік

ЗМІСТ

Кафедра_КОПЗ_2022рік

Кафедра_КОПЗ_2022рік

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ І ТЕРМІНІВ

АС	–	автоматизована система
ДВЧ	–	датчик випадкових чисел
ОС	–	операційна система
ПЗ	–	програмне забезпечення
ПЗП	–	постійний запам’ятовуючий пристрій
ППК	–	пристрій прийому карт
ЕЦП	–	електронний цифровий підпис
DFA	–	Differential Fault Analysis – диференціальний аналіз помилок
DPA	–	Differential Power Analysis – диференціальна атака по енергоспоживанню
EEPROM	–	електрично-стираємий програмуємий постійний запам’ятовуючий пристрій
PIN	–	Personal Identification Number

Кафедра_КБПЗ_2022рік

ВСТУП

Актуальність теми. У цей час пластикові карти одержали широке поширення в додатках захисту інформації: це й таксофонні карти, і SIM-карти в стільникових телефонах, це, звичайно ж, платіжні карти різних типів, карти медичного страхування, проїзду в міському транспорті, карти постійного покупця, що стимулюють попит, називані дисконтними, контейнери криптографічних ключів, карти-ключі, що відкривають електронний замок у дверях, електронні посвідчення особи, засоби підтвердження оплати й дійсності абонента в стільниковій телефонії й супутниковому телебаченні, засоби автентифікації користувачів обчислювальної системи й т.д.

Основними завданнями розвитку технології вітчизняних чіп-карт на сьогодні є:

– пошук методів збільшення ефективності використання ресурсів кристала, в умовах неможливості переходу на іншу норму проектування;

– пошук шляхів інтеграції чіп-карт закордонного виробництва в системи, що використовують вітчизняні криптографічні стандарти;

– проектування захищених малоресурсоємних протоколів електронних платежів і ідентифікації на основі чіп-карт;

– проектування захищених безконтактних карт, що несуть як ідентифікаційну, так і платіжну функціональність, які задовольняють вітчизняним стандартам в області захисту інформації;

– пошук нових областей застосування для чіп-карт.

Кафедра_КОПЗ_2022рік

Рішенню комплексу вищевказаних теоретичних і практичних питань і присвячена дана магістерська робота. Крім того, розроблені методи зменшення ресурсоємності, застосовувані для рішення набору даних прикладних завдань, можуть бути використані й в інших областях, у яких є подібні завдання. Останнє робить магістерську роботу актуальною не тільки для розглянутої предметної області.

Мета й завдання дослідження. Метою роботи є дослідження та програмна реалізація системи даних карти тахографа.

Для досягнення поставленої мети визначена програма дослідження, що складається з наступних завдань:

– Огляд існуючих систем даних карти тахографа.

– Дослідження системи даних карти тахографа.

– Програмна реалізація системи даних карти тахографа.

Об'єктом дослідження є процес даних карти тахографа.

Предметом дослідження є методи даних карти тахографа.

Методи дослідження базуються на методах захисту інформації, методах математичної статистики, методах розробки програмного забезпечення.

Наукова новизна отриманих результатів. У процесі рішення завдань, обумовлених цілями дослідження, отримані наступні результати:

– Удосконалено метод даних карти тахографа.

– Розроблено вітчизняний продукт даних карти тахографа, який має більш широкі можливості, на відміну від існуючих аналогів.

Практична цінність отриманих результатів полягає в тому, що розроблені алгоритми дозволяють успішно вирішувати задачі даних карти тахографа.

Достовірність наукових результатів підтверджена теоретичними викладеннями, даними комп'ютерного моделювання, коректними дослідженнями параметрів на функціонуючій обчислювальній мережі, а також відповідністю отриманих результатів окремим результатам, наведеним у науковій літературі.

Кафедра_КБПЗ_2022рік

Робота апробована на LVI Науково-технічній конференції здобувачів вищої освіти «Наука – виробництву», 2022, основні положення випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти надруковані у статті збірника праць молодих науковців ЦНТУ, випуск №13.

Таким чином, виходячи з вищеперерахованого, дослідження та програмна реалізація системи даних карти тахографа, є актуальною задачею, яка потребує вирішення у даній випускній кваліфікаційній роботі за другим (магістерським) рівнем вищої освіти.

Кафедра_КбПЗ_2022рік

1 ПРИЗНАЧЕННЯ ТА ОБЛАСТЬ ВИКОРИСТАННЯ

1.1 Призначення системи

Призначенням системи є захист даних на чіп-картах. Для цього визначимо, які можуть бути види впливів зловмисників, і класифікуємо їх.

За аналогією з тими загрозами, що приводяться в керівному документі "Концепції захисту засобів обчислювальної техніки й автоматизованих систем від несанкціонованого доступу до інформації" ДССЗЗІ України розглядається класифікація порушників по рівнях можливостей:

Рівень 1: Порушник є непривілейованим користувачем таких систем, має фізичний доступ до кардридерів, PIN-падів і інших аналогічних пристроїв, функціональність яких впливає на безпеку системи. При цьому його дії контролюються візуально персоналом компанії, що експлуатує АС і програмно – системами документування подій виявлення атак. Варто уточнити, що візуальний контроль не перешкоджає здійсненню атак по підміні / перехопленню повідомлень, переданих по безконтактному інтерфейсу, а також не перешкоджає застосуванню зловмисником підроблених або програмно або апаратно модифікованих карт, що візуально не відрізняються від оригінальних. Порушник має у своєму розпорядженні всю відкриту інформацію про систему й відкриті галузеві стандарти.

Рівень 2: Порушник є порушником рівня 1, але при цьому його дії не контролюються візуально, що дає йому можливість безкарно робити модифікацію термінального устаткування, у тому числі для перехоплення / підміни переданих повідомлень.

Рівень 3: Порушник є порушником рівня 2, але при цьому він має доступ до серверного устаткування на читання/модифікацію внутрішніх даних, що не перебувають всередині модулів безпеки, має можливість подачі інженерних

Кафедра_КОПЗ_2022рік

команд серверному встаткуванню, у т.ч. на запуск власних програмних модулів або підключення апаратних засобів, однак не має знання криптографічних ключів і не має доступу до устаткування персоналізації.

Рівень 4: Порушник є порушником рівня 3, але при цьому має доступ до устаткування персоналізації, яке поміщає який-небудь із криптографічних ключів на карту у відкритому виді й модулів безпеки.

Рівень 5: Порушник має доступ до устаткування, що дозволяє здійснювати інженерне проникнення, ефективно проводити атаки DPA (диференціальна атака по енергоспоживанню) і DFA (диференціальний аналіз помилок). Даний рівень може містити в собі елементи якого-небудь із рівнів 1-4.

1.2 Область застосування

Областю застосування розроблювального програмного забезпечення є тахографи, які встановлюються на транспортні засоби та банківська система, зокрема механізми забезпечення безпеки при обігу пластикових карт.

Тахограф – це бортовий самописець, який здійснює запис параметрів руху транспортного засобу та режимів праці та відпочинку водіїв, у внутрішню пам'ять пристрою та на Карту Водія для тахографа.

Порядок дій при користуванні цифровим тахографом:

1. На початку вашої робочої зміни водій повинен вставити індивідуальну карту водія для тахографа у відповідний «Перший» слот цифрового тахографа . Карта повинна бути вставлена "Чипом догори". Після встановлення картки необхідно ввести її PIN-код. У випадку, коли pin-код введено неправильно кілька разів, картка блокується. Подальше розблокування можливе введенням PUK-коду, який має йти в конверті разом із карткою. Якщо екіпаж транспортного засобу складається з двох водіїв, то другий водій також повинен вставити свою карту водія у відповідний другий слот тахографа та активувати її введенням pin-коду. Картки водіїв мають обмежений термін дії. Відповідно до закінчення

Кафедра_КБПЗ_2022рік

терміну дії картки необхідно подбати про отримання нової картки, інакше можна отримати штраф за водіння без картки водія. При встановленні карти водія, на дисплеї тахографа відображається ПІБ власника картки, якщо на дисплеї не з'явилося жодної інформації, можливо, картка вставлена не до кінця. Необхідно перевірити правильність установки картки (установка супроводжується характерним клацанням). Карти не повинні знаходитися в області сильного впливу електромагнітного випромінювання – це може призвести до поломки водія. Якщо тахограф не читає картку або видає її назад, необхідно звернутися до сервісного центру. При цьому, якщо екіпаж знаходиться в дорозі, то він все одно повинен протягом 7 днів відремонтувати тахограф у спеціальній майстерні, а до цього часу вносити відповідні записи на тахографову стрічку вручну підписуючи дані особистим підписом.

2. Після встановлення карти водія в тахограф необхідно внести дані про те, де знаходився водій до початку робочої зміни: відпочивав «Режим відпочинок» на тахографі, або був зайнятий іншою роботою «Режим робота».

3. Далі після введення всіх даних водій починає рух. Деякі режими праці та відпочинку фіксуються тахографом автоматично. Наприклад, при включеному запалюванні завжди буде включений режим роботи незалежно від того, їде автомобіль чи ні. Зверніть увагу, що на дисплеї тахографа ви можете спостерігати поточний режим діяльності водія.

4. На початку руху тахограф автоматично включає Режим "Водіння" (піктограма у вигляді керма).

5. Після зупинки автомобіля, водій повинен перевірити поточний режим на дисплеї тахографа. Вимкнувши запалювання ТС, тахограф автоматично перейде в «Режим відпочинок», або водій перемикається в даний режим вручну, натисканням відповідної клавіші (залежить від моделі тахографа).

6. Після закінчення робочої зміни водій повинен вийняти свою карту, натиснувши на відповідну клавішу.

Кафедра_КПЗ_2022рік

Таким чином бачимо, що захист інформації яка зберігається на карті тахографа, аналогічний захисту інформації банківської карти. Тому далі будемо розглядати обидва види карт.

У цей час всі пластикові карти можна умовно підрозділити на два види: пластикові карти з пам'яттю й пластикові карти без пам'яті. Пластикові карти з пам'яттю у свою чергу підрозділяються на карти із захищеною пам'яттю й карти з незахищеною пам'яттю. Карти з незахищеною пам'яттю, як правило, найпоширеніші серед дисконтних карт, на них немає обмежень по запису або читанню даних. Карти з незахищеною пам'яттю практично не зустрічаються серед платіжних карт – дебетових або кредитних, тому що це вкрай небезпечно. Пластикова карта з незахищеною пам'яттю може бути структурована на окремі байти, у яких утримується інформація. Інформацію з незахищеної карти можна скопіювати в оперативну пам'ять або на інший пристрій і використовувати надалі в незаконних цілях. У Україні, як показує практика, досить поширені такі типи шахрайства із пластиковими картами.

У пластикових картах із захищеною пам'яттю використовується спеціальний механізм, що дозволяє записувати інформацію або зчитувати її, використовуючи спеціальний секретний код, доступний винятково емітенту пластикової карти або її власникові. Введення даного коду власником або емітентом пластикової карти встановлює зв'язок із внутрішнім кодом пластикової карти, відбувається порівняння й у випадку збігу, пластикові карти стають доступними для читання/запису або зміни інформації, що втримується на пластиковій карті. При цьому основним елементом захисту виступає той факт, що читання коду, що містить усередині карти, з яким відбувається зіставлення, неможливо. Також елементом захисту пластикою карти від підробки або несанкціонованого зчитування інформації є також те, що пластикові карти з пам'яттю містять область, у яку записані ідентифікаційні дані самої пластикової карти, і ці дані взагалі не можуть бути змінені після виготовлення пластикових

Кафедра_КОПЗ_2022рік

карт. Ідентифікаційні дані наносяться на пластикову карту з пам'яттю методом так званого «пропалювання».

Для найбільш повного захисту пластикових карт із пам'яттю, як правило, грошових карт, застосовується принцип дублювання захисних областей, тобто кожна пластикова карта містить дві області захищеної інформації – дебетову область і кредитову область. Кожні дебетові або кредитна пластикові карти використовується в безготівкових розрахунках за товар або послугу трьома самостійними особами – двома юридичними особами – банком (емітентом пластикових карт) і магазином, що стягує кошти за товар або послугу, а також однією фізичною особою (як правило фізичним) – клієнтом або власником (тримачем пластикових карт). Банк перераховує гроші на пластикові карти (не важливо дебетову пластикову карту або кредитну карту), тобто іншими словами банк «кредитує» пластикові карти. Магазин списує кошти при здійсненні клієнтом покупки товару або послуги, тобто «дебетує» пластикові карти із санкції клієнта. Таким чином, необхідно розмежовувати дебетування й кредитування пластикової карти, з огляду на те при цьому, що тримач карти має доступ до обох даних операцій. Даний захист досягається поділом областей захисту карти з пам'яттю на дебетову й кредитову області, кожна зі сторін має свій секретний ключ для доступу в можливу для неї область. Ключ запису або зчитування інформації із кредитної області карти наданий тільки банку; а ключем запису або зчитування інформації з дебетною області володіє магазин або постачальник послуг. PIN-кодом до карти володіє безпосередньо клієнт, тобто тримач пластикової карти. Будь-яка фінансова операція може бути зроблена по карті тільки при пред'явленні й зіставленні двох ключів: для списання коштів уводиться PIN-код власника карти й ключ запису даних магазина, для внесення коштів на пластикову карту вводиться PIN-код власника карти й код запису банку – емітента пластикових карт. Читання записаних до пам’яті карти ключів захисту або копіювання пам'яті пластикової карти неможливо.

Кафедра_КОПЗ_2022рік

Існують пластикові карти з однією захищеною областю, однак даний вид карт із пам'яттю розповсюджений менш, тому що надає менш надійний захист. У випадку використання даного виду карт і банк – емітент карти, і магазин працюють із однією й тією же областю інформації, використовують однаковий ключ захисту. У цьому випадку порушується принцип захисту інформації, розміщеної на пластиковій карті, і найчастіше такі пластикові карти найбільше часто стають об'єктом шахрайства.

Таким чином, виходячи з вищеперерахованого, дослідження та програмна реалізація системи даних карти тахографа, є актуальною задачею, яка потребує вирішення у даній випускній кваліфікаційній роботі за другим (магістерським) рівнем вищої освіти.

Кафедра_КбПЗ_2022рік

2 ПЕРЕГЛЯД АНАЛОГІЧНИХ ІСНУЮЧИХ СИСТЕМ

2.1 Огляд існуючих систем, технологій, архітектур, програмних рішень за профілем теми випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти

Проведемо огляд сучасних системи даних карти тахографа або платіжної системи.

Сучасну практику банківських операцій, торговельних угод і взаємних платежів неможливо представити без розрахунків із застосуванням пластикових карт.

Система безготівкових розрахунків за допомогою пластикових карт називається електронною платіжною системою.

Для забезпечення нормальної роботи електронна платіжна система повинна бути надійно захищена.

С точки зору інформаційної безпеки в системах електронних платежів існують наступні уразливі місця:

– пересилання платіжних і інших повідомлень між банками, між банком і банкоматом, між банком і клієнтом;

– обробка інформації усередині організацій відправника й одержувача повідомлень;

– доступ клієнтів до засобів, акумульованим на рахунках.

Пересилання платіжних і інших повідомлень пов'язана з такими особливостями:

– внутрішні системи організацій відправника й одержувача повинні забезпечувати необхідний захист при обробці електронних документів (захист кінцевих систем);

Кафедра_КбПЗ_2022рік

– взаємодія відправника й одержувача електронного документа здійснюється опосередковано – через канал зв'язку.

Ці особливості породжують наступні проблеми:

– взаємне впізнання абонентів (проблема встановлення взаємної дійсності при встановленні з'єднання);

– захист електронних документів, переданих по каналах зв'язку (проблема забезпечення конфіденційності й цілісності документів);

– захист процесу обміну електронними документами (проблема доказу відправлення й доставки документа);

– забезпечення виконання документа (проблема взаємної недовіри між відправником і одержувачем через їхню приналежність до різних організацій і взаємної незалежності).

Для забезпечення функцій захисту інформації на окремих вузлах системи електронних платежів повинні бути реалізовані наступні механізми захисту:

– керування доступом на кінцевих системах;

– контроль цілісності повідомлення;

– забезпечення конфіденційності повідомлення;

– взаємна автентифікація абонентів;

– неможливість відмови від авторства повідомлення;

– гарантії доставки повідомлення;

– неможливість відмови від вживання заходів по повідомлення;

– реєстрація послідовності повідомлень;

– контроль цілісності послідовності повідомлень.

Отже, як платіжний засіб в електронній платіжній системі використовуються електронні пластикові карти.

Електронна пластикова карта – це носій інформації, що ідентифікує власника й зберігає певні облікові дані.

Розрізняють кредитні й дебетові карти.

Кафедра_КБПЗ_2022рік

Кредитні карти є найпоширенішим видом пластикових карт. До них відносяться карти загальнонаціональних систем США Visa і MasterCard, American Express і ряду інших. Ці карти пред'являють для оплати товарів і послуг. При оплаті за допомогою кредитної карти банк покупця відкриває йому кредит на суму покупки, а потім через якийсь час (звичайно 25 днів) надсилає рахунок поштою. Покупець повинен повернути оплачений чек (рахунок) назад у банк. Природно, подібну схему банк може запропонувати тільки найбільш заможним і перевіреним зі своїх клієнтів, які мають гарну кредитну історію перед банком або солідні вкладення в банк у вигляді депозитів, цінностей або нерухомості.

Власник дебетової карти повинен заздалегідь внести на свій рахунок у банку-емітенту певну суму. Розмір цієї суми визначає ліміт доступних засобів. При здійсненні розрахунків з використанням цієї карти відповідно зменшується й ліміт. Для поновлення або збільшення ліміту власник повинен знову внести гроші на свій рахунок. Для страхування тимчасового розриву між моментом здійснення платежу й моментом одержання банком відповідної інформації на рахунку клієнта повинен підтримуватися незнижуваний залишок.

Як кредитна, так і дебетова карти можуть бути не тільки персональними, але й корпоративними. Корпоративні карти надаються компанією своїм співробітникам для оплати відрядних або інших службових витрат. Корпоративні карти компанії пов'язані з яким-небудь одним її рахунком. Ці карти можуть мати розділений або нерозділений ліміт. У першому випадку кожному із тримачів корпоративних карт установлюється індивідуальний ліміт. Другий варіант більше пасує невеликим компаніям і не припускає розмежування ліміту.

Пластикова карта являє собою пластину, виготовлену зі спеціальної пластмаси, стійкої до механічних і термічних впливів. По стандарту ISO 9001 всі пластикові карти мають розміри 85.6?53.9?0.76 мм.

Для ідентифікації власника на пластикову карту наносяться:

– логотип банку-емітента;

Кафедра_КОПЗ_2022рік

– логотип платіжної системи, що обслуговує цю карту;

– ім'я власника карти;

– номер рахунку власника карти;

– термін дії карти й т.п.

Крім того, на карті може бути присутнім фотографія власника і його підпис.

Алфавітно-цифрові дані (ім'я, номер рахунку й ін.) можуть бути ембосовані, тобто нанесені рельєфним шрифтом. Це дає можливість при ручній обробці прийнятих до оплати карт швидко перенести дані на чек за допомогою спеціального пристрою – імпринтера, що здійснює "прокатування»карти.

За принципом дії розрізняють пасивні й активні пластикові карти. Пасивні пластикові карти всього лише зберігають інформацію. До них відносяться пластикові карти з магнітною смугою.

Карти з магнітною смугою є поки найпоширенішими – в обігу перебуває понад два мільярди карт подібного типу. Магнітна смуга розташовується на звороті карти й, у відповідності зі стандартом ISO 7811, складається із трьох доріжок. З них перші дві призначені для зберігання ідентифікаційних даних, а на третю доріжку можна записувати інформацію (наприклад, поточне значення ліміту дебетової карти). Однак через невисоку надійність багаторазово повторюваного процесу запису/зчитування запис на магнітну смугу звичайно не практикується.

Карти з магнітною смугою відносно уразливі для шахрайства. Для підвищення захищеності своїх карт системи Visa і MasterCard/Europay використовують додаткові графічні засоби захисту: голограми й нестандартних шрифтів для ембосування. Ембосери (пристрої для тиснення рельєфу на карті) випускає обмежене коло виготовлювачів. У ряді країн Заходу законодавчо заборонений вільний продаж ембосерів. Спеціальні символи, що підтверджують приналежність карти до тої або іншої платіжної системи, поставляються власникові ембосера тільки з дозволу керівного органа платіжної системи.

Кафедра_КОПЗ_2022рік

Платіжні системи з подібними картами вимагають on-line авторизації в торговельних точках і, як наслідок, наявності розгалужених, високоякісних засобів комунікації (телефонних ліній).

Відмінна риса активної пластикової карти – наявність вбудованої в неї електронної мікросхеми. Стандарт ISO 7816 визначає основні вимоги до карт на інтегральних мікросхемах або чіпових карт.

Карти з мікросхемою можна класифікувати по двох ознаках.

Перша ознака – принцип взаємодії із пристроєм, що зчитує. Основні типи:

– карти з контактним зчитуванням;

– карти з безконтактним (індукційним) зчитуванням.

Карта з контактним зчитуванням має на своїй поверхні від 8 до 10 контактних пластин. Розміщення контактних пластин, їхня кількість і призначення виводів різні в різних виробників і природно, що зчитувачі для карт даного типу розрізняються між собою.

Обмін даними між картою з безконтактним зчитуванням і пристроєм, що зчитує, виробляється індукційним способом. Очевидно, що такі карти надійніші й більш довготривалі.

Друга ознака – функціональні можливості карти. Основні типи:

– карти-лічильники;

– карти з пам'яттю;

– карти з мікропроцесором.

Карти-лічильники застосовуються, як правило, у тих випадках, коли та або інша платіжна операція вимагає зменшення залишку на рахунку тримача карти на деяку фіксовану суму. Подібні карти використовуються в спеціалізовані додатки з передоплатою (плата за використання телефону-автомата, оплата автостоянки й т.д.). Очевидно, що застосування карт із лічильником обмежене й не має великої перспективи.

Карти з пам'яттю є перехідними між картами-лічильниками й картами з мікропроцесором. Карта з пам'яттю – це перезаписувана карта-лічильник, у якій

Кафедра_КБПЗ_2022рік

вжиті заходи, що підвищують її захищеність від атак зловмисників. Найпростіші карти з пам'яттю мають обсяг пам'яті від 32 байт до 16 Кбайт. Ця пам'ять може бути організована у вигляді:

– програмувального постійного запам'ятовувального пристрою ППЗП (EPROM), що допускає однократний запис і багаторазове зчитування;

– стираємого електрично-програмувального постійного запам'ятовувального пристрою ЕСППЗП (EEPROM), що допускає багатократний запис і багаторазове зчитування.

Карти з пам'яттю можна підрозділити на два типи:

– с незахищеної (повнодоступною) пам'яттю;

– с захищеною пам'яттю.

У картах першого типу немає ніяких обмежень на читання й запис даних. Ці карти не можна використовувати в якості платіжних, тому що їх досить просто "зламати".

Карти другого типу мають область ідентифікаційних даних і одну або кілька прикладних областей. Ідентифікаційна область допускає лише однократний запис при персоналізації й далі доступна тільки для зчитування. Доступ до прикладних областей регламентується й здійснюється тільки при виконанні певних операцій, зокрема при введенні секретного PIN-коду.

Рівень захисту карт із пам'яттю вище, ніж у магнітних карт. Як платіжний засіб карти з пам'яттю використовуються для оплати таксофонів загального користування, проїзду в транспорті, у локальних платіжних системах (клубні карти). Карти з пам'яттю застосовуються також у системах допуску в приміщення й доступу до ресурсів комп'ютерних мереж (ідентифікаційні карти).

Карти з мікропроцесором називають також інтелектуальними картами або смарт-картами. Це по суті мікрокомп'ютери, які містять всі основні апаратні компоненти:

– мікропроцесор з тактовою частотою 5Мгц;

– оперативне ЗП ємністю до 256 байт;

Кафедра_КОПЗ_2022рік

– постійне ЗП ємністю до 10 Кбайт;

– енергонезалежне ЗП ємністю до 8 Кбайт.

Смарт-карта забезпечує широкий набір функцій:

– розмежування повноважень доступу до внутрішніх ресурсів;

– шифрування даних із застосуванням різних алгоритмів;

– формування електронного цифрового підпису;

– ведення ключової системи;

– виконання всіх операцій взаємодії власника карти, банку й торговця.

Деякі смарт-карти забезпечують режим "самоблокування»при спробі несанкціонованого доступу.

Все це робить смарт-карту високозахищеним платіжним інструментом, що може бути використаний у фінансових додатках, до якого пред'являють підвищені вимоги до захисту інформації. Саме тому смарт-карти є найбільш перспективним видом пластикових карт.

Важливими етапами підготовки й застосування пластикової карти є персоналізація й авторизація.

Персоналізація здійснюється при видачі карти клієнтові. При цьому на карту заносяться дані, що дозволяють ідентифікувати карту і її власника, а також здійснити перевірку платоспроможності карти при прийманні її до оплати або видачі готівки. Первісним способом персоналізації було ембосування.

До персоналізації відносяться також кодування магнітної смуги й програмування мікросхеми.

Кодування магнітної смуги виробляється, як правило, на тім же встаткуванні, що й ембосування. При цьому частина інформації про карту, що містить номер карти й період її дії, однакова як на магнітній смузі, так і на рельєфі. Однак бувають ситуації, коли після первинного кодування потрібно додатково занести інформацію на магнітну смугу. У цьому випадку застосовуються спеціальні пристрої з функцією «читання-запис". Це можливо,

Кафедра_КбПЗ_2022рік

зокрема, коли PIN-код для користування картою не формується спеціальною програмою, а вибирається клієнтом за своїм розсудом.

Програмування мікросхеми не вимагає особливих технологічних прийомів, але зате має деякі організаційні особливості. Так операції по програмуванню окремих областей мікросхеми рознесені територіально й розмежовані по правах різних співробітників. Звичайно ця процедура розбивається на три етапи:

– на першому робочому місці виконується активація карти (введення її в дію);

– на другому робочому місці виконуються операції, пов'язані із забезпеченням безпеки;

– на третьому робочому місці виробляється властиво персоналізація.

Такі міри підвищують безпеку й виключають можливі зловживання.

Авторизація – це процес твердження продажу або видачі готівки по карті. Для проведення авторизації точка обслуговування робить запит платіжній системі про підтвердження повноважень пред'явника карти і його фінансових можливостей. Технологія авторизації залежить від типу карти, схеми платіжної системи й технічної оснащеності точки обслуговування.

Авторизація проводиться або "вручну", або автоматично. У першому випадку здійснюється голосова авторизація, коли продавець або касир передає запит операторові по телефону. У другому випадку карта міститься в автоматизований торговельний POS-термінал (Point-Of-Sale – оплата в точці продажу), дані зчитуються з карти, касир уводить суму платежу, а власник карти – PIN-код (Personal Identication Number – персональний ідентифікаційний номер). Після цього термінал здійснює авторизацію, встановлюючи зв'язок з базою даних платіжної системи (on-line режим), або реалізуючи додатковий обмін даними із самою картою (off-line режим). При видачі готівки процес має аналогічний характер, з тією лише особливістю, що гроші в автоматичному режимі видаються банкоматом, що і проводить авторизацію.

Кафедра_КОПЗ_2022рік

Випробуваним способом ідентифікації власника пластикової карти є використання секретного персонального ідентифікаційного номера PIN. Значення PIN повинне бути відомо тільки власникові карти. З одного боку, PIN повинен бути досить довгим, щоб імовірність угадування за допомогою повного перебору була прийнятно малою. З іншого боку, PIN повинен бути досить коротким, щоб власник міг його запам'ятати. Звичайно довжина PIN коливається від 4 до 8 десяткових цифр, але може досягати 12.

Значення PIN однозначно пов'язане з відповідними атрибутами пластикової карти, тому PIN можна трактувати як підпис власника карти.

Захист персонального ідентифікаційного номера PIN для пластикової карти є критичною для безпеки всієї платіжної системи. Пластикові карти можуть бути можуть бути загублені, украдені або підроблені. У таких випадках єдиним контрзаходом проти несанкціонованого доступу залишається секретне значення PIN. Тому відкрита форма PIN повинна бути відома тільки законному власникові карти. Вона ніколи не зберігається й не передається в рамках системи електронних платежів.

Метод генерації значення PIN впливає на безпеку електронної платіжної системи. Взагалі, персональні ідентифікаційні номери можуть формуватися або банком, або власниками карт.

Якщо PIN призначається банком, то звичайно використовується один із двох варіантів.

При першому варіанті PIN генерується криптографічно з номера рахунку власника картки. Шифрування проводиться за алгоритмом DES с використанням секретного ключа. Достоїнство: значення PIN не потрібно зберігати усередині електронної платіжної системи. Недолік: при необхідності зміни PIN треба міняти або номер рахунку клієнта, або криптографічний ключ. Але банки воліють, щоб номер рахунку клієнта залишався фіксованим. А з іншого боку, оскільки всі PIN обчислюють, використовуючи один ключ, зміну одного PIN при

Кафедра_КБПЗ_2022рік

збереженні рахунку клієнта спричиняє зміну всіх персональних ідентифікаційних номерів.

При другому варіанті банк вибирає PIN випадковим образом, зберігаючи це значення у вигляді криптограми. Обрані значення PIN передається власникам карт по захищеному каналу.

Використання PIN, призначеного банком, незручно клієнтам навіть при невеликій його довжині. Такий PIN важко удержати в пам'яті, і тому власник карти може записати його куди-небудь. Головне – це не записувати PIN безпосередньо на карту або інше видне місце. Інакше завдання зловмисників буде сильно полегшено.

Для більшої зручності клієнта використовують значення PIN, обиране самим клієнтом. Такий спосіб визначення PIN дозволяє клієнтові:

– використовувати той самий PIN для різних цілей;

– задавати в PIN не тільки цифри, але й букви (для зручності запам'ятовування).

Обраний клієнтом PIN може бути переданий у банк замовленою поштою або відправлений через захищений термінал банківського офісу, що негайно його шифрує. Якщо банку необхідно використовувати обраний клієнтом PIN, то надходять у такий спосіб. Кожну цифру обраного клієнтом PIN складають за модулем 10 (без обліку переносів) з відповідною цифрою PIN, виведеного банком з рахунку клієнта. Одержуване десяткове число називається "зсувом". Цей зсув запам'ятовується на карті клієнта. Оскільки виведений PIN має випадковий характер, то обраний клієнтом PIN неможливо визначити по його зсуву.

Головна вимога безпеки полягає в тому, що значення PIN повинне запам'ятовуватися власником карти й ніколи не повинне зберігатися в будь-якій читабельній формі. Але люди недосконалі й дуже часто забувають свої PIN. Тому для таких випадків призначені спеціальні процедури: відновлення забутого PIN або генерація нового.

Кафедра_КБПЗ_2022рік

При ідентифікації клієнта за значенням PIN і пред'явленій карті використовуються два основних способи перевірки PIN: неалгоритмічний і алгоритмічний.

Неалгоритмічний спосіб здійснюється шляхом безпосереднього порівняння введеного клієнтом PIN зі значеннями, збереженими в базі даних. Звичайно база даних зі значеннями PIN клієнтів шифрується методом прозорого шифрування, щоб підвищити її захищеність, не ускладнюючи процесу порівняння.

Алгоритмічний спосіб перевірки PIN полягає в тому, що введений клієнтом PIN перетворять по певному алгоритму з використанням секретного ключа й потім порівнюють зі значенням PIN, що зберігається в певній формі на карті. Достоїнства цього методу перевірки:

– відсутність копії PIN на головному комп'ютері виключає його розкриття персоналом банку;

– відсутність передачі PIN між банкоматом або POS-терміналом і головним комп'ютером банку виключає його перехоплення або нав'язування результатів порівняння;

– спрощення роботи зі створення програмного забезпечення системи, тому що вже немає необхідності дій у реальному масштабі часу.

2.2 Обгрунтування вибору засобів для побудови системи та мови програмування

Embarcadero Delphi, раніше Borland Delphi і Codegear Delphi, – інтегроване середовище розробки ПЗ для Microsoft Windows, Mac OS, iOS і Android мовою Delphi (що раніше носила назву Object Pascal), створена спочатку фірмою Borland і на даний момент приналежна й розроблювальна Embarcadero Technologies. Embarcadero Delphi є частиною пакета Embarcadero RAD Studio і поставляється в

Кафедра_КБПЗ_2022рік

чотирьох редакціях: Community (поширюється безкоштовно й має обмежену ліцензію на використання в комерційних цілях), Professional, Enterprise і Architect.

Delphi 10.4 Sydney

Випущено 26 травня 2020 року. RAD Studio Delphi 10.4 забезпечує значно поліпшену високопродуктивну нативну підтримку Windows, кращу продуктивність розробки, миттєві підказки code completion, прискорення виконання коду із синтаксисом керованих записів, поліпшення виконання паралельних завдань на сучасних багатоядерних CPU, а також містить більш 1000 виправлень багів, поліпшення продуктивності середовища й бібліотек і багато чого крім того.

Основні можливості Delphi 10.4.1:

− Істотні розширення для Windows: поліпшення для застосунків на моніторах 4K High DPI, інтеграція з новим WebView2 на базі Chromium, використання розширених title bars, таких же, як в Office, Explorer, Google Chrome.

− Керування пам'яттю в Delphi тепер стандартизоване на всіх підтримуваних платформах – мобільних, настільних і серверних – використовучи класичну реалізацію керування пам'яттю об'єктів.

− Істотне поліпшення Delphi Code Insight (без можливого блокування IDE – в окремому процесі), що допоможе при роботі з великими проектами.

− Тип даних Delphi «record» тепер підтримуть довільні ініціалізацію, фіналізацію й операції копіювання.

− Розширена підтримка бібліотек C++: ZeroMQ, SDL2, SOCI, libSIMDpp і Nematode.

− Відладник Win 64 (на LLDB) і збирач для C++.

− Поліпшення для C++: Включена велика кількість поліпшень STL з Dinkumware.

− Підтримка Metal Driver GPU для macOS і iOS.

− Вбудований Fmxlinux.

Кафедра_КОПЗ_2022рік

− Компонент Twebbrowser для iOS тепер реалізований на Wkwebview API. Реалізація компонента Media Player для macOS тепер використовує Avfoundation. Реалізований заново стилізуємий FMX компонент TMemo на платформі Windows значно поліпшений і тепер має відмінну підтримку IME.

− Численні поліпшення швидкості й стабільності роботи нашої бібліотеки The Parallel Programming Library (PPL).

− Додані оновлені драйвери для FireBird, PostgreSQL і SQLite.

− Клієнтські бібліотеки HTTP і REST Client розширені застосунковими можливостями роботи з HTTPS. Також були розширені можливості підтримки Amazon AWS services

− У технологію Visual LiveBindings внесена безліч поліпшень, у тому числі швидкодії, що стосуться, застосунків на VCL і FireMonkey

RAD Studio 10.4 Короткий огляд:

– Істотні розширення для Windows. Створення застосунків, що чудово виглядають, із чіткими елементами інтерфейсу на 4k моніторах High DPI за допомогою нової гнучкої підтримки стилів елементів керування на екрані. Інтеграція із сучасними, безпечними web-технологіями від Microsoft – новим WebView2 на базі Chromium. Використання сучасних розширених title bars, таких же, як в Office, Explorer, Google Chrome, у своїх проектах. Істотні поліпшення надійності налагодження в новому відладнику для C++ Windows 64-bit.

– Зросла продуктивність розробки. Ріст продуктивності за рахунок миттєвої реакції підказок code completion у середовищі IDE. Краща сумісність із уже наявною кодовою базою, і спрощення програмування за рахунок уніфікованої архітектури керування пам'яттю. Швидке зв'язування даних і візуальних елементів за допомогою розширеної технології Visual LiveBindings з підвищеною швидкодією. Просте використання розповсюджених бібліотек C++, наприклад, ZeroMQ, SDL2, SOCI, libSIMDpp і Nematode. Оновлена підтримка Amazon AWS cloud.

Кафедра_КОПЗ_2022рік

– Поліпшення швидкодії і якості. Більш 1000 поліпшень швидкодії і якості. Краща ефективність коду за допомогою нового синтаксису custom managed records. Більш швидке виконання паралельних завдань на сучасних багатоядерних CPU. Переконаєтеся в прискоренні відображення на екрані з підтримкою Metal API на macOS і iOS. Краща сумісність із уже наявною кодовою базою й спрощення програмування за рахунок уніфікованої архітектури керування пам'яттю.

Істотне поліпшення Delphi Code Insight

Як найбільше й головне поліпшення інструментів програмування Delphi за багато років, в 10.4 Delphi Code Insight реалізований через Language Server Protocol (LSP). LSP – це технологія генерації результатів для code completion, навігації й інших сервісів в окремому процесі. Це значить, що code completion і Code Insight одержать більш точні результати без блокування IDE. 10.4 забезпечує набагато більш високу продуктивність розроблювачів, які працюють із більшими проектами, що містять мільйони рядків коду.

Delphi Custom Managed Records

Ключове розширення мови Delphi: тип даних Delphi «record» тепер підтримуть довільні ініціалізацію, фіналізацію й операції копіювання. Управляйте тем, як ці структури створюються, копіюються й звільняються з допомогу вашого коду, який буде виконуватися у відповідний момент.

Це розширює потужність конструкцій records в Delphi, які використовуться щоб одержати більшу ефективність у порівнянні із класами.

Єдине керування пам'яттю

Керування пам'яттю в Delphi тепер стандартизоване на всіх підтримуваних платформах – мобільних, настільних і серверних – використовучи класичну реалізацію керування пам'яттю об'єктів.

У порівнянні з Automatic Reference Counting (ARC), це дає кращу сумісність із існучим кодом і спрощує написання компонентів, бібліотек і застосунків.

Кафедра_КБПЗ_2022рік

ARC модель керування пам'яттю model залишилася для керування рядками й посиланнями на тип інтерфейсу на всіх платформах. Для C++ це означає, що при створенні й звільненні Delphi-style класів в C++ використовується звичайне керування пам'яттю, як у будь-якого heap-allocated класу C++, що значно знижує складність коду.

Розширена підтримка бібліотек C++

В 10.4 ми портували багато популярних бібліотек C++ у C++Builder.

Забезпечивши оптимізовану підтримку бібліотек ZeroMQ, SDL2, SOCI, libSIMDpp і Nematode, поряд із уже підтримуваними Boost і Eigen, які можуть бути додані за допомогою менеджера пакетів Getit.

Win 64-відладник і збирач для C++

В 10.4 з'явився новий відладник C++ для Windows 64-bit. Відладник заснований на LLDB і показує значне збільшення стабільності при налагодженні 64-bit застосунків поряд з новими відладочними можливостями, такими як перегляд і інспекція типів начебто рядків C++ і Delphi, а також колекцій STL, включаючи std::vector, std::map і інших. Крім того, згенерована для застосунку відладочна інформація має інший внутрішній формат, сприяючи більш стабільному й багатому на можливості процесу налагодження, більш докладним перегляду й інспекції в debug-time.

Підвищення якості й швидкодії інструментів

- Велика кількість поліпшень STL від Dinkumware.
- Поліпшені деякі найважливіші методи й області RTL, на базі поліпшень сумісності з популярними бібліотеками C++.
- Поліпшена підтримка Cmake.
- Велика кількість виправлень для підвищення стабільності і якості.
- Відновлення Windows API – Обновлено й додали безліч декларацій API щоб добитися ще більшої інтеграції із платформою Windows.

Кафедра_КОТЗ_2022рік

– Загальні вдосконалення в бібліотеці доступу до БД FireDAC, включаючи оновлені драйвера для FireBird, PostgreSQL і SQLite. Вибір статичного або динамічного підключення SQLite до застосунку.

Змінені стилі VCL для High DPI

В 10.4, архітектура стилізації VCL була суттєво розширена для підтримки High DPI і 4K моніторів. Тепер усі елементи UI на формі VCL автоматично масштабуться під відповідне до монітора дозвіл для показу форми. Був оновлений API стилізації для підтримки стилів high DPI.

Кожний графічний елемент UI може бути обраний з наборів різних масштабів і масштабований до потрібного DPI, що дає чітке зображення елементів UI на всіх моніторах.

Нові High DPI стилі й стилізація окремих VCL компонент

Обновлено велике число вбудованих і преміальних VCL стилів для підтримки нового режиму стилізації High-dpi. Це дозволяє вам створювати застосунку з відмінним дизайном для всіх моніторів.

Розроблювачі VCL застосунків тепер можуть використовувати трохи VCL стилів на різних формах в одному застосунку або в різних компонентів на одній формі. Це також включає стилізацію компонентів загальною темою для платформи. Крім застосункової гнучкості використання стилів, це дозволяє використовувати нестилізуємі компоненти із зовнішніх бібліотек в VCL застосунках, що використовуть стиль.

Поліпшена кроссплатформеність

– Додана підтримка Metal Driver GPU для macOS і iOS.

– Крім підтримки останнього iOS SDK, в RAD Studio 10.4 розроблювачі можуть задовольнити нові вимоги Apple до набору стартових екранів.

– Реалізований заново стилізуємий FMX компонент TMemo на платформі Windows значно поліпшений і тепер має відмінну підтримку IME.

– Користувачам редакцій Enterprise або Architect доступна повна інтеграція Fmxlinux з IDE для створення клієнтських застосунків Linux з GUI.

Кафедра_КОПЗ_2022рік

– Компонент Twebbrowser для iOS тепер реалізований на Wkwebview API.

– Реалізація компонента Media Player для macOS тепер використовує Avfoundation.

Оновлений менеджер пакетів Getit

Менеджер пакетів Getit в IDE був значно вдосконалений.

Дати випуску релізів пакетів тепер видні, і можливе сортування списку по цих датах; відбір тільки встановлених пакетів, контенту, доступного тільки при наявності підписки, багато чого іншого.

Універсальний інсталятор для установки Online і Offline

В 10.4 включений новий універсальний інсталятор, який використовує технологію на базі Getit. Цей інсталятор підтримує як online, так і offline (з ISO) варіанти установки.

Тепер обоє варіанта установки дозволяють вам указати початковий набір можливостей RAD Studio для установки, наприклад, свою комбінацію мов програмування й цільових платформ, мов інтерфейсу, і додавати до нього або видаляти непотрібне в будь-який момент.

2.3 Розгорнута постановка завдання

Згідно з технічним завданням на випускну кваліфікаційну роботу за другим (магістерським) рівнем вищої освіти, реалізації підлягає програмне забезпечення, яке призначено для системи даних карти тахографа.

В процесі розробки випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти необхідно виконати наступний обсяг роботи:

а) провести аналіз існуючих систем-аналогів для виявлення їх позитивних і негативних якостей. Результати аналізу врахувати в подальших розробках;

б) вибрати та обґрунтувати методику побудови системи контролю роботи технологічного обладнання на виробництві в автоматизованому режимі. Розробити функціональну та структурну схеми системи;

Кафедра_КОПЗ_2022рік

в) розробити програмне забезпечення системи, що дозволить реалізувати поставлену технічним завданням задачу. Побудувати блок-схеми алгоритмів програми та підпрограми;

г) організувати інтерфейс користувача з метою формування та виводу на екран ЕОМ повідомлень про некоректні дії користувача та нестандартні ситуації в роботі технологічного обладнання;

д) розробити рекомендації по організаційних та методичних заходах, які забезпечать впровадження системи в промислову експлуатацію та її подальшу успішну експлуатацію;

е) провести розрахунки по визначенню економічної ефективності розробленої системи;

ж) розробити заходи по охороні праці при впровадженні та експлуатації системи, а також розробити заходи з цивільного захисту;

з) сформувати висновки про виконаний обсяг робіт та одержані результати.

Кафедра_КбПЗ_2022рік

3 ОПИС І ОБҐРУНТУВАННЯ ПРОЕКТНИХ РІШЕНЬ

3.1 Опис функціонування системи

Розглянемо основні уразливості, при роботі з чіп-картами. До основних класів типових атак на АС на основі чіп-карт, відносяться наступні:

– Соціальна інженерія.

– Соціальна інженерія із застосуванням апаратних засобів.

– Доступ до каналів зв'язку.

– Підміна / модифікація устаткування.

– Інженерне проникнення, DPA/ DFA-атаки, криптоаналіз.

– Закладки, залишені розроблювачами системи.

Розглянуті в магістерській роботі протоколи безпеки спроектовані таким чином, що захищають від атак зловмисників описаних рівнів до рівня 3 включно на доступ до каналів зв'язку й підміну / модифікацію устаткування, якщо в описі протоколу не зроблене уточнення. Захист від атак видів "соціальна інженерія", "соціальна інженерія із застосуванням апаратних засобів" повинна бути забезпечена організаційно-адміністративними мірами. Захист від атак класу "інженерне проникнення, DPA/DFA-атаки, криптоаналіз" здійснюється розроблювачами кристалів чіп-карт, і базових криптоалгоритмів. Захист від закладок, залишених розроблювачами системи, не здійснюється.

Вирішимо завдання побудови архітектури компактної файлової системи мікропроцесорної карти, збільшення ефективності використання ресурсів існуючого кристала вітчизняної мікропроцесорної карти (у першу чергу, EEPROM).

Для цього розробимо архітектуру компактної файлової системи, що дозволяє використовувати ресурси EEPROM істотно більш раціональним образом, рекомендації зі зміни архітектури мікроконтролера карти для

Кафедра_КБПЗ_2022рік

забезпечення можливості виконання внутрішніх скриптів безпосередньо мікроконтролером карти й швидкодіюча реалізація ДСТ 28147:2009 на чіп-картах.

Самим ресурсномістким і дефіцитним видом пам'яті сьогоднішніх інтелектуальної карти є EEPROM. Вона займає більше половини кристала і його розмір обмежує можливості використання карти. Для рішення поставленого завдання були проведені:

– Розробка організації файлової системи таким чином, щоб перенести незмінні частини файлів додатків карти (а їх – до 90% від усього обсягу прикладних даних) у більше дешеве й менш дефіцитне масочне ПЗП. Таким чином, байти того самого файлу зберігаються, залежно від їхнього призначення, у різних пристроях зберігання. При цьому таке зберігання є прозорим для операційної системи й додатків карти (Рисунок 3.1). Дане завдання було вирішено за допомогою FAT із кластерами змінної довжини.

– Зменшення розмірів службових областей.

– Був зроблений перехід від блок-орієнтованої організації файлової системи (що приводить до втрат при вирівнюванні до границі блоку) до байт-байт-орієнтованого.

– Замість розрахунку CRC на файл (коли для читання хоча б одного байта було потрібно перечитати весь файл, щоб перевірити CRC) був реалізований підрахунок CRC на сектор (Рисунок 3.2).

Кафедра_КПЗ_2022рік

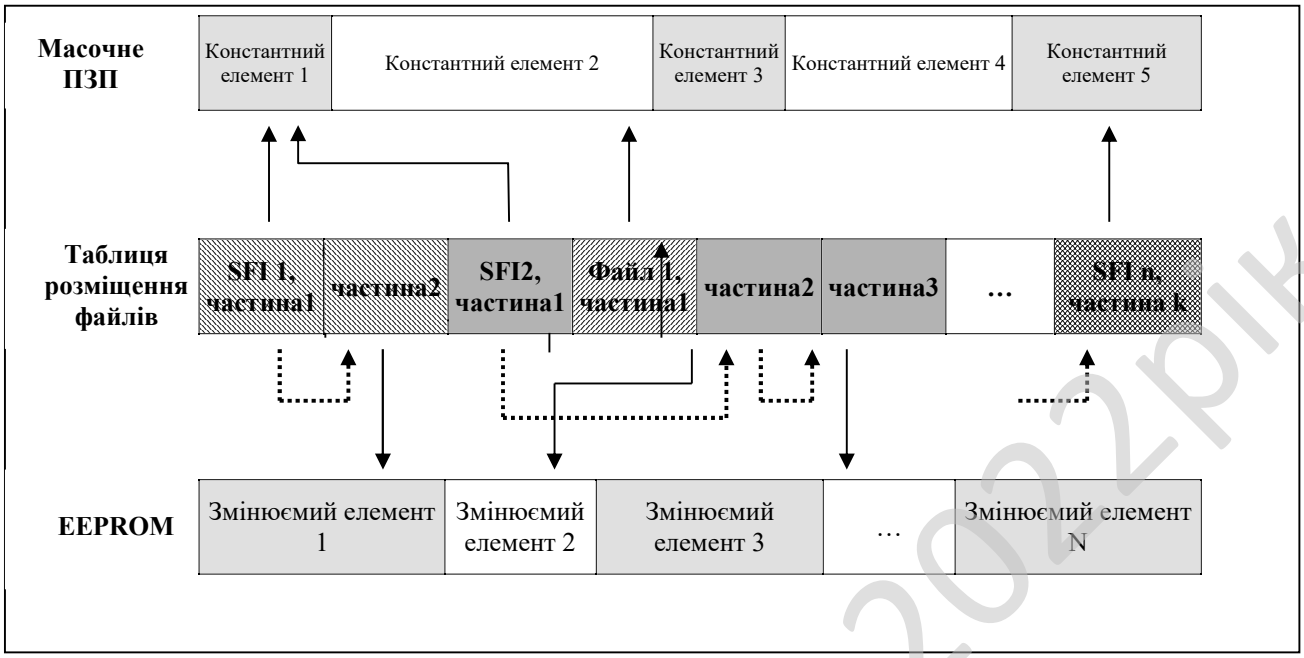

Рисунок 3.1 – Таблиця розміщення файлів

Розроблена архітектура файлової системи має наступні властивості:

– Мінімізовано втрати від вирівнювань.

– Кількість рівнів файлової системи становить не менш трьох.

– Для додатків, емісія яких становить достатній обсяг, з'являється можливість переміщення всіх константних даних з EEPROM у вільну область масочного ПЗП.

– Реалізовано можливість видалення з карти додатків, що мають константні дані в масочному ПЗП для забезпечення можливості використання карти в інших додатках. Оскільки видалення даних з масочного ПЗП неможливо, віддаляються лише посилання на них з EEPROM.

– Для забезпечення можливості використання кристалів з окремими збійними ділянками EEPROM, зсуву даних, збережених в EEPROM, не зберігаються в масочному ПЗП.

Кафедра_КОПЗ_2022рік

– Можливе видалення файлів (що дозволяється лише в деяких файлових системах чіп-карт), однак видалення файлів і додатків не повинні носити масового характеру.

– Залежно від типу, файли можуть мати заголовки різної довжини.

– Структура даних файлової системи поліпшена з погляду мінімізації часу звертання до файлів.

– Файлова система дозволяє забезпечувати збалансоване навантаження по перезапису на сектори EEPROM, тобто немає секторів, перезаписуваних істотно частіше, ніж інші.

Оцінка практичної ефективності (на прикладі EMV-сумісного додатка) запропонованої методики показала, що розмір що вимагається EEPROM може зменшуватися до 3-4 разів.

Далі розглянемо завдання вироблення рекомендацій зі зміни архітектури мікроконтролера карти для забезпечення можливості виконання внутрішніх скриптів безпосередньо мікроконтролером карти, що має метою зменшення простору масочного ПЗП, займаного кодом ОС карти, а також забезпечення можливості кастомізації конфігурації ОС на етапі персоналізації шляхом додавання різних додаткових модулів ОС в EEPROM.

Для забезпечення ізольованості друг від друга додатків, що перебувають на карті, виберемо шлях введення програмного супервізора й введемо аналог захищеного режиму, що дозволяє додатку робити лише безпечні операції, а виконання операцій, критичних з погляду безпеки, здійснюється під контролем (або за допомогою) супервізора.

Кафедра_КОПЗ_2022рік

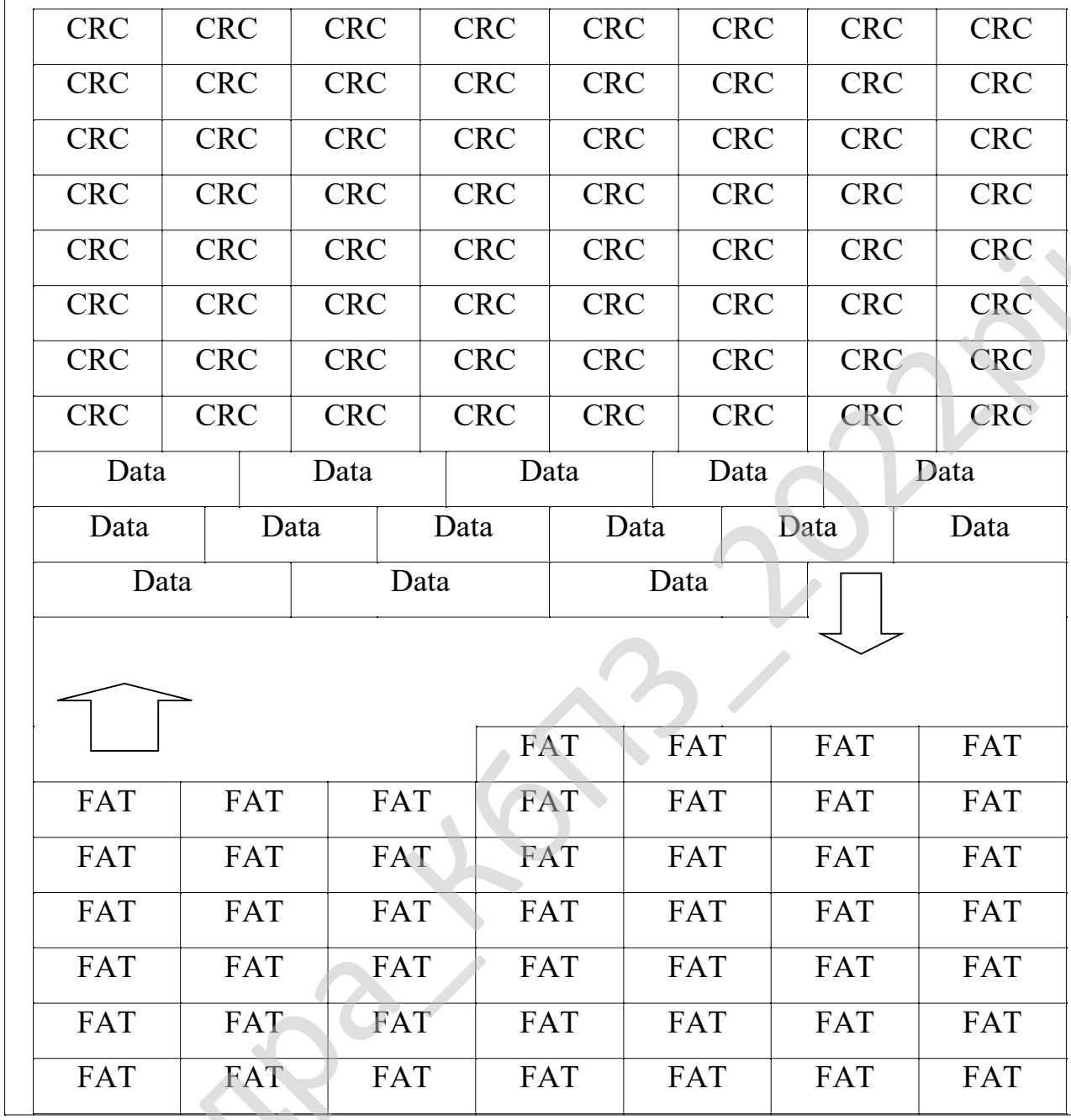

Рисунок 3.2 – Заповнення EEPROM карти файловою системою

Рекомендуємо, до застосування, реалізацію мінімального достатнього набору модифікацій в архітектурі кристала інтелектуальної карти КБ5004ВЕ1 (An15M04):

– введення прапора наявності віртуального режиму в регістрі стану процесора;

Кафедра_КбПЗ_2022рік

– об'єднання адресного простору масочного ПЗП і EEPROM в один адресний простір, для того, щоб програмний код, збережений в EEPROM, був доступний для виконання кристалом;

– поділ набору команд на привілейовані й непривілейовані;

– введення в систему команд кристала додаткової команди перемикання на супервізор;

– використання двох переривань під порушення захисту й під супервізор;

– спроектовані протоколи дозволяють захиститися від атак зловмисників рівнів 1-3, описаних у розділі 1 на доступ до каналів зв'язку й підміну / модифікацію устаткування.

Вироблені рекомендації дозволять розвантажити простір у масочному ПЗП, тому що розмір реалізації супервізора представляється істотно меншим у порівнянні із традиційним інтерпретатором скриптів віртуальної машини. По оцінці, заснованій на розмірах реалізацій інтерпретаторів віртуальної машини в існуючих операційних системах для кристала КБ5004ВЕ1, абсолютна величина виграшу може скласти від одного до півтора кілобайт із шістнадцяти на сьогоднішній день.

Розглянемо реалізацію вітчизняного криптоалгоритму ДСТ 28147:2009 на закордонних картах без необхідності модифікації їх масочного ПЗП, з поліпшеними швидкісними характеристиками. Дане рішення необхідно для забезпечення можливості використання закордонних чіп-карт (не підтримуючих ДСТ 28147:2009) у додатках, що вимагає використання вітчизняної криптографії й наявності обсягів EEPROM, істотно перевищуючих існуючі EEPROM чіп-карт російського виробництва.

ДСТ 28147:2009 орієнтований на мікропроцесори із тридцятидвобітною архітектурою, з порядком байтів BigEndian. Ніяких складностей у реалізації ДСТ 28147:2009 на інтелектуальній карті в принципі немає, якщо тільки карта підтримує опціональний для інтелектуальної карти тридцятидвобітний тип int. На жаль, на сьогоднішній день більшість реалізацій інтелектуальної карти тип int не

Кафедра_КБПЗ_2022рік

підтримують, а карти, на яких інтелектуальної карти підтримують int, мають досить високу вартість. Виходить, необхідно робити тридцятидвобітне додавання як пари шістнадцятибітних, контролюючи переповнення, благо шістнадцятибітний тип short зобов'язаний бути присутнім у будь-якій реалізації інтелектуальної карти. Однак в інтелектуальній карті неможливо здійснити контроль переповнення за допомогою прапора, тому для контролю переповнення при додаванні двох чисел певного типу доводиться перетворювати їх до старшого типу й потім порівнювати з маскою, переконуючись, що результат приводиться до вихідного типу без втрат. Таким чином, при традиційному підході, одне додавання тридцятидвобітних чисел виливається в додавання чотирьох восьмибайтових чисел, і це навіть без обліку інкрементів.

Пропонована схема дозволить здійснювати додавання тридцятидвобітних чисел з будь-якими значеннями шляхом виконання двох шістнадцятибітних додавань, одного інкремента й трьох шістнадцятибітних операцій перевірки знака.

Для прискорення реалізації криптоалгоритма були отримані наступні теоретичні результати:

Визначення 1: Назвемо знаковою інтерпретацією беззнакового числа $x \in \overline{0,2^n - 1}$ конструкцію виду:

$$s(x) := \begin{cases} x, & x < 2^{n-1} \\ x - 2^{n-1}, & x \geq 2^{n-1} \end{cases}, \tag{3.1}$$

Уведемо знакові інтерпретації для що складаються A, B і їхньої суми:

$$a := s(A),\ b := s(B),\ c := s((A + B) \bmod 2^n). \tag{3.2}$$

Під сумою доданків у цьому випадку розуміється операція додавання без обліку переповнення, тобто за модулем 2^n.

Лема 1: $c \geq 0 \Leftrightarrow A + B \in [0, 2^{n-1}) \cup [2 \cdot 2^{n-1}, 3 \cdot 2^{n-1})$.

Лема 2: $\forall n \geq 2 \qquad 2 \cdot (2^n - 1) \geq 3 \cdot 2^{n-1}$.

Кафедра_КБПЗ_2022рік

Твердження 1 (Критерій наявності переносу): Перенос при додаванні чисел A і B (тобто $A+B \geq 2^n$) виникає тоді й тільки тоді, коли щира умова:

$$\begin{cases} b<0 \,\&\, c \geq 0, & a \geq 0 \\ b<0 \vee c \geq 0, & a<0 \end{cases}. \tag{3.2}$$

Отриманий результат дозволяє одержати реалізацію, істотно більш швидку в порівнянні з існуючими аналогами.

Розробимо протокол гнучких і малоресурсоємних типових додатків чіп-карт, що відповідають сучасним вимогам безпеки, придатних до використання в широкому колі виникаючих прикладних завдань, у тому числі з урахуванням специфіки додатків безконтактних карт.

Уведемо поняття універсального облікового додатка й розробимо його в також виробимо ряд рекомендацій з розробки архітектури вітчизняних безконтактних карт, що задовольняють вимогам вітчизняних стандартів в області захисту інформації, на базі кристала MIFARE компанії Philips Semiconductor.

Основною проблемою в даній області є складність розбору виникаючих нестандартних ситуацій, що виникають внаслідок атак або ненавмисних аварійних ситуацій, таких як, переривання живлення або розриви зв'язку в процесі виконання транзакції. Звичайно в подібній ситуації тримач карти приречений очікувати два тижні – час, протягом якого банк повинен зібрати всі оффлайн-журнали для відновлення послідовності подій.

Для забезпечення можливості невідкладного й однозначного трактування всіх відомих атак і помилок, що виникають у процесі проведення платіжних транзакцій були уведені три платіжних лічильники:

– лічильник запитів сертифіката балансу;

– лічильник дебетований;

– лічильник онлайн-операцій.

Розроблено алгоритм виявлення атак шляхом трактування станів лічильників.

Кафедра_КПТВ_2022рік

Розроблена архітектура універсального платіжного додатка задовольняє наступним вимогам:

– можливість захищеного онлайн-поповнення, дебетування, синхронізації й віддаленої зміни платіжних лімітів;

– можливість безпечних оффлайн-дебетування й, можливо, оффлайн-скасування з виконанням всіх вимог безпеки.

– можливість безпечного дебетування на дрібні суми без уведення PIN-коду;

– стійкість до збоїв зв'язку під час онлайн-транзакції: у цьому випадку клієнт не повинен втратити кошти з балансу карти, навіть тимчасово;

– оффлайн- і онлайн-операції рознесені, тобто клієнт навіть після невдало завершеної онлайн-транзакції повинен мати можливість, як і колись, проводити оффлайн-операції;

– можливість використання симетричних криптоалгоритмів, і можливість застосування (у випадку потреби й наявності криптографічного співпроцесора на кристалі карти) асиметричних криптоалгоритмів без серйозних архітектурних змін.

Спроектовані протоколи дозволяють захиститися від атак зловмисників рівнів 1-3, описаних у розділі 1 на доступ до каналів зв'язку й підміну / модифікацію устаткування.

Спроектовано універсальний механізм, називаний універсальним обліковим додатком, на зразок універсального платіжного додатка, що має однакові принципи функціонування в зовсім різних платіжних проектах. Забезпечується унікальність криптограм, передані дані захищаються від модифікації при читанні/запису.

Були зроблені наступні кроки:

– Вичленовано загальні прикладні особливості в різних по призначенню облікових додатків.

Кафедра_КБПЗ_2022рік

– Виходячи з вимог атомарності перезапису секторів з урахуванням контролю автентичності записуваних даних був обраний розмір сектора.

– Обрано й описана конфігурація маски доступу.

– Розроблено набір прикладних команд універсального облікового додатка.

Отриманий універсальний обліковий додаток має наступні властивості:

– Універсальність: наскільки це можливо, додаток задовольняє вимогам різних дисконтних схем.

– Низька ресурсоємність. Функціональність додатка легко реалізується на різних чіп-картах, як вітчизняних, так і закордонних, що дозволяють розширювати набір команд за допомогою скриптів або апплетів інтелектуальної карти. Також повинна бути можливість апаратної реалізації додатку в безконтактній карті без мікропроцесора.

– Для спрощення реалізації додатка на безконтактних картах додаток не вимагає наявності апаратного ДВЧ.

– Гнучка схема розмежування доступу.

– Можливість реалізації декількох облікових додатків на одній карті.

– Можливість розмежування доступу до прикладним даних шляхом двосторонньої криптографічної автентифікації між картою й пристроєм прийому карт (ППК).

– Можливість автентифікації тримача карти за паролем.

– Наявність убудованого в додаток криптографічного контролю цілісності прикладних даних, переданих як з карти, так і на карту.

– Мінімізовано кількість обмінів між картою й ППК.

– Наявність механізму забезпечення унікальності криптограм.

– Спроектовані протоколи дозволяють захиститися від атак зловмисників рівнів 1-3, описаних у розділі 1 на доступ до каналів зв'язку й підміну / модифікацію устаткування.

Кафедра_КОПЗ_2022рік

Розробимо архітектуру вітчизняної безконтактної карти. Вирішимо наступні підзадачи:

– Вибір продукту з лінійки MIFARE як прототип вітчизняної безконтактної карти. Після детального дослідження лінійки кристалів MIFARE компанії Phillips Semiconductors, як прототип кристала вітчизняної безконтактної карти був обраний кристал MF1 IC S50.

– Механізми криптографічної автентифікації були наведені у відповідність із вітчизняними стандартами.

– Організація пам'яті була наведена у відповідність із розмірами ключа застосовуваного криптоалгоритмом ДСТ 28147:2009;

– Механізм автентифікації став забезпечувати можливість контролю цілісності переданих в обох напрямках по ефіру даних.

– Забезпечено унікальність (неповторюваність) даних, записуваних на карту.

Знайдемо шляхи застосування чіп-карт вітчизняного виробництва в системах захисту ПЗ від несанкціонованого копіювання. У ній вирішуються проблеми:

– Можливості використання вітчизняних чіп-карт у системах захисту ПЗ від несанкціонованого копіювання шляхом розробки протоколу симетричної автентифікації суб'єктом, що не зберігає секретний ключ автентифікації.

– Криптографічний протокол голосової активації ПЗ, що захищається від несанкціонованого копіювання, що володіє архітектурою, схожою з попереднім протоколом архітектурою.

– Спроектовані протоколи дозволяють захиститися від атак зловмисників рівнів 1-3, описаних у розділі 1 на доступ до каналів зв'язку й підміну / модифікацію устаткування.

Пропонується протокол симетричної однобічної автентифікації. Загалом, на автентифікуючій стороні зберігається таблиця еталонних відповідей на запити

Кафедра_КБПЗ_2022рік

автентифікації, що не дозволяє, проте, відтворювати свій вміст в емуляторі зловмисника.

Стійкість протоколу забезпечується тимчасовими і ємнісними міркуваннями, а сам протокол показує можливість застосування чіп-карт, що здійснюють лише симетричні криптографічні перетворення, у системах захисту ПЗ від несанкціонованого копіювання.

Діаграма обмінів протоколу автентифікації, наведена в таблиці 3.1.

Істотною вимогою є забезпечення цілісності програмно-апаратного середовища автентифікуючій сторони. При цьому допускається можливість дослідження зловмисником алгоритмів функціонування автентифікуючій сторони, у т.ч. можливість читання зловмисником таблиці еталонних відповідей.

Розробимо криптопротокол віддаленої активації що захищається ПЗ через голосовий телефонний канал, що задовольняє наступним вимогам:

– можлива передача даних тільки один раз від клієнта до сервера й потім один раз назад;

– розмір переданих даних для кожного напрямку не повинен перевищувати 64-128 бітів, більший обсяг передати через голосовий телефонний канал (диктування) представляється скрутним;

– ПЗ, що захищається від несанкціонованого копіювання, повинне бути захищене від модифікації зловмисником;

– допускається наявність у зловмисника повної інформації про протокол активації;

– ПЗ, що захищається від несанкціонованого копіювання, повинне бути захищене від надання апаратним середовищем і операційною системою нав'язувальних зловмисником даних, що представляють собою ідентифікатори й метрики устаткування, а також джерела інформації для програмних ДВЧ;

– на стороні клієнта не повинні вимагатися які-небудь додаткові апаратні засоби;

Кафедра_КБПЗ_2022рік

Таблиця 3.1 – Діаграма обмінів протоколу автентифікації

Мікропроцесорна карта		Система захисту ПЗ від несанкціонованого копіювання
Ініціалізація даних		
Ключ K генерується й міститься на карту		Таблиця $T' = \{h(i) \mid 0 \le i < N\}$, що відповідає ключу K генерується й міститься до пам'яті ПЗ, що захищається, де $h(i) = H(v(i))$, $N = 2^{25} = 33\,554\,432$, розмір таблиці T' складе 128Мб.
Автентифікація карти		
		Генерується випадкове число $r \in \overline{0, N-1}$, і передається карті.
	←	
Формується й вертається відповідь $t = v(r)$ довжиною більше 1 кб		
	→	
		Обчислюється хеш-значення $w = H(t)$, потім по таблиці T', перевіряється рівність $w = h(r)$. Автентифікація вважається успішною у випадку збігу, і неуспішною в противному випадку.

Кафедра_КбПЗ_2022рік

– на клієнті не повинні зберігатися секретні криптографічні ключі;

– повинно бути розрахунково складно для зловмисника земулювати відповідь сервера на запит клієнта;

– можливе використання різних базових криптоалгоритмів одного класу в криптографічному протоколі активації.

Обмеження, що накладаються у вищеописаних вимогах, на розмір і кількість переданих повідомлень унеможливлюють використання асиметричних криптоалгоритмів у силу того, що розміри ЕЦП відомих авторові криптоалгоритмів істотно перевищують задані межі. Використання ж традиційної схеми автентифікації на базі симетричного криптоалгоритма, що припускає зберігання секретного ключа на обох сторонах, також неможливо.

Для підготовки до виконання криптопротоколу на сервері пропонується згенерувати таблицю пар виду:

$$\{(r,c) \mid r = H(c)\}, \tag{3.4}$$

де:

r – двійковий вектор розміром l_r від 8 до 16 байт, називаний запитом активації (або, просто, запитом);

c – двійковий вектор розміром l_c від 8 до 16 байт, називаний підтвердженням активації (або, просто, підтвердженням), вибирається довільно, можливо, випадковим образом.

Вибір діапазону значень параметрів l_r і l_c обмежений знизу міркуваннями колізійної стійкості, а зверху – міркуваннями зручності й зменшення ймовірності помилки при диктуванні криптограми по голосовому каналі зв'язку.

$H(x)$ – криптографічна хеш-функція, значення якої усікається до необхідного розміру.

Параметри $\{c\}$ пар таблиці будуть секретними параметрами.

На клієнті зберігається таблиця запитів активації, що є підмножиною першого стовпця таблиці пар на сервері. Розмір таблиці – 2^{16} записів, тобто 0.5 – 1 Мб. Вибір підмножини здійснюється довільно.

Кафедра_КОПЗ_2022рік

Активація що захищається ПЗ здійснюється в такий спосіб. Клієнт здійснює збір прив'язочних значень обчислювального середовища E, зберігає його й потім хеширует цей набір значень у двійковий вектор $e = h(E)$ довжиною 16 біт. Для цього рекомендується використовувати криптографічний алгоритм хешування. Даний хеш буде індексом у таблиці запитів активації клієнта. Обраний запит $r[e]$ активації відправляється клієнтом на сервер. Сервер знаходить у своїй таблиці пар запит-підтвердження $c = H^{-1}(r[e])$ необхідне підтвердження й відправляє його клієнтові. Клієнт шляхом обчислення хеш-функції від підтвердження й порівняння результату із запитом активації $H(c) = r[e]$ переконується в автентичності підтвердження. Після чого клієнт зберігає підтвердження у своїй області даних.

Для перевірки прив'язки в обчислювальному середовищі, клієнт, аналогічно вищеописаному, здійснює збір прив'язочних значень обчислювального середовища *E*' і робить порівняння векторів *E* і *E*'. У випадку успішного порівняння, клієнт переконується у відповідності підтвердження активації збереженому еталону, тобто перевіряє рівність $H(c) = r[h(E)]$.

3.2 Розробка структурної схеми

Структурна схема розробленої системи зображена на рисунку 3.3. На ній показано структурні блоки, з яких складається система, та структурні взаємозв'язки між цими блоками.

Структурна схема складається з трьох основних блоків:

– Інтелектуальна мікропроцесорна карта з EEPROM.

– Програмне забезпечення на серверній частині.

– Банкомат (Картрідер).

Розглянемо ці блоки більш детально.

Кафедра_КОПЗ_2022рік

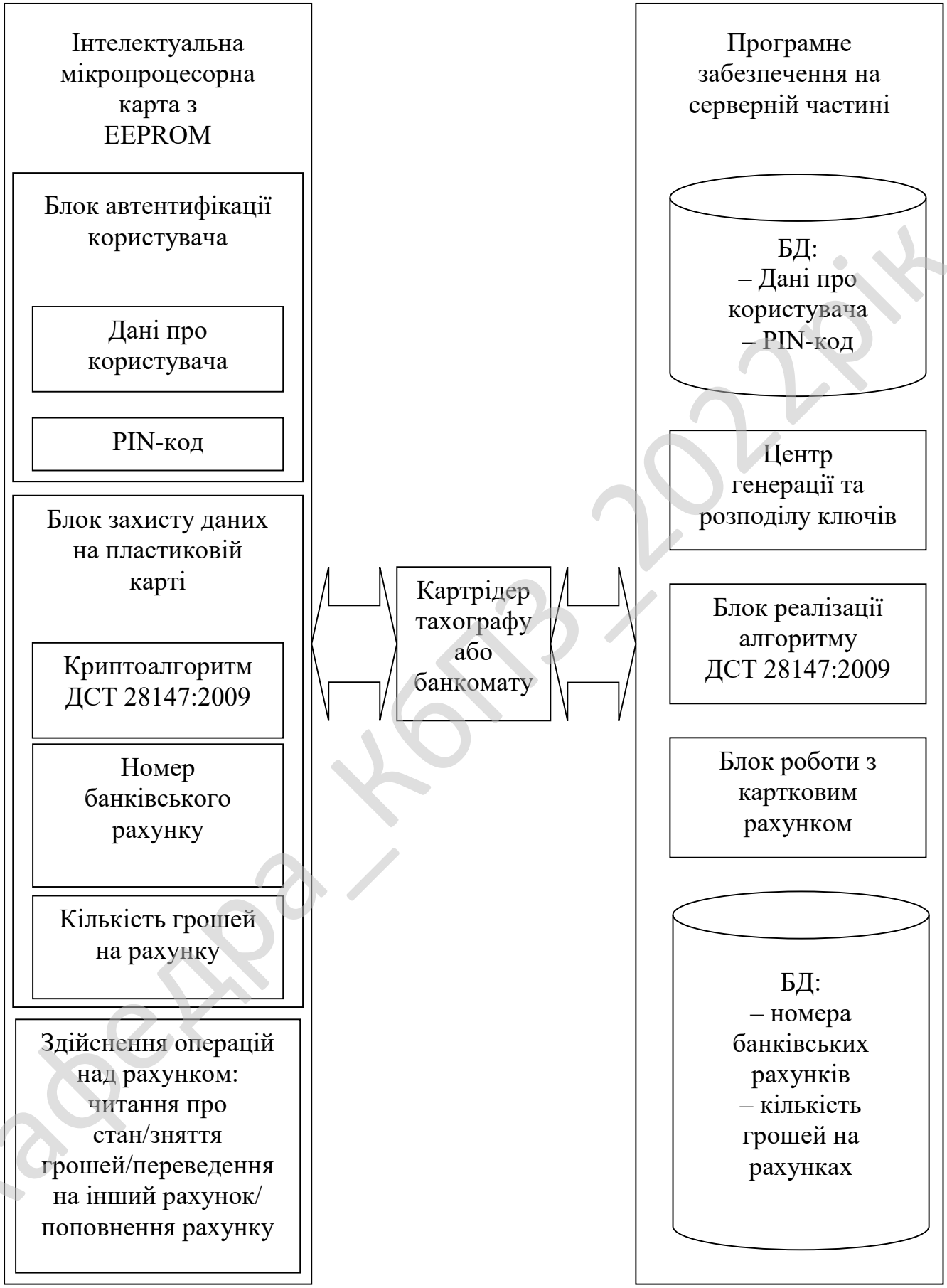

Рисунок 3.3 – Структурна схема системи

Кафедра_КбПЗ_2022рік

Інтелектуальна мікропроцесорна карта з EEPROM являє собою пластикову картку, яку можливо використовувати для операцій з грошима. EEPROM – (Electrically Erasable Programmable Read-Only Memory, електрично стираємий перепрограмувальний постійний запам'ятовувальний пристрій ЕСППЗП). Пам'ять такого типу може стиратися й заповнюватися даними кілька десятків тисяч разів. Використовується у твердотільних накопичувачах. Однією з різновидів EEPROM є флеш-память.

Структурно вона включає в себе наступні блоки:

– Блок автентифікації користувача.

– Блок захисту даних на пластиковій карті.

– Блок здійснення операцій над рахунком.

Блок автентифікації користувача включає в себе наступні дані:

– Дані про користувача – прізвище, ім’я та по батькові.

– PIN-код користувача.

Блок захисту даних на пластиковій карті включає в себе наступні складові:

– Номер банківського рахунку користувача.

– Кількість грошей на рахунку.

– Криптоалгоритм ДСТ 28147:2009, яким зашифровані перераховані вище дані користувача.

Блок здійснення операцій над рахунком включає в себе наступні операції:

– читання про стан рахунку;

– зняття грошей з рахунку;

– поповнення рахунку;

– переведення грошей на інший рахунок.

Розглянувши структурний склад інтелектуальної мікропроцесорної карти з EEPROM, перейдемо до розгляду іншої складової – програмного забезпечення на серверній частині.

Кафедра_КОПЗ_2022рік

Програмне забезпечення на серверній частині складається з наступних структурних блоків:

– База даних про користувача.

– Центр генерації та розподілу ключів.

– Блок реалізації алгоритму ДСТ 28147:2009.

– Блок роботи з картковим рахунком.

– База даних про рахунок.

База даних про користувача включає в себе наступні дані:

– Дані про користувача – прізвище, ім’я та по батькові.

– PIN-код користувача.

База даних про рахунок включає в себе наступні дані:

– Номер банківського рахунку користувача.

– Кількість грошей на рахунку.

Опишемо алгоритм ДСТ 28147:2009. Для захисту даних на мікропроцесорній карті запропоновано використовувати алгоритм ДСТ 28147:2009, що є класичним алгоритмом симетричного шифрування на основі мережі Фейстеля (Рисунок 3.4). Даний алгоритм шифрує інформацію блоками по 64 біта (такі алгоритми називаються "блоковими"). Зміст мережі Фейстеля полягає в тому, що блок шифруемої інформації розбивається на два або більше субблоків, частина яких обробляється за певним законом, після чого результат цієї обробки накладається (операцією побітового додавання за модулем 2) на необроблювані субблоки. Потім субблоки міняються місцями, після чого обробляються знову й т.д. певне для кожного алгоритму число раз – раундів.

$$L_i = R_{i-1}$$

$$R_i = L_i \oplus f(R_{i-1}, K_i)$$

Функція F проста. Спочатку права половина й і-ий підключ складаються за модулем 2^{32}. Потім результат розбивається на вісім 4-бітових значень, кожне з яких подається на вхід S-box. ДЕРЖСТАНДАРТ 28147 використовує вісім різних S-boxes, кожний з яких має 4-бітовий вхід і 4-бітовий вихід. Виходи всіх S-boxes

Кафедра_КОПЗ – 2022рік

поєднуються в 32-бітне слово, що потім циклічно зсувається на 11 бітів вліво. Нарешті, за допомогою XOR результат поєднується з лівою половиною, у результаті чого виходить нова права половина.

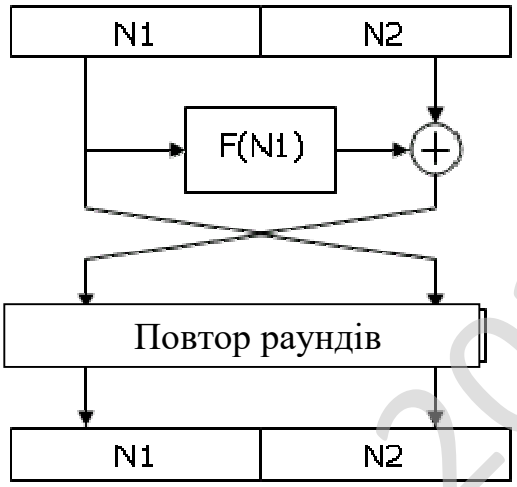

Рисунок 3.4 – Мережа Фейстеля

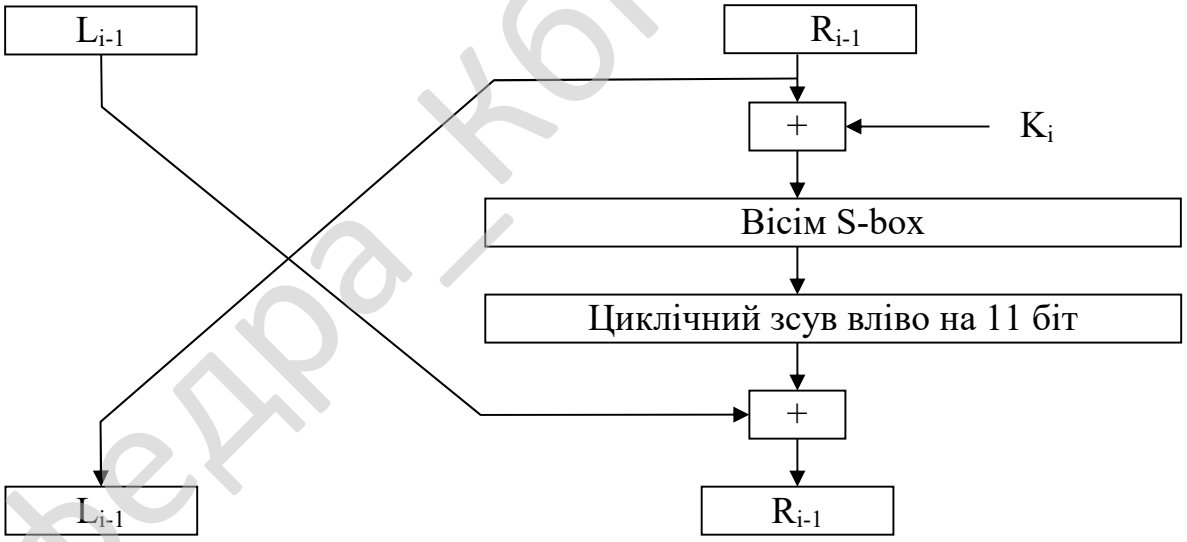

Рисунок 3.5 – і-ий раунд ГОСТ 28147:2009

Основна відмінність алгоритмів симетричного шифрування друг від друга складається саме в різних функціях обробки субблоків. Дана функція часто називається "основним криптографічним перетворенням", оскільки саме вона несе

Кафедра_КбПЗ_2022рік

основне навантаження при шифруванні інформації. Основне перетворення алгоритму ДСТ 28147:2009 є досить простим, що забезпечує високу швидкодію алгоритму; у ньому виконуються наступні операції (Рисунок 3.6).

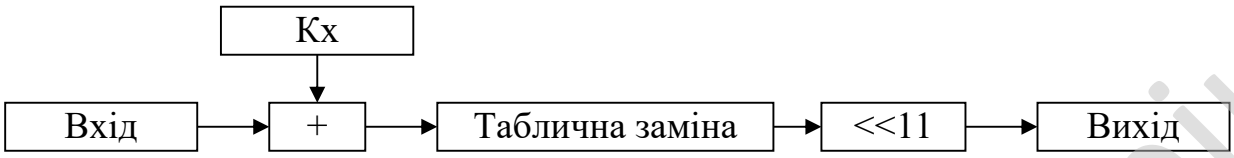

Рисунок 3.6 – Основне перетворення алгоритму ДСТ 28147:2009

1. Додавання субблоку з певним фрагменом ключа шифрування за модулем 2^{32}. Кх – це 32-бітна частина ("підключ") 256-бітного ключа шифрування, якому можна представити як конкатенацію 8 підключей: $К = К_0К_1К_2К_3К_4К_5К_6К_7$. Залежно від номера раунду й режиму роботи алгоритму (про їх – нижче), для даної операції вибирається один з підключей.

2. Таблична заміна. Для її виконання субблок розбивається на 8 4-бітних фрагментів, кожний з яких прогоняється через свою таблицю заміни. Таблиця заміни містить у певній послідовності значення від 0 до 15 (тобто всі варіанти значень 4-бітні фрагменти даних); на вхід таблиці подається блок даних, числове подання якого визначає номер вихідного значення. Наприклад, подається значення 5 на вхід наступної таблиці: "13 0 11 74 91 10 143 5 122 15 8 6". У результаті на виході виходить значення 9 (оскільки 0 заміняється на 13, 1 – на 0, 2 – на 11 і т.д.).

3. Побітове циклічне зрушення даних усередині субблока на 11 біт уліво.

3.3 Розробка функціональної схеми

Функціональна схема розробленої системи зображена на рисунку 3.7. розглянемо її більш детальніше.

Кафедра_КБПЗ_2022рік

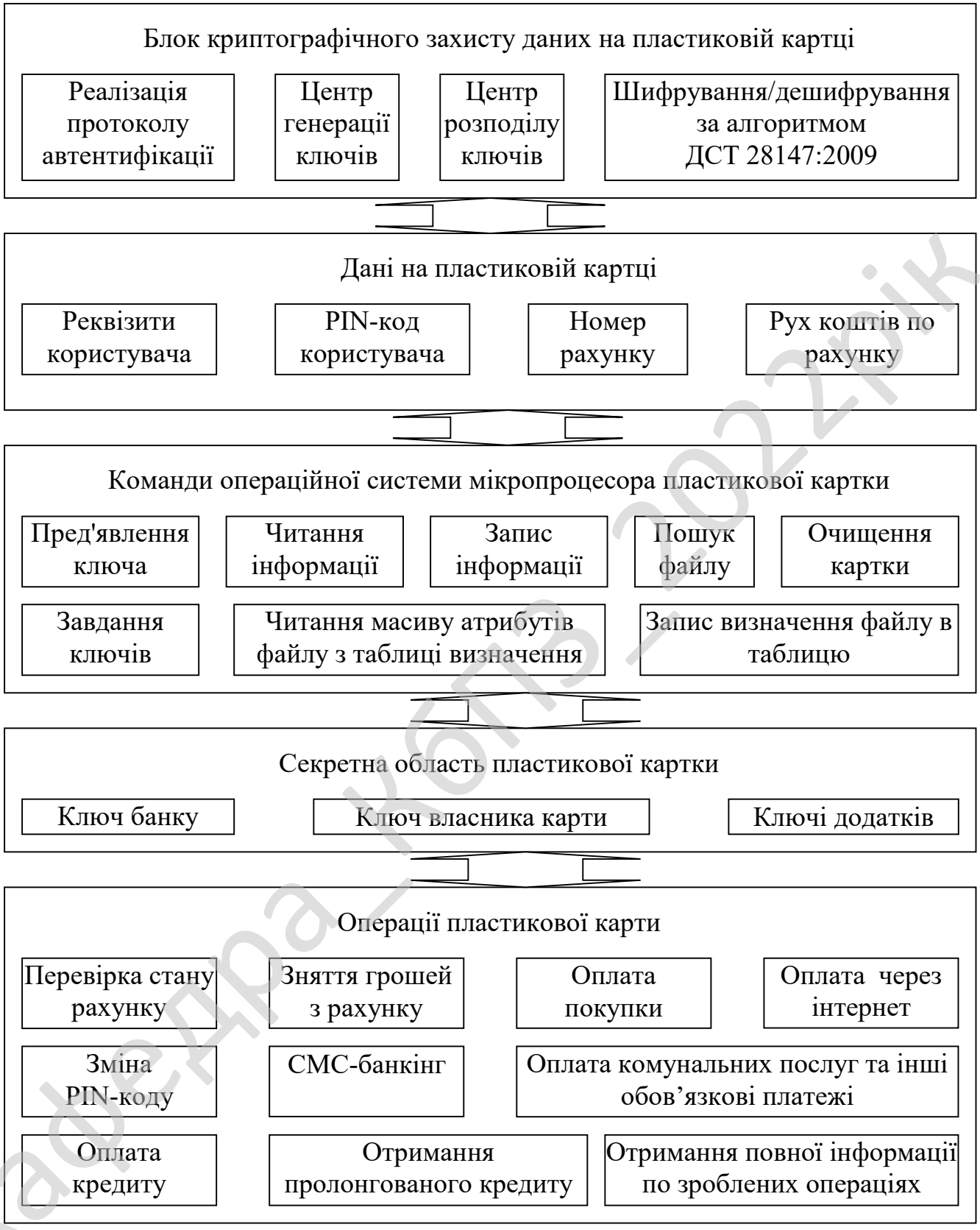

Рисунок 3.7 – Функціональна схема системи

Кафедра_КОПЗ_2022рік

З рисунку видно, що розроблена система функціонально складається з наступних частин:

– Блок криптографічного захисту даних на пластиковій картці.

– Дані на пластиковій картці.

– Команди операційної системи мікропроцесора пластикової картки.

– Секретна область пластикової картки.

– Операції пластикової карти.

Мікропроцесорна карта зроблена із пластику й містить мікросхему з мікропроцесором і різними запам'ятовувальними пристроями: ПЗП – для зберігання операційної системи, ОЗП – для виконання команд, ЕСППЗП – електрично-стираємий програмуємий постійний запам’ятовуючий пристрій, енергонезалежна пам'ять для зберігання прикладної інформації. ЕСППЗП розбито на дві області: секретну й користувальницьку. Секретна область недоступна для прикладних програм і призначена тільки для зберігання ключів. Користувальницька область організована аналогічно пам'яті на гнучких дисках. При ініціалізації мікросхеми карти формується таблиця визначення файлів, розташовувана на початку користувальницької області. Файли розташовуються в пам'яті від кінця до початку. Кожний файл розбитий на певне число записів фіксованої довжини. У більшості операційних систем кожний файл має наступні атрибути: початкова адреса, мітки захисту по читанню/запису, розширення захисту по читанню/запису, довжина запису, число записів, тип і ім'я файлу, що поточно записується, покажчик кінця файлу. Файли можуть бути послідовного й прямого доступу.

Блок криптографічного захисту даних на пластиковій картці включає в себе наступні складові:

– Реалізація протоколу автентифікації.

– Центр генерації ключів.

– Центр розподілу ключів.

– Шифрування/дешифрування за алгоритмом ДСТ 28147:2009.

Кафедра_КОПЗ_2022рік

Дані на пластиковій картці, включають наступні:

– Реквізити користувача.

– PIN-код користувача.

– Номер рахунку.

– Рух коштів по рахунку.

В операційній системі мікропроцесора передбачені наступні команди:

– пред'явлення ключа;

– читання масиву атрибутів файлу з таблиці визначення;

– читання інформації;

– запис інформації;

– пошук файлу;

– очищення картки;

– запис визначення файлу в таблицю;

– завдання ключів.

Ключі зберігаються в секретній області, передбачені три типи ключів:

– ключ банку;

– ключ власника карти;

– ключі додатків.

Файли можуть бути захищені цими ключами по читанню/запису.

Карти із криптографічною логікою використовуються в системах захисту інформації для прийняття особистої участі в процесі шифрування даних або вироблення криптографічних ключів, електронних цифрових підписів і іншої необхідної інформації для роботи системи.

Сфера застосування чіпових карт набагато ширше в порівнянні з фінансовою сферою, вони використовуються й у системах контролю доступу, і в охороні здоров'я (карти здоров'я), і страхуванні. Існують і телефонні карти, які також застосовуються для оплати, але у зв'язку зі специфічністю їх, звичайно, не варто було б ставити в один ряд із платіжними картами.

Кафедра_КбПЗ_2022рік

Операції пластикової карти:

– Перевірка стану рахунку.

– Зняття грошей з рахунку.

– Оплата покупки.

– Оплата через Інтернет.

– Зміна PIN-коду.

– СМС-банкінг.

– Оплата комунальних послуг та інші обов’язкові платежі.

– Оплата кредиту.

– Отримання пролонгованого кредиту.

– Отримання повної інформації по зроблених операціях.

Облік операцій із пластиковими картками:

1. Для видачі готівки по пластикових картках кредитні організації використовують банкомати, установлені як у будинку кредитної організації, так і поза нею. Облік коштів, використовуваних через банкомати, ведеться на спеціально відкритому рахунку "Кошти в банкоматах". Порядок заправлення банкоматів, контролю за використанням у них коштів здійснюється відповідно до нормативних документів Національного банку України по касовій роботі кредитних організацій. Заправлення банкоматів оформляється проводкою:

– Дебет рахунку по обліку коштів у банкоматах.

– Кредит рахунку по обліку каси кредитних організацій.

При розвантаженні банкоматів робиться зворотна проводка.

2. Розрахунки пластиковими картками відбуваються в межах засобів, внесених на спеціально відкриті для цієї мети рахунку. Зарахування засобів на зазначені рахунки оформляється проводкою:

– Дебет рахунків по обліку каси при внеску засобів фізичними особами готівкою або розрахункових (поточних) рахунків при перерахування засобів клієнтами кредитних організацій, або кореспондентського при надходженні засобів у безготівковому порядку від не клієнтів кредитних організацій, або

Кафедра_КОПЗ_2022рік

рахунку по обліку внесків, депозитів при перерахуванні засобів із цих рахунків, по особових рахунках власників внесків, депозитів.

– Кредит відповідних рахунків по обліку розрахунків з використанням пластикових карток, по особових рахунках їхніх власників.

З метою недопущення дебетового сальдо по рахунках юридичних і фізичних осіб, яке відбиває розрахунки пластиковими картками, може встановлюватися при укладанні договорів на незнижуваний залишок засобів на рахунку. За дотриманням установленого незнижуваного залишку засобів контроль здійснює структурний підрозділ, відповідальний за цю ділянку роботи.

Кредитні організації за домовленістю із клієнтами можуть установлювати для страховки ризику, створення за рахунок засобів клієнтів страхового фонду. Засоби цього фонду враховуються на окремих особових рахунках, що відкриваються по кожному клієнті на рахунках по обліку розрахунків за допомогою пластикових карток. Страхові суми використовуються в порядку, установленому договором. Видані картки списуються з позабалансового рахунку по обліку бланків суворої звітності, по відповідному особовому рахунку.

При припиненні розрахунків пластиковими картками й закритті рахунків по депонуванню засобів для цієї мети робляться зворотні проводки.

При цьому повернуті пластикові картки по позабалансовому рахунку не прибуткуються, а робляться фізично неплатоспроможними й містяться в документи дня.

3. Кредитні організації самостійно організують виготовлення пластикових карток. Ці операції, а також операції по видачі карток оформляються проводками.

3.1. Попередня оплата карток:

– Дебет рахунку по обліку розрахунків з постачальниками, підрядниками, покупцями, по особовому рахунку постачальника.

– Кредит кореспондентського рахунку.

3.2. Одержання карток:

– Дебет рахунку по обліку витрат.

Кафедра_КОПЗ_2022рік

– Кредит рахунку по обліку розрахунків з постачальниками, підрядниками, покупцями, по особовому рахунку постачальникові.

Одночасно отримані картки із прикладеними документами (комплект) прибуткуються по позабалансовому рахунку по обліку бланків суворої звітності в оцінці одна гривна за комплект у кореспонденції з кореспондуючим рахунком.

Якщо картки були отримані без попередньої оплати, то спочатку робиться проводка, зазначена в п.3.2., а потім проводка, вказівки в п.3.1.

4. При організації, проведенні роботи й здійсненні облікових операцій із пластиковими картками доцільно, щоб у банку були розроблені з урахуванням конкретних умов діяльності банку необхідні документи, такі як:

– положення про організацію системи безготівкових розрахунків з використанням пластикових карток;

– технологія робіт, пов'язана з емісією власних карток і обробкою інформації з карток; правила користування карткою;

– положення про порядок роботи з автоматичним терміналом безготівкової оплати при обслуговуванні карток у пунктах видачі наявних засобів;

– положення про порядок роботи зі спеціальною апаратурою при обслуговуванні карток у пункті видачі наявних засобів.

Роботу з виконання цих документів повинний організовувати й проводити спеціалізований підрозділ банку, або окремі працівники. Цей підрозділ повинний підтримувати необхідні контакти з банками-кореспондентами, що ведуть операції із пластиковими картками банку, одержувати від банків необхідні документи, розробляти їх, направляти відповідні документи в банки-кореспонденти, здійснювати спостереження за джерелами засобів на рахунках банку для здійснення операцій по картках. За результатами розгляду документів підрозділ передає службі бухгалтерського обліку службові записки за підписом працівників банку, виділених керівництвом банку для відбиття операцій в обліку.

Кафедра_КОПЗ_2022рік

5. Операції по використанню засобів за допомогою пластикових карток відбиваються проводками.

5.1. Одержання готівки через банкомати:

– Дебет відповідних рахунків для розрахунків пластиковими картками, по особових рахунках їхніх власників.

– Кредит рахунку по обліку коштів у банкоматах, по особових рахунках відповідних банкоматів.

5.2. Використання засобів, коли всі операції по пластикових картках здійснює сама кредитна організація:

– Дебет відповідних рахунків для розрахунків пластиковими картками, по особових рахунках їхніх власників.

– Кредит рахунків розрахункових (поточних) організацій, що продала товари, що зробила послуги, якщо ці організації обслуговуються кредитною організацією, або кореспондентського рахунку, якщо зазначені організації обслуговуються іншим банком.

5.3. Якщо операції по використанню пластикових карток веде інша кредитна організація, що має необхідне встаткування, то розрахунки по цих операціях між кредитними організаціями ведуться через рахунки, що відкриваються друг у друга на балансових рахунках: "Кореспондентські рахунки кредитних організацій-кореспондентів" і "Кореспондентські рахунки в кредитних організаціях-кореспондентах". На цих рахунках відкриваються окремі особові рахунки для обліку цих операцій. Взаємини по веденню цих рахунків і інших операцій із пластиковими картками визначаються договорами, що містяться між кредитними організаціями.

5.3.1. Перерахування засобів у кредитну організацію для ведення операцій із пластиковими картками здійснюється проводкою:

– Дебет кореспондентського рахунку в кредитних організаціях-кореспондентах, по окремому особовому рахунку.

Кафедра_КОПЗ_2022рік

– Кредит кореспондентського рахунку кредитної організації, з якого перераховуються засоби.

Одержання засобів кредитною організацією для ведення операцій із пластиковими картками оформляється зворотною проводкою.

5.3.2. Використання засобів шляхом пластикових карток (у кредитній організації, що веде ці операції із клієнтами-одержувачами засобів):

– Дебет кореспондентського рахунку в кредитних організаціях-кореспондентах по окремому особовому рахунку.

– Кредит рахунків розрахункових (поточних) організацій, що продала товари, що зробила послуги, якщо рахунки ведуться в кредитній організації, або кореспондентського рахунку кредитної організації, якщо рахунку одержувачів засобів ведуться в інших кредитних організаціях.

5.3.3. Відбиття операцій у кредитній організації, що веде рахунок власників пластикових карток:

– Дебет відповідних рахунків для розрахунків пластиковими картками, по особових рахунках їхніх власників.

– Кредит кореспондентського рахунку в кредитних організаціях-кореспондентах, по окремому особовому рахунку.

6. Нарахування відсотків по рахунках для розрахунків пластиковими картками, а також сплата у встановлених випадках комісії власникам карток:

– Дебет рахунку обліку витрат кредитної організації.

– Кредит відповідних рахунків, по особових рахунках власників карток.

7. Сплата комісії з операцій із пластиковими картками на користь кредитної організації, що веде ці операції.

7.1. У кредитній організації, що сплачує комісію:

– Дебет рахунку по обліку витрат.

– Кредит кореспондентського рахунку в кредитних організаціях, по окремому особовому рахунку.

7.2. У кредитній організації, що одержує комісію:

Кафедра_КбПЗ_2022рік

– Дебет кореспондентського рахунку кредитних організацій-кореспондентів, по окремому особовому рахунку.

– Кредит рахунку по обліку доходів.

8.Стягнення, у встановлених випадках, комісії із власників карток:

– Дебет відповідних рахунків по обліку розрахунків пластиковими картками, по особових рахунках їхніх власників.

– Кредит рахунку по обліку доходів.

9. Відновлення засобів неправильно списаних по картках.

– Дебет рахунку по обліку розрахунків з іншими дебіторами й кредиторами по особовому рахунку власника картки – відкривається окремий особовий рахунок.

– Кредит відповідних рахунків по обліку розрахунків пластиковими картками, по особових рахунках їхніх власників.

Структурний підрозділ кредитної організації, що займається пластиковими картками, вживає необхідних заходів до відновлення банку засобів.

10. Використання пластикових карток у розрахунках банку по придбанню матеріальних цінностей і послуг для своїх потреб.

10.1. Для розрахунків за допомогою пластикових карток, засоби на рахунку по обліку цих розрахунків не депонуються. Видана посадовій особі кредитної організації пластикова картка в сумі встановленого ліміту прибуткується по позабалансовому рахунку обліку різних цінностей і документів, відісланих і виданих під звіт, по особовому рахунку.

10.2. Використання засобів за допомогою картки:

– Дебет відповідного рахунку по обліку господарських матеріалів, або рахунку по обліку витрат за зроблені послуги.

– Кредит рахунку по обліку розрахунків з постачальниками, підрядниками й покупцями: по особовому рахунку постачальника, якщо він має рахунок у цьому банку, або рахунок розрахунків з іншими дебіторами й кредиторами, якщо

Кафедра_КбПЗ_2022рік

рахунок постачальника в іншому банку, з наступним перекладом засобів з кореспондентського рахунку.

Одночасно сума, використана за допомогою картки, списується з позабалансового рахунку.

3.4 Розробка діаграми процесів

Діаграма процесів розробленої системи зображена на рисунку 3.8. З неї видно, що першим процесом, який завантажується у системі є процес автентифікації користувача.

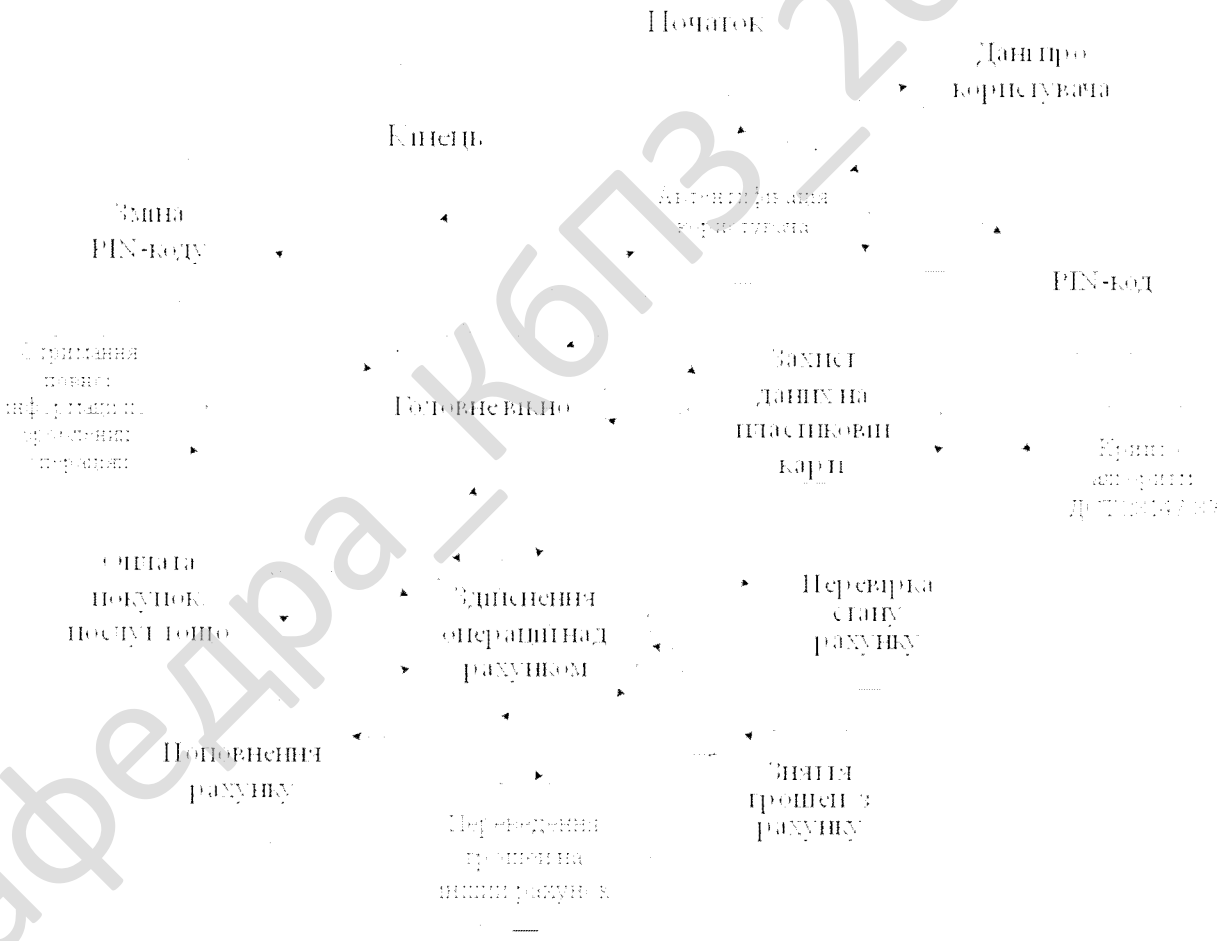

Рисунок 3.8 – Діаграма взаємодії процесів

Кафедра_КбПЗ_2022рік

Цей процес взаємодіє з наступними процесами:

– Дані про користувача.

– PIN-код.

– Головне вікно.

Процес запуску головного вікна взаємодіє з наступними процесами:

– Зміна PIN-коду.

– Здійснення операцій над рахунком.

– Захист даних на пластиковій карті, який, у свою чергу, взаємодіє з процесом криптоалгоритму ДСТ 28147:2009.

Процес здійснення операцій над рахунком взаємодіє з наступним процесами, які відбуваються у системі, розробленій у результаті виконання магістерської роботи:

– Отримання повної інформації по зроблених операціях.

– Оплата покупок, послуг тощо.

– Поповнення рахунку.

– Переведення грошей на інший рахунок.

– Зняття грошей з рахунку.

– Перевірка стану рахунку.

Таким чином, розглянувши опис системи, структурну, функціональну схеми системи, та діаграму взаємодії процесів перейдемо до опису блок-схем основної програми, та підпрограм, які використовуються, для реалізації системи.

Кафедра_КОПЗ_2022рік

4 РЕАЛІЗАЦІЯ РОБОТИ. РОЗРАХУНКИ І ЕКСПЕРИМЕНТАЛЬНІ ДАНІ, ЩО ПІДТВЕРДЖУЮТЬ ВІРНІСТЬ ПРОЕКТНИХ ТА ПРОГРАМНИХ РІШЕНЬ

4.1 Блок-схеми та опис алгоритмів функціонування системи

Розглянемо алгоритм роботи основної програми. Його блок-схема зображена на рисунку 4.1.

З рисунку видно, що після запуску програми спочатку відбувається вивід інтерфейсу користувача програми. Потім здійснюється читання даних з карти.

Після цього користувач вводить PIN-код.

Якщо PIN-код невірний то виводиться попередження, й відбувається перехід до вікна введення PIN-коду.

Якщо PIN-код вірний, тоді відбувається виконання наступних дій:

– Дешифрування даних на карті алгоритмом ДСТ 28147:2009.

– Виведення списку можливих дій.

Після цього користувач може обрати одну з наступних дій:

– Перевірка стану рахунку.

– Зняття грошей.

– Переведення грошей.

– Поповнення рахунку.

– Оплата.

– Вихід.

Якщо обрано перевірку стану рахунку, тоді відбувається виведення стану рахунку.

Якщо обрано зняття грошей, тоді відбувається зняття грошей з рахунку.

Якщо обрано переведення грошей, тоді відбувається переведення грошей на інший рахунок.

Кафедра_КОПЗ_2022рік

Якщо обрано поповнення рахунку, тоді відбувається поповнення поточного рахунку.

Якщо обрано оплату, тоді відбувається оплата покупок, послуг тощо.

Якщо обрано вихід з програмного продукту, тоді відбувається шифрування даних на карті алгоритмом ДСТ 28147:2009.

На цьому програма закінчує свою роботу.

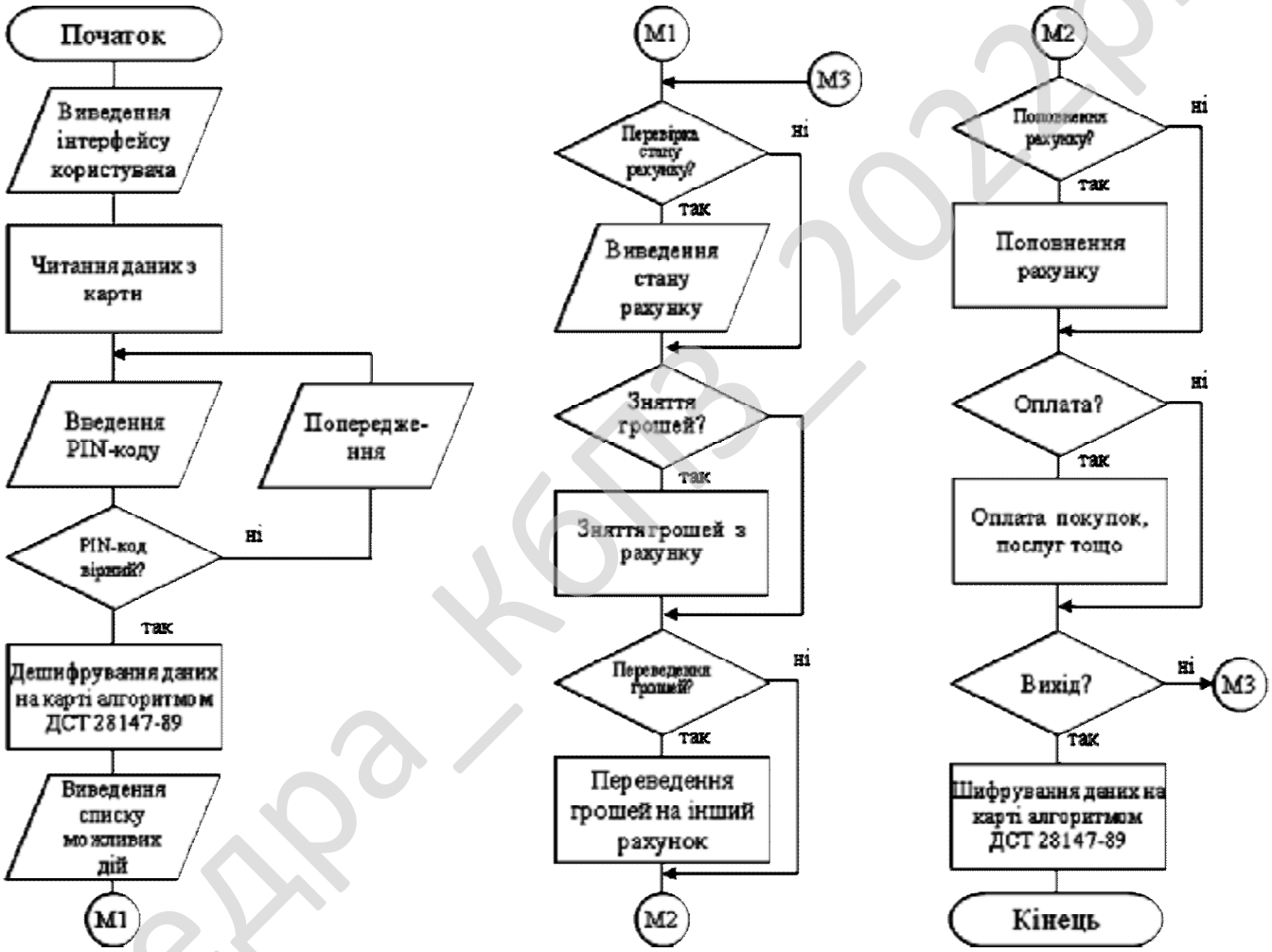

Рисунок 4.1 – Блок-схема основної програми

На рисунку 4.2 зображено блок-схема підпрограми шифрування ДСТ 28147:2009. Вона працює наступним чином. Виконується послідовне виконання наступних ітерацій:

– Розбиття даних для шифрування на 64-бітні блоки.

Кафедра_КБПЗ_2022рік

– Розбиття 256-бітного ключа на вісім 32-бітних підключів.

– Ініціалізація S-блоків.

– Вибір раундового ключа.

– Трансформація раундового ключа.

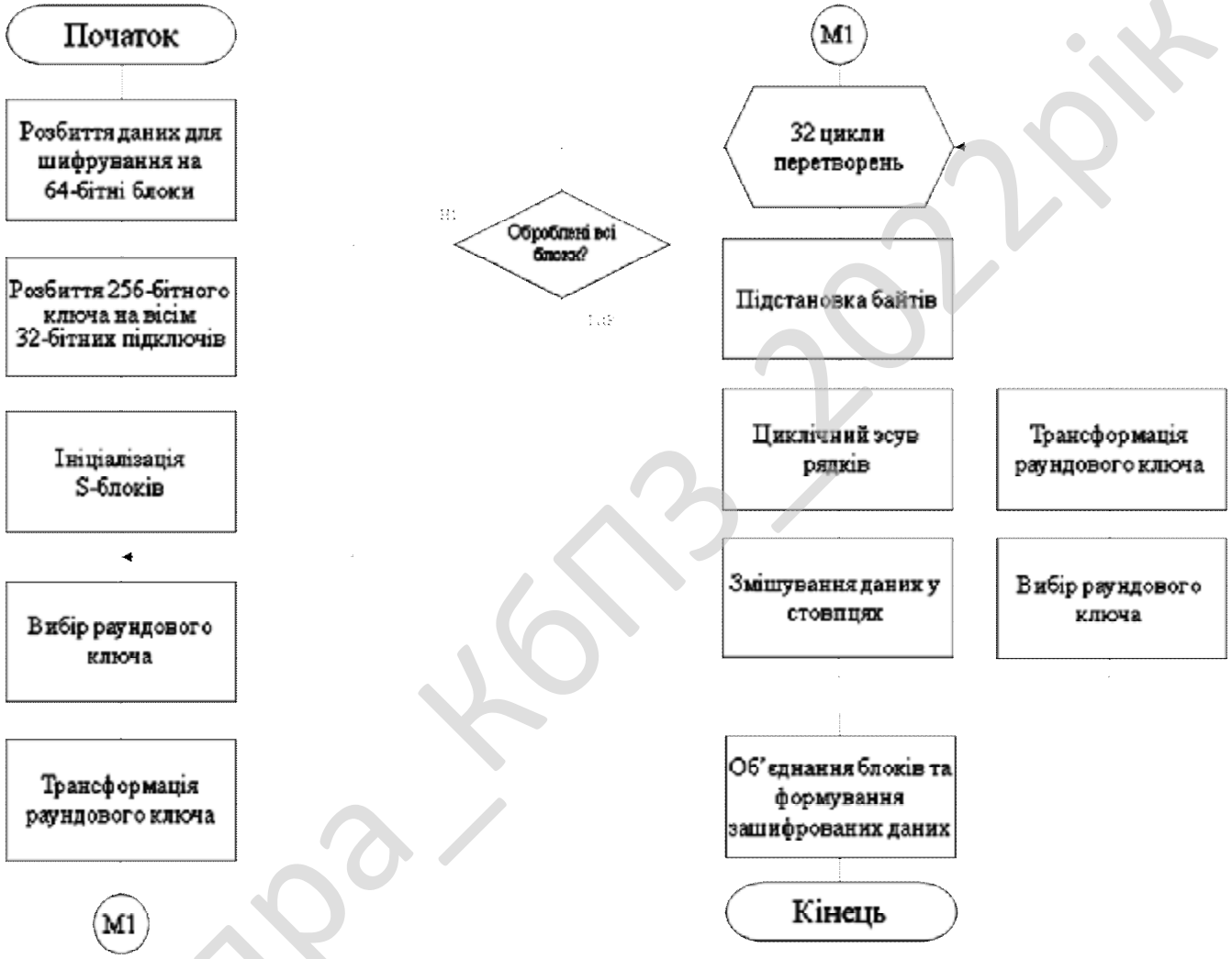

Рисунок 4.2 – Блок-схема підпрограми шифрування ДСТ 28147:2009

Після цього починаються 32 цикли перетворень:

– Підстановка байтів.

– Циклічний зсув рядків.

– Змішування даних у стовпцях.

– Вибір раундового ключа.

Кафедра_КбПЗ_2022рік

– Трансформація раундового ключа.

Коли цикл закінчується, тоді програма визначає чи оброблені усі дані.

Якщо усі дані оброблені, то відбувається об’єднання блоків та формування зашифрованих даних.

На цьому підпрограма закінчує роботу.

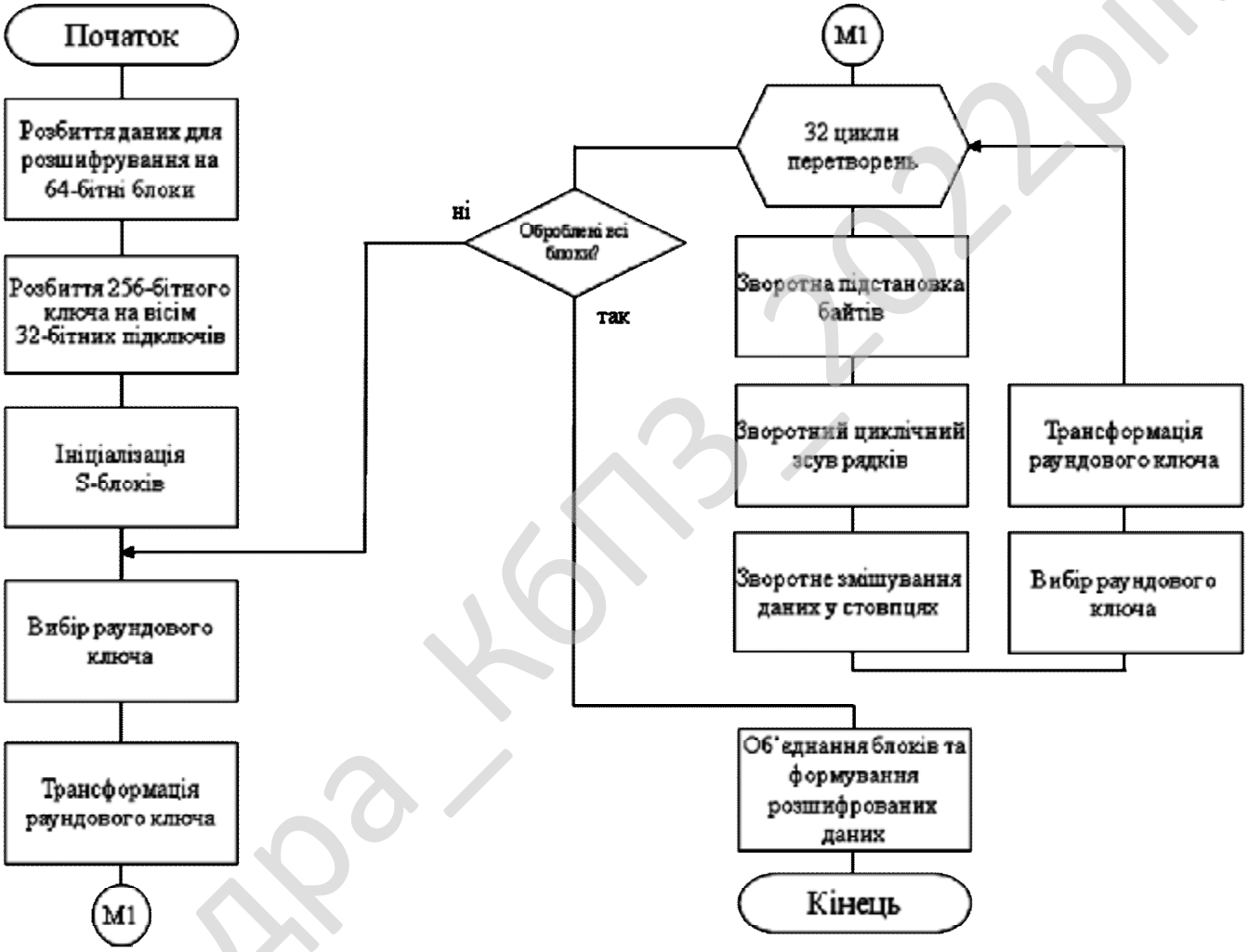

Рисунок 4.3 – Блок-схема підпрограми дешифрування ДСТ 28147:2009

На рисунку 4.3 зображено блок-схему підпрограми дешифрування ДСТ 28147:2009. Вона працює наступним чином. Виконується послідовне виконання наступних ітерацій:

– Розбиття даних для розшифрування на 64-бітні блоки.

Кафедра_КбПЗ_2022рік

– Розбиття 256-бітного ключа на вісім 32-бітних підключів.

– Ініціалізація S-блоків.

– Вибір раундового ключа.

– Трансформація раундового ключа.

Після цього починаються 32 цикли перетворень:

– Зворотна підстановка байтів.

– Зворотний циклічний зсув рядків.

– Зворотне змішування даних у стовпцях.

– Вибір раундового ключа.

– Трансформація раундового ключа.

Коли цикл закінчується, тоді програма визначає чи оброблені усі дані.

Якщо усі дані оброблені, то відбувається об’єднання блоків та формування розшифрованих даних.

На цьому підпрограма закінчує роботу.

```
{-------------------------
 Реалізація універсального циклу шифрування алгоритму
 криптографічного перетворення ДСТ 28147:2009.
----------------------------
>Параметри  при  виклику:
   EAX=N1, EDX=N2;
   ESI – адреса першого елемента ключа;
   EBX – адреса таблиці замін;
   CX    – число основних кроків
>Результати на виході:
   EDX=N1, EAX=N2 для циклів 32-З, 32-Р;
   EAX=N1, EDX=N2 для циклу 16-З;
}
procedure TGOSTEncryption.Gost386;
asm
    {Внутрішній  цикл  роботи  П/П}
    {1. Початок  циклу  й  збереження  старого  N1}
@iloop:  push      EAX
    {2. Додавання  до  S  ключа  за модулем  2^32}
         add       EAX,[ESI] { додати    ключ}
         add       ESI,4     { наступний  елемент ключа}
    {3. Поблочна заміна в S з обертанням на 8 біт уліво}
```

Кафедра_КБПЗ_2022рік

```
            xlat                { перекодування  байта}
            ror       EAX,8     { AL <- наступний байт}
            add       ebx,256   { наступний вузол замін}
            xlat                { перекодування  байта}
            ror       EAX,8     { AL <- наступний байт}
            add       ebx,256   { наступний вузол замін}
            xlat                { перекодування  байта}
            ror       EAX,8     { AL <- наступний байт}
            add       ebx,256   { наступний вузол замін}
            xlat                { перекодування  байта}
            sub       ebx,3*256 { BX -> 1-й вузол замін}
        {4. Довернення  S  на  3  біти  вліво}
            rol       EAX,3
        {5. Обчислення  нових  значень  N1,N2}
            xor       EAX,EDX
            pop       EDX
        { Завершення  внутрішнього  циклу}
            loop      @iloop
    end;
    var vesi,vedi,veax,vebx,vecx,vedx:DWORD;
    procedure PushAll;
    asm {Збереження регістрів відповідно до угод}
        mov vesi,esi
        mov vedi,edi
        mov veax,eax
        mov vebx,ebx
        mov vecx,ecx
        mov vedx,edx
    end;
    procedure PopAll;
    asm {Відновлення регістрів}
        mov esi,vesi
        mov edi,vedi
        mov eax,veax
        mov ebx,vebx
        mov ecx,vecx
        mov edx,vedx
    end;
    {-------------------------
     Генерація масиву криптографічної гами згідно
     криптоалгоритму ДСТ 28147:2009.
    ---------------------------
```

Кафедра_КбПЗ_2022рік

```
        Зауваження:
        1. Дана процедура виробляє гаму  блоками по 8
        байтів, що відбиває властивість використаного алго-
        ритму бути блоковим шифром.
       ----------------------------
       }
       procedure TGOSTEncryption.GenerateGamma(dst:pointer;noblocks:DWORD);
       var s1,s2:Item;
       begin
            s1:=0;s2:=0;
            GenerateGamma(dst,noblocks,s1,s2);
       end;
       procedure TGOSTEncryption.GenerateGamma(dst:pointer;noblocks:DWORD;var
s1,s2:Item);
       const
       // визначаємо константи C1,C2 за ДСТ
           C1=$01010101;
           C2=$01010104;
       var
           k:TPExtEncKey;  // адреса ключа
           c:TPExtTable;  // адреса таблиці замін
           l:DWORD;
           cur:pointer;
       begin
           k:=@ExtKey;
           c:=@ExtChTab;
           l:=noblocks;
           cur:=dst;
           ExpandKey(@Key,k,8,4,KeyMask);
       asm
                call      PushAll
                cmp       l,0
                je        @ex
                                        // настроювання регістрів
                mov       EBX,c         // EBX <- адреса таблиці замін
                mov       edi,s2
       @circle: mov       esi,s1        // Додавання  констант
                mov       EAX,[esi]
                mov       EDX,[edi]
                add       EAX,C1        // зміна S1, s2:
                adc       EDX,C2        //    S1=(S1+C1) mod 2^32
                                        //    S2=(S2+C2) mod (2^ 32-1)
```

Кафедра_КбПЗ_2022рік

```
                adc       EDX,0          // якщо був перенос,
                                         // треба додати 1 до результату
                mov       [esi],EAX      // заносимо нові
                mov       [edi],EDX      // значення S1,S2 на їхнє місце
                                         // Готовимо регістри для виклику циклу
32-З;
                mov       ESI,k          // ESI <- адреса ключа
                mov       ECX,32         // CX <- число основних кроків
                call      Gost386        // крок простої заміни
                                         // Використання згенерованого блоку гами
                mov       esi,cur
                mov       [ESI],EDX      // Заносимо 8-байтний блок
                mov       [ESI+4],EAX    // гами в область призначення
                add       cur,8          // корекція покажчика
                dec       l              // корекція лічильника
                jnz       @circle        // на генерацію нового блоку
        @ex:    call      PopAll
        end;
        end;
        procedure TGOSTEncryption.ApplyGamma(src:pointer;gamma:pointer;len:DWORD);
//Застосовуємо гаму
        asm
                call      PushAll
                mov       esi,src
                mov       edi,gamma
                mov       ecx,len
                cmp       ecx,0
                je        @ex
        @l1:    mov       al,[edi]
                xor       [esi],al
                inc       esi
                inc       edi
                loop      @l1
        @ex:    call      PopAll
        end;
        {-------------------------
         Виробіток імітовставки для масиву даних згідно
         криптоалгоритму ДСТ 28147:2009.
        ---------------------------
        }
        procedure TGOSTEncryption.GenerateImito(src:pointer;noblocks:DWORD);
        begin
```

Кафедра_КбПЗ_2022рік

```
            GenerateImito(src,noblocks,0,0);
        end;
        procedure
TGOSTEncryption.GenerateImito(src:pointer;noblocks:DWORD;s1,s2:Item);
        const ExtMask:DWORD=$2;
        var
            l:DWORD;
            pt:TPExtTable;
            pk:TPExtEncKey;
            p1,p2:^DWORD;
        begin
            ExpandKey(@Key,@ExtKey,8,2,ExtMask);
            l:=noblocks;
            pt:=@ExtChTab;pk:=@ExtKey;
            p1:=@I1;p2:=@I2;
        asm
                call      PushAll
                mov       EAX,s1          // Завантаження початкового значення
контрольної комбінації
                mov       EDX,s2
                                          // Початкове завантаження покажчиків
                mov       EBX,pt  // EBX <- адреса ТЗ
                mov       EDI,src         // EDI <- адреса даних
                cmp       l,0
                je        @ex
                                          // Додаємо блок даних
        @circle: xor      EAX,[EDI]       // Гаммуємо
                xor       EDX,[EDI+4]     // 8-байтный
                                          // Виклик простої заміни
                mov       ECX,16          // CX <- число основних кроків
                mov       ESI,pk          // ESI <- адреса ключа
                call      Gost386
                                          // Організація циклу
                add       EDI,8           //корекція адреси призначення
                dec       l
                jnz       @circle         // робимо цикл
        @ex:    mov       esi,p1
                mov       [esi],eax       // заносимо нові значення
                mov       esi,p2
                mov       [esi],edx
                call      PopAll
        end;
```

Кафедра_КбПЗ_2022рік

```
        end;
        {-------------------------
         Побудова розширеного ключа шифрування з однократ-
         ного ключа для шифру типу ДСТ 28147:2009.
         ** Схема розширення задається маскою розширення.
         ** Маска розширення може мати розмір 1..32767 біт.
        ----------------------------
        }
        procedure TGOSTEncryption.ExpandKey
(src:TPEncKey;dst:TPExtEncKey;KeyLength:DWORD;
                        KeyRepeat:DWORD;RepeatMask:DWORD);
        asm
                 call      PushAll
                                        // Настроювання регістрів
                 mov       ESI,src         // ESI=адреса джерела ключа
                 mov       EDI,dst         // EDI=адреса призначення
                 mov       EBX,KeyLength  // BX <- адреса кінця ключа
                 shl       EBX,2
                 add       ebx,esi
                 sub       ebx,4
                 xor       EAX,EAX         // AX=0
                 cld
                                        // Перевірка умови завершення циклу
        @iloop:  cmp           EAX,KeyRepeat  // потрібне число зроблене ?
                 jge           @ex            // якщо так, вихід
                 mov           ECX,KeyLength  // CX=довжина ключа
                 bt            RepeatMask,AX  // CF=черговий біт маски
                 jc          @Rev             // CF=1 -> реверс
                                       // Повторення елементів ключа в прямому
порядку
                 mov           ESI,src         // ESI=зсув ключа
                 rep       movsd          // копіюємо ключ
                 jmp       @Incr          // на рахунок повтор. ключа
                                        // Повторення елементів ключа у
зворотному порядку
        @Rev:    mov       ESI,EBX        // SI=адреса кінця ключа
        @rl:     movsd                          // слова ключа
                 sub             ESI,8          // до попереднього елемента
                 loop      @rl            // організація циклу
        @Incr:   inc       AX             // лічильник повторів ключа
                 jmp       @iloop
        @ex:     call      PopAll
```

Кафедра_КБПЗ_2022рік

```
end;
{-------------------------
 Шифрування масиву даних у режимі простої заміни
 згідно криптоалгоритму ДСТ 28147:2009.
----------------------------
 Зауваження:
 1.Зашифрування або  розшифрування задається передачею
 відповідної  адреси  ключа – порядок  елементів
 у  ключах  зашифрування  й  розшифрування  взаємно
 зворотний.
2.Відповідно  ДО ДСТУ 28147:2009  цей модуль може використовуватися
 тільки  для шифрування  ключової інформації
 (і  синхропосилки  для  гаммування).
}
procedure TGOSTEncryption.simple(src,dst:pointer;noblocks:DWORD);
// Шифрування простою заміною
var l:DWORD;
    pt:TPExtTable;
    pk:TPExtEncKey;
begin
    l:=noblocks;
    pt:=@ExtChTab;
    pk:=@ExtKey;
asm
         call      PushAll
                                  //Початкове завантаження покажчиків
         mov       ebx,pt// BX <- адреса ТЗ
         mov       esi,src        //source address
         mov       edi,dst        // address призначення
                                  // Завантаження блоку даних
@circle: mov       EAX,[esi]      // EAX <- S1
         mov       EDX,[esi+4]    // EDX <- S2
                                  // Виклик простої заміни;
         mov       ecx,32         // CX <- число основних кроків
         push      esi
         mov       esi,pk// ESI <- адреса ключа
         call      Gost386        //     <крок простої заміни>
         pop       esi
                                  // Запис  результату  на  місце
         mov       [edi+4],EAX    // Заносимо результат
         mov       [edi],EDX      //   на його місце
                                  // Організація циклу
```

Кафедра_КбПЗ_2022рік

```
            add     edi,8           // корекція адреси призначення
            add     esi,8           //go to next element in the source
            dec     l               // корекція лічильника  б.
            jnz     @circle         // робимо цикл
            call    PopAll
    end;
    end;
    {-------------------------
     Побудова розширеної таблиці замін (1024 байт) з
     таблиці замін (128 байт) алгоритму  ДСТ 28147:2009.

    ----------------------------
    }
    procedure TGOSTEncryption.Set128Table(p:TP128Table);
    var pt:TPExtTable;
    begin
        pt:=@ExtChTab;
    asm
            call    PushAll
            mov     ESI,p           // ESI->  джерело
            mov     EDI,pt          // EDI->  приймач
            cld                     // Цикл  по  блоках  розширеної  таблиці
замін
            mov     ecx,4           // CX <- число блоків великий ТЗ
    @blocks:mov     EBX,ESI         // BX<-адреса  початку  лінії
            add     EBX,10h         //     старших байтів блоку
                                    // Цикл  по  лініях  блоку
            push    eCX             // зберегти  лічильник блоків
            mov     eCX,16          // завантажити  лічильник  ліній
    @lines: push    ESI             // збереження показника поточного  блоку
            mov     AH,[EBX]        // AH <- старший    напівбайт
            push    eCX             // зберегти   лічильник ліній
            mov     CL,4            // зрушення напівбайта  на місце
            shl     AH,CL           //     старшого  напівбайта
                                    // Цикл  по  байтах  лінії
            mov     ECX,16          // завантажити лічильник  байтів
    @bytes: lodsb                   // завантажити черговий  байт
            or      AL,AH           // додати старший напівбайт
            stosb                   // ...  і записати результат
            loop    @bytes          // цикл   по  байтах   лінії
                                    // Перевірка  циклу  по  лініях
            pop     eCX             // відновлюємо лічильник  ліній
```

Кафедра_КбПЗ_2022рік

```
        pop     ESI             // відновлюємо вказівник тек. блоку
        inc     EBX             // просунути покажчик байт
        loop    @lines          // цикл  по рядках  таблиці
                                // Перевірка  циклу  по  блоках
        pop     eCX             // відновлюємо лічильник блоків
        add     ESI,20h         // просунути  вказівник блоку
        loop    @blocks         // цикл по блоках
        call    PopAll
end;
end;
```

4.2 Захист розробленого програмного забезпечення

Для захисту розробленого програмного забезпечення запропоновано використовувати алгоритм PRESENT – окремий випадок SP-мережі й складається з 31 раунду. Довжина блоку становить 64 біта, а ключі підтримуються в 2 варіантах, 80- і 128-бітні. Такого рівня захисту повинно цілком вистачати для низькозахищених додатків, звичайно використовуваних для розгортання на основі тегів, а крім того, що важливіше, PRESENT багато в чому збігається своїми конструктивними особливостями з потоковими шифрами проекту estream, заточеними на ефективну реалізацію в залозі, що дозволяє нам адекватно порівнювати їх.

Вимоги з безпеки й експлуатаційні властивості 128-бітних версій надані в додатку до оригінальної статті.

Кожний з 31 раундів складається з операції XOR, щоб увести ключ K_i для $1 \leq i \leq 32$, де K_{32} використовується для «відбілювання» ключа, лінійної побітової перестановки й нелінійного шару заміщення (або, попросту говорячи, збільшення стійкості шифрування . Нелінійний шар використовує роздільні 4-бітні S-блоки, які застосовуються паралельно 16 раз на кожному раунді. Шифр, описаний псевдо-кодом представлено на рисунку 4.4.

Кафедра_КОПЗ_2022рік

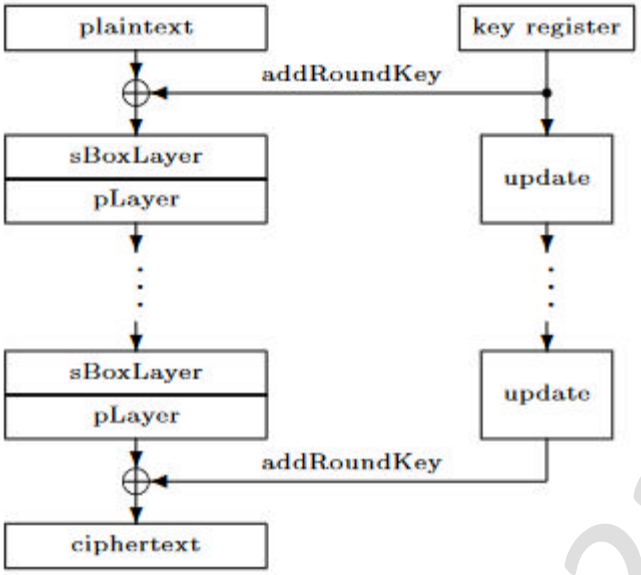

Рисунок 4.4 – Шифр PRESENT

Тепер кожна стадія визначається по черзі. Обґрунтування конструкції наведені нижче, а біти всюди нумеруються з нуля, починаючи із правого в блоці або слові.

Додавання раундового ключа (addRoundKey)

Заданий раундовий ключ $K_i = k^i_{63} \ldots k^i_0$, де $1 \le i \le 32$, а так само поточний стан $b_{63} \ldots b_0$. Додавання раундового ключа до поточного стану відбувається за модулем 2 ($b_j = b_j \oplus k^i_j$, де $0 \le j \le 63$).

Шар S-блоків (sBoxlayer)

Використовувані в PRESENT S-блоки відображають 4-бітні блоки в 4-бітні блоки. Дія цього блоку в шістнадцятковій системі числення наведене в наступній таблиці.

x	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
$S[x]$	C	5	6	B	9	0	A	D	3	E	F	8	4	7	1	2

Рисунок 4.5 – Дія блоку в шістнадцятковій системі числення

Кафедра_КБПЗ_2022рік

Для шару S-блоків поточний стан $b_{63}\ldots b_0$ представляє із себе 16 4-бітних слів $w_{15}\ldots w_0$, де $w_i = b_{4*i+3} \,||\, b_{4*i+2} \,||\, b_{4*i+1} \,||\, b_{4*i}$ для $0 \le i \le 15$. Вихід рамки $S[w_i]$ видає оновлені значення станів очевидним образом.

Шар перестановки (player)

Побітова перестановка, використовувана в PRESENT задається наступною таблицею (біт і стану зміщається на позицію P(i).

i	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$P(i)$	0	16	32	48	1	17	33	49	2	18	34	50	3	19	35	51
i	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
$P(i)$	4	20	36	52	5	21	37	53	6	22	38	54	7	23	39	55
i	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
$P(i)$	8	24	40	56	9	25	41	57	10	26	42	58	11	27	43	59
i	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
$P(i)$	12	28	44	60	13	29	45	61	14	30	46	62	15	31	47	63

Рисунок 4.6 – Побітова перестановка, використовувана в PRESENT

Перетворення ключа (The key schedule)

PRESENT може використовувати 80- і 128-бітні ключі, однак зосередимося на 80-бітній версії. Надаваний користувачем ключ зберігається в регістрі ключів K, представлений у вигляді $k_{79}k_{78}\ldots k_0$. На i-ом раунді 64-бітний раундовий ключ $K_i = k_{63}k_{62}\ldots k_0$, що полягає з 64 лівих битов поточного вмісту регістру K. Таким чином, на i-ом раунді маємо:

$K_i = k_{63}k_{62}\ldots k_0 = k_{79}k_{78}\ldots k_{16}$.

Після розпакування раундового ключа K_i регістр ключа $K = k_{79}k_{78}\ldots k_0$ обновляється в такий спосіб:

1. $[k_{79}k_{78}\ldots k_1k_0] = [k_{18}k_{17}\ldots k_{20}k_{19}]$
2. $[k_{79}k_{78}k_{77}k_{76}] = S[k_{79}k_{78}k_{77}k_{76}]$
3. $[k_{19}k_{18}k_{17}k_{16}k_{15}] = [k_{19}k_{18}k_{17}k_{16}k_{15}] \oplus$ round_counter

Кафедра_КбПЗ_Аспірант

Отже, регістр ключа зрушується на 61 позицію вліво, 4 крайніх лівих біта, що пройшли через S-блок і round_counter значення і складається за модулем 2 з бітами $k_{19}k_{18}k_{17}k_{16}k_{15}$ з K з найменшим значущим бітом з round_counter праворуч.

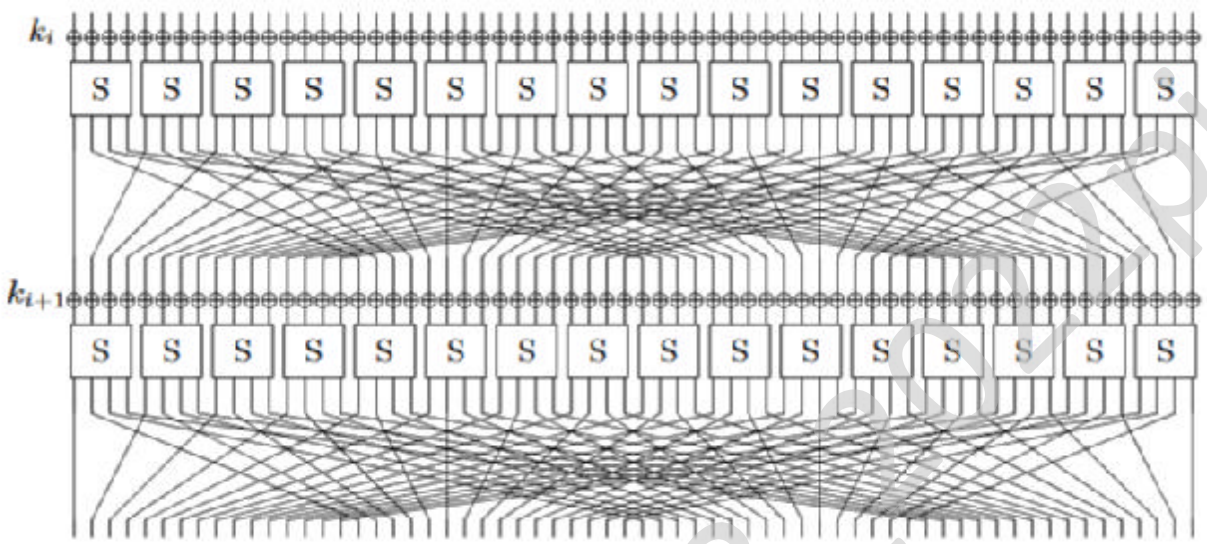

Рисунок 4.7 – Перетворення ключа (The key schedule)

Перетворення ключа для 128-бітного алгоритму можна знайти в додатку до оригінальної статті.

Конструктивні особливості PRESENT

Крім безпеки й ефективної реалізації, основне досягнення PRESENT – його простота. по цьому не дивно, що схожі проекти були прийняті в інших обставинах, і навіть були використані як навчальний посібник для студентів. У даній секції ми обґрунтуємо рішення, прийняті нами при проектуванні PRESENT. Однак, у першу чергу, опишемо очікувані прикладні вимоги.

Мети й середовище застосування

При проектуванні блокового шифру, застосовного в жорстко обмежених оточеннях, важливо зрозуміти, що ми не створюємо блоковий шифр, який, неодмінно, буде застосовний у багатьох ситуаціях – для цього існує AES.

Кафедра_КБПЗ_ЛНТУ_Ярик

Навпаки, ми націлені на досить специфічне застосування, для якого AES не підходить. Вищесказане визначає нам наступні характеристики:

– Шифр буде реалізований «у залізі»

– Додатки будуть вимагатися лише для регулювання рівня безпеки. Отже, 80-бітний ключ буде здоровим розв'язком. Відзначимо, що такої ж позиції дотримуються розроблювачі потокових шифрів проекту eSTREAM.

– Додатки не припускають шифровки великої кількості даних. Таким чином, реалізація може бути оптимізована для продуктивності або простору без внесення занадто великих змін.

– У деяких застосуваннях можлива ситуація, що ключ буде зафіксований при виробництві. У такому випадку, не треба буде змінювати ключ пристрою (що може вилитися в атаки з маніпуляцією ключем .

– Фізичний обсяг пристрою буде першим пріоритетом, після безпеки, що спричинить обмеження на пікові й середні споживання енергії, і, отже, зрушить швидкодія в область низькопріоритетних параметрів.

– У пристроях, що вимагають найбільш ефективного використання фізичного простору, блоковий шифр найчастіше зможе лише шифрувати дані (encryption-only mode). Таким чином, він зможе бути використаний у запит-відповідь (challenge-response) протоколах авторизації, і, при дотриманні контролю стану, може бути використаний для шифровки й дешифрування переговорів із пристроєм, використовуючи режим лічильника.

Виходячи з таких міркувань, розв'язали створити PRESENT як 64-бітний блоковий шифр із 80-бітним ключем. Шифровка й дешифрування, у цьому випадку, мають приблизно схожі фізичні вимоги. Маючи можливість підтримувати як шифрацію, так і дешифрацію, PRESENT буде компактніше, чим підтримуючий лише шифрацію AES. А у випадку encryption-only виконання, наш шифр виявиться й зовсім понад-легко. Суб-ключі що шифрують будуть обчислюватися на ходу.

Кафедра_КОПЗ_2022рік

У літературі є безліч прикладів атак компромісу між часом, датою й пам'яттю, або атак з використанням парадокса днів народження при шифровці великих обсягів даних. Однак, дані атаки залежать тільки від параметрів шифру й не використовують внутрішню структуру. Наша мета полягає в тому, щоб ці атаки були кращим, що можуть застосувати проти нас. Атаки стороннього каналу й атаки з безпосереднім зломом чипа загрожують PRESENT тією самою мірою, як і іншим криптографічним примітивам. Однак для ймовірних застосувань, помірні вимоги безпеки роблять вигоду, одержувану зловмисником на практиці, досить обмеженої. В оцінці ризиків, подібні погрози не сприймаються як істотний фактор.

Перестановочний шар

При виборі шару змішування ключа, наша увага до апаратної ефективності вимагає наявності лінійного шару, який може бути реалізований з мінімальною кількістю керуючих елементів (наприклад, транзисторів. це приводить до побітової перестановки. Приділяючи увагу простоті, ми вибрали регулярну бітову перестановку, що допомагає провести прозорий аналіз безпеки.

Кафедра_КБПЗ_2022рік

5 МЕТОДИКА ВПРОВАДЖЕННЯ СИСТЕМИ В ПРОМИСЛОВУ ЕКСПЛУАТАЦІЮ

Програма встановлюється у тахограф, й має простий та інтуїтивно зрозумілий інтерфейс, який зображений на рисунку 5.1.

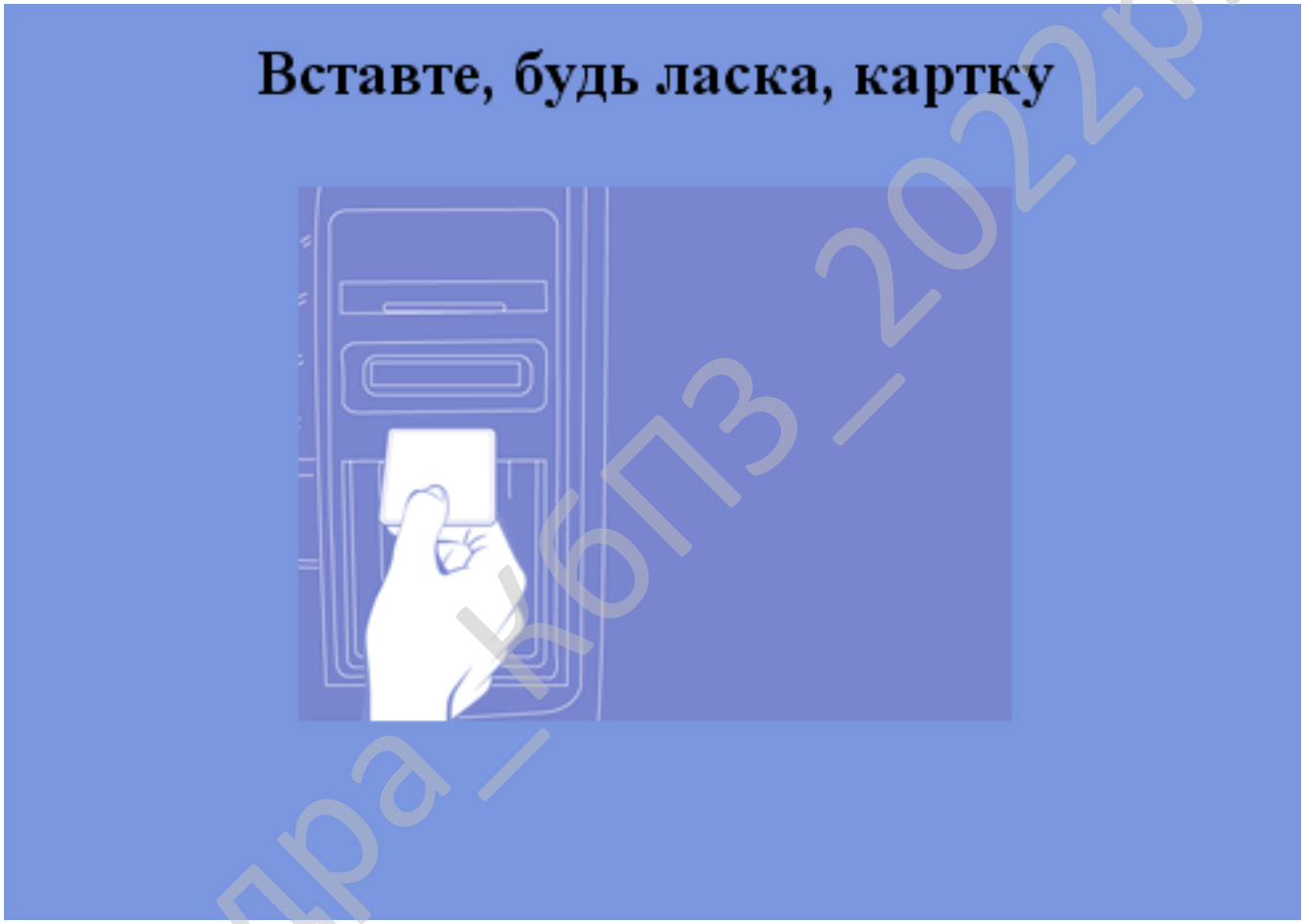

Рисунок 5.1 – Стан очікування карти

На рисунку 5.2-5.8 зображено скріншоти наступних вікон:

– Вікно введення PIN-коду.

– Заставка під час дешифрування даних алгоритмом ДСТ 28147:2009.

– Головне меню.

Кафедра_КБПЗ_2022рік

– Видача готівки (вибір суми).

– Видача готівки (введення суми).

– Видача готівки (запит друку чеку).

– Видача готівки (заставка під час видачі грошей).

– Видача готівки (заставка під час видачі чеку).

– Видача готівки (заставка під час видачі картки).

– Довідка.

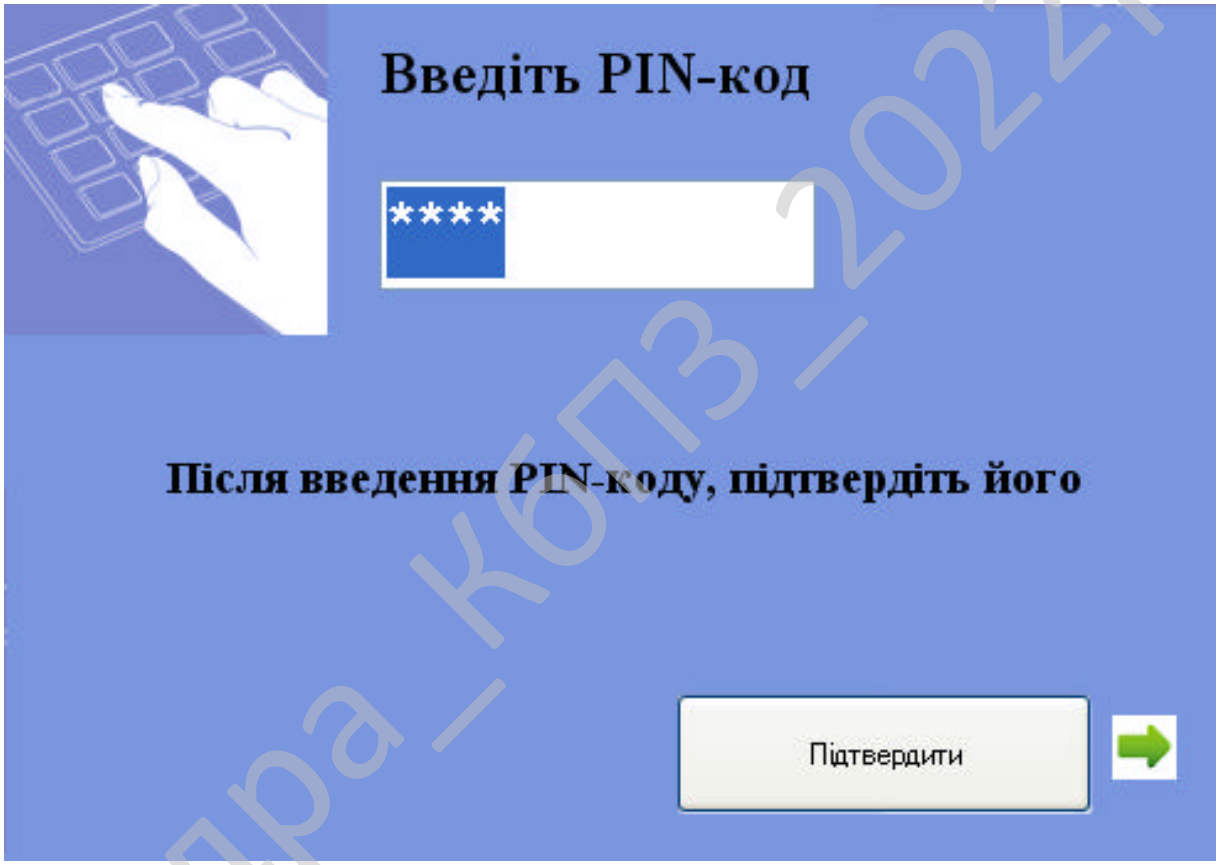

Рисунок 5.2 – Введення PIN-коду

Кафедра_КбПЗ_2022рік

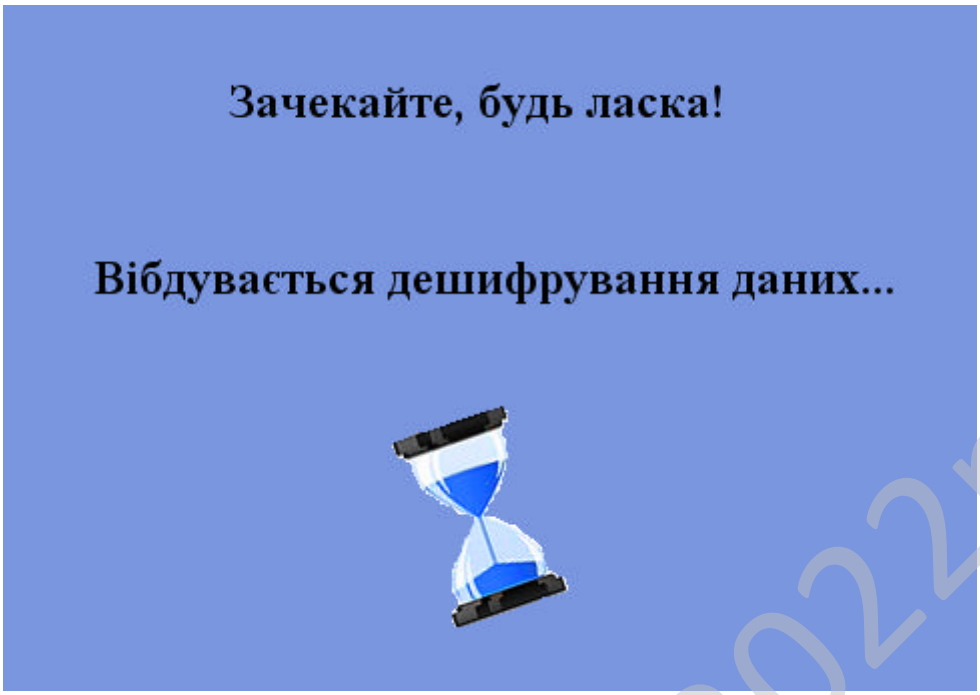

Рисунок 5.3 – Заставка під час дешифрування даних алгоритмом ДСТ 28147:2009

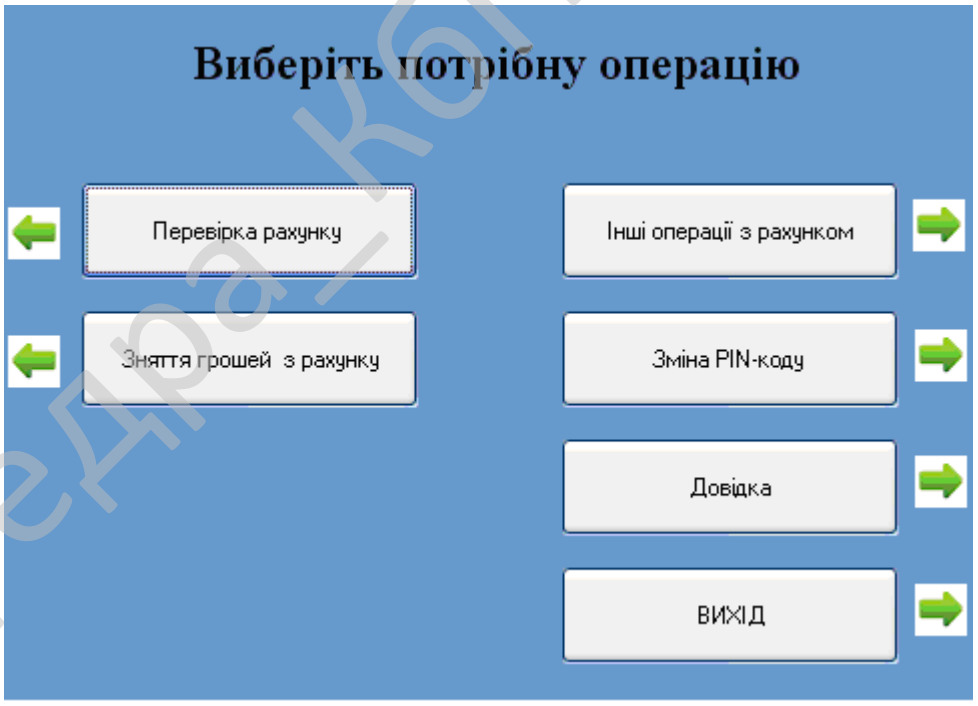

Рисунок 5.4 – Головне меню

Кафедра_КбПЗ_2022рік

Рисунок 5.5 – Видача готівки (вибір суми)

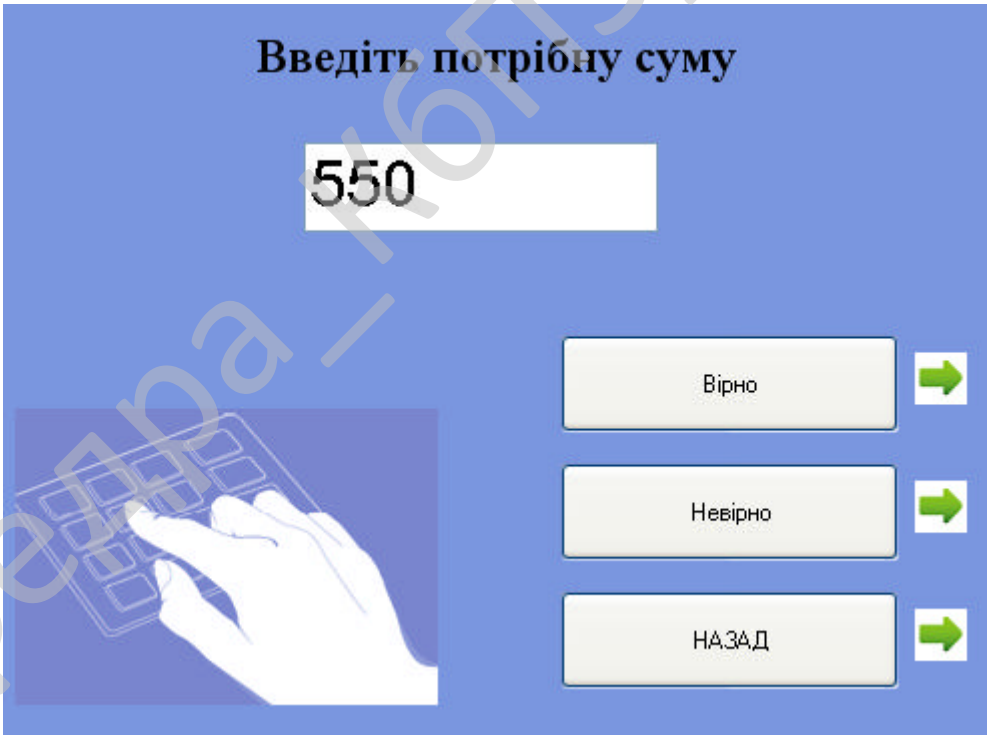

Рисунок 5.6 – Видача готівки (введення суми)

Кафедра_КбПЗ_2022рік

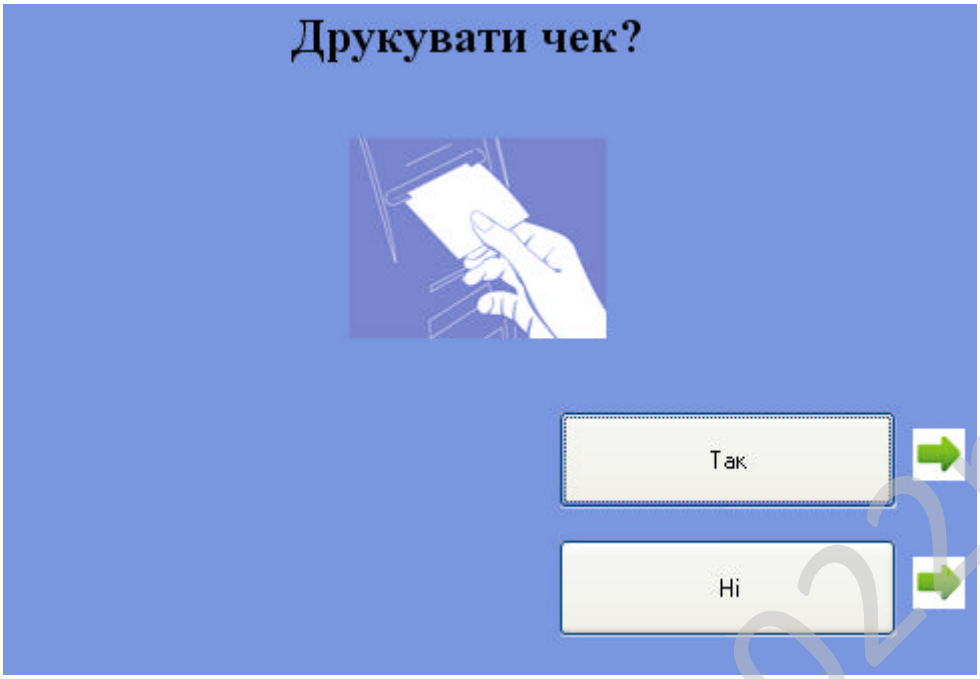

Рисунок 5.7 – Видача готівки (запит друку чеку)

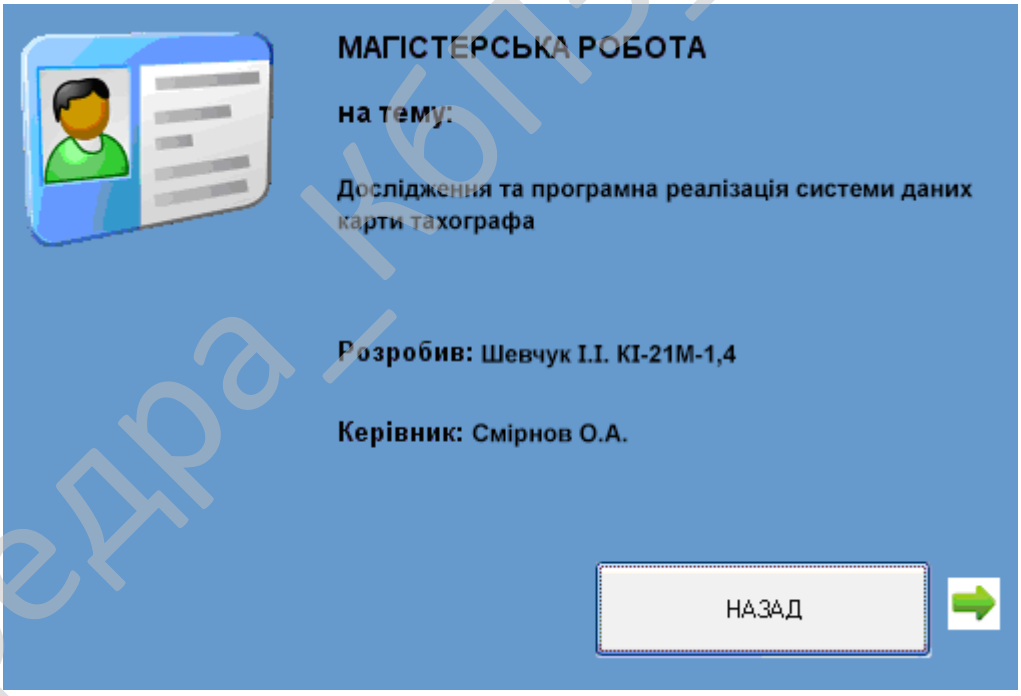

Рисунок 5.8 – Довідка

Кафедра_КБПЗ_2022рік

6 НАУКОВА НОВИЗНА

У випускній кваліфікаційній роботі за другим (магістерським) рівнем вищої освіти розроблено програмне забезпечення, яке призначено для системи даних карти тахографа.

Метою розробки є дослідження та програмна реалізація системи даних карти тахографа.

Об'єктом дослідження є процес даних карти тахографа.

Предметом дослідження є методи даних карти тахографа.

Методи дослідження базуються на методах захисту інформації, методах математичної статистики, методах розробки програмного забезпечення.

Наукова новизна отриманих результатів. У процесі рішення завдань, обумовлених цілями дослідження, отримані наступні результати:

– Удосконалено метод даних карти тахографа.

– Розроблено вітчизняний продукт даних карти тахографа, який має більш широкі можливості, на відміну від існуючих аналогів.

Кафедра_КОПЗ_2022рік

7 ДАНІ ПРО ЕКОНОМІЧНУ ЕФЕКТИВНІСТЬ РОЗРОБЛЕНОЇ ПРОГРАМИ

7.1 Техніко-економічне обґрунтування теми випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти

Після ознайомлення з підприємством та засобами розробки програмної продукції був розроблений план розробки програми. Був підрахований необхідний час для розробки та впровадження програми. Цей час склав 60 днів (три місяці).

В магістерській роботі було проведено дослідження та виконана програмна реалізація системи даних карти тахографа.

Розроблене програмне забезпечення має достатню надійність і задовольняє усім поставленим умовам, а саме:

а) невеликі системні потреби;

б) незалежність від встановлених на комп'ютері баз даних;

в) зручність у користуванні;

г) надійність.

Таблиця 7.1 – Початкові дані

Показники	Позначення	Характеристика або величина
1	2	3
1. Кількість розроблених програм період, шт.	N	1
2. Кількість екземплярів програм, шт.	Ne	19
3. Запланований термін розробки, днів	Fpq	60 (3 місяці)
4. Група задачі підсистеми управління (1-6)	–	1
5. Ступінь новизни задачі (А, Б, В, Г)	–	Б
6. Складність алгоритму (1, 2, 3)	–	2

Кафедра_КБПЗ_2022рік

Продовження таблиці 7.1

1	2	3
7. Кількість макетів вхідної інформації	–	3
8. Кількість форм вихідної інформації.	–	4
9. Мова програмування (1-6)	–	2
10. Попередній досвід (1-6)	–	3
11. Гнучкість проекту ПП (1-6)	–	3
12. Детальність проекту ПП (1-6)	–	2
13. Рівень спрацьованості колективу (1-6)	–	2
14. Ступінь вимірності процесів (1-6)	–	3
15. Необхідна надійність програмного забезпечення (1-6)	–	2
16. Розмір бази даних (порівняно з розміром програми) (1-6)	–	2
17. Складність кінцевого програмного продукту (1-6)	–	2
18. Необхідний рівень забезпечення повторного використання (1-6)	–	2
19. Документованість відповідно до планованого життєвого циклу (1-6)	–	2
20. Вимоги до швидкодії ПП (1-6)	–	2
21. Обмеження на розміри основного сховища даних (1-6)	–	2
22. Різноманітність використовуваних обчислювальних платформ (1-6)	–	2
23. Професійний рівень аналітиків (1-6)	–	2
24. Професійний рівень програмістів (1-6)	–	2
25. Постійність складу команди розробників (1-6)	–	2
26. Досвід розробки додатків (1-6)	–	2
27. Досвід роботи з обчислювальною платформою (1-6)	–	2

Кафедра КбПЗ 2022рік

Продовження таблиці 7.1

1	2	3
28. Досвід роботи з мовою і інструментами середовища розробки (1-6)	–	2
29. Досвід роботи з програмними інструментами розробки (1-6)	–	3
30. Розробка ПЗ для декількох серверів одночасно (1-6)	–	2
31. Вимоги до дотримання встановленого графіка робіт (1-6)	–	2
32. Вартість ПЗ у розробника (НМА), грн.	–	19000
33. Норматив додаткової зарплати, % :	$Н_д$	10
34. Норматив відрахувань у соціальні фонди, %	Нс	22
35. Норматив загальногосподарських витрат, %	Нг	15
36. Норматив витрат на освоєння нових мов програмування, %	Нп	15
37. Рівень рентабельності програмної продукції, %	Ре	50
38. Ставка податку на додану вартість, %	Ндв	20

7.2 Розрахунок трудомісткості розробки програмної продукції

Значення трудомісткості розробки програмного забезпечення для стадій ТЗ, ЕК, ТП та ВП визначаємо по типовим нормам часу приведеним в додатках МВ. Стадія РП є найбільш тривалою і трудомісткою, що робить значний вплив на інші стадії проекту.

Визначимо трудомісткість розробки ПЗ для стадії РП.

Обчислюємо номінальні трудовитрати, люд-міс.:

$$T_{ном} = A\ \text{Size}^B, \qquad (7.1)$$

де: A – коефіцієнт Боема, A = 2,45;

Кафедра_КбПЗ_2022рік

Size – загальний об'єм відлагодженого програмного коду, тис. рядків;

B – показник ступеня, що визначається співвідношенням:

$$B = 1{,}01 + 0{,}001\sum W_i, \qquad (7.2)$$

де: *Wi* – сумарне значення п'яти показників (МВ, додаток 2), що відображають особливості розробки проекту програмного продукту (ПП) і колективу розробників.

$$B = 1{,}01+0{,}001(2{,}43+3{,}64+3{,}38+3{,}95+2{,}73) = 1{,}027.$$

$$T_{ном} = 2{,}45 \cdot 2{,}7^{1{,}026} = 6{,}78 \text{ люд-міс.}$$

Визначаємо уточнені (з урахуванням приведених в МВ додатку 3 сімнадцяти додаткових коефіцієнтів) трудовитрати, люд-міс.:

$$T_{уточн} = T_{ном}\, ПВ_j, \qquad (7.3)$$

де: $ПВ_j$ – добуток сімнадцяти додаткових коефіцієнтів, приведених в МВ додатку 3.

$$T_{уточн} = 6{,}78 \cdot (0{,}88 \cdot 0{,}93 \cdot 0{,}88 \cdot 0{,}91 \cdot 0{,}95 \cdot 1 \cdot 1 \cdot 0{,}87 \cdot 1{,}22 \cdot 1{,}16 \cdot 1{,}1 \cdot 1{,}1 \cdot$$
$$\cdot 1{,}12 \cdot 1{,}1 \cdot 1 \cdot 1{,}1 \cdot 1{,}1) = 9{,}37 \text{ люд-міс.}$$

Ці коефіцієнти дозволяють диференційовано оцінювати результати роботи програмістів, беручи до уваги швидкодію програми, використання різноманітних обчислювальних платформ і інструментів розробки, взаємодію декількох серверів, вимоги до об'ємів баз даних і ін.

Визначаємо підсумкові трудовитрати по стадії робочий проект, люд-дні:

$$T_{РП} = 0{,}3 C T_{уточн}^{0{,}33+0{,}2(B-1{,}01)} S, \qquad (7.4)$$

де: *C* – визначений емпірично коефіцієнт, запропонований авторами методики, (МВ, додаток 4).

S – коефіцієнт стиснення (або подовження) графіка робіт %, що дозволяє коректувати терміни розробки ПЗ згідно встановленим вимогам. Вибираємо в межах (25…350)%. Приймаємо *S* =80 %

$$T_{РП} = 0{,}3 \cdot 2{,}66 \cdot 9{,}37^{0{,}33+0{,}2(1{,}026-1{,}01)} \cdot 80 = 134 \text{ люд/день.}$$

Для зручності визначення загальної трудомісткості на розробку програмного забезпечення результати розрахунків по стадіям зводимо до таблиці 7.2.

Кафедра_КПЗ_2022рік

Таблиця 7.2 – Визначення трудомісткості розробки програмного забезпечення

Стадії розробки	Трудомісткість за типовими нормами та розрахунками	
	Величина, люд/дні	Підстава
Технічне завдання	9	Д5
Ескізний проект	10	Д6
Технічний проект	9	Д7
Робочий проект	134	Ф 7.1-7.4
Впровадження	13	Д13
Всього	175	–

7.3 Визначення чисельності виконавців і планового фонду зарплати

Чисельність ставок інженерів-програмістів для розробки програмного забезпечення визначається за формулою:

$$Ч = \frac{T_{пз} N}{F_{рq} - H_{ев}}, \qquad (7.5)$$

де: $F_{рq}$ – плановий фонд робочого часу одного спеціаліста, днів;

$T_{пз}$ – трудомісткість розробки програмного забезпечення люд-дні.

$$Ч = \frac{175 \cdot 1}{60 - 5} = 3,2 \text{ ставки.}$$

Чисельність інженерів-електронщиків для проведення технічного обслуговування та ремонту комп'ютерних мереж визначається в залежності від наявності технічних засобів і норм витрат часу на виконання профілактичних робіт на протязі року.

Визначаємо затрати часу на виконання профілактичних робіт по обслуговуванню обладнання за період розробки. Результати розрахунку зводимо до таблиці 7.3.

Кафедра_КПЗ_2022рік

Таблиця 7.3 – Затрати часу на виконання профілактичних робіт по обслуговуванню обладнання за розрахунковий період

Найменування обладнання	Профілактичне обслуговування			
	Кількість хв. на один. обл.	Кількість облад-нання	Затрати часу в хв.	Затрати часу в год.
Системний блок ПК	90	6	540	9
Монітор	60	6	360	6
Клавіатура	30	6	180	3
Маніпулятор «мишка»	30	6	180	3
Принтер матричний	60	0	0	0,0
Принтер лазерний	120	1	120	2
Принтер струминний	60	1	60	1
Сканер	20	1	20	0,33
Концентратор-маршрутизатор	30	1	30	0,5
Кабельні господарства ЛОМ на 1 м.п.	2,5	100	250	4,17
Копіювальний апарат	140	1	140	2,33
Усього за рік:			$З_ч$	31,33

Час на профілактику обладнання в загальному балансі робочого часу інженерів-електронщиків неповинен складати більше 10%.

Виходячи з цього фонд робочого часу інженерів-електронщиків складає:

$$Ф_{др}^{с} = \frac{З_ч \cdot n_{тіс}}{1{,}2}, \tag{7.6}$$

$$Ф_{др}^{с} = \frac{31 \cdot 3}{1{,}2} = 77{,}5 \text{ год.}$$

Визначаємо необхідну кількість ставок штатного персоналу сектора ТО:

$$Ч_{ел} = \frac{Ф_{др}^{с}}{F_{др} \cdot T_{зм}}, \tag{7.7}$$

Кафедра_КОПЗ_2022рік

$$Ч_{ел} = 77,5/(60 \cdot 8) = 0,2 \text{ ставки.}$$

Для забезпечення нормального технічного обслуговування засобів ТО та мереж, необхідно прийняти найбільше ціле значення розрахункової чисельності інженерів-електронщиків.

Чисельність інженерів-системотехніків, адміністраторів мережі, дизайнерів WEB вузлів, системних програмістів (аналітиків), бухгалтерів-економістів визначається за потребою в залежності від функціональних обов'язків. Після визначення чисельності персоналу складається штатний розклад.

Таблиця 7.4 – Розрахунок чисельності штатного персоналу сектору системного та адміністративного обслуговування засобів ОТ та комп'ютерних мереж

Посада	Вид роботи	Час	К-ть штатних одиниць
Адміністратор загальної мережі, аналітик	Адміністрування локальної мережі, поштового та серверу DNS (ОС FreeBSD), маршрутизатора Cisco, доменного контролеру Windows Server 2016, серверу доступу ADSL (ОС Linux), налаштування ADSL, VPN PPPoE, Frame Relay, Wi-Fi	2	0,5
	Налаштування і конфігурування базової станції безпровідного зв'язку (CMTS)	0,5	
	Розробка та впровадження проектів з організації зв'язку між віддаленими об'єктами, ЛОМ	0,5	
	Забезпечення цілодобової роботи зв'язку клієнтів до мережі Інтернет	1	
Всього		4	

Кафедра_КБПЗ_2022рік

Продовження таблиці 7.4

Посада	Вид роботи	Час	К-ть штатних одиниць
Продакт-менеджер	Презентації нової продукції, пошук каналів збуту	1	0,25
	Підтримка постійних клієнтів	0,5	
	Оформлення договорів, ведення тендерів	0,25	
	Контроль взаєморозрахунків з постачальниками	0,25	
Всього		2	
Дизайнер WEB	Розробка концепції оформлення та інтерфейсу сайту, оптимізація дизайну існуючих, проектує їх структуру та навігацію	1	0,25
	Створення графічних і стилістичних елементів сайту	0,5	
	Оформлення банерів і промо-сторінок	0,25	
	Розміщення графіки і контенту на Інтернет сторінках	0,25	
Всього		2	
Інженер верстальник	Розробка та верстка макетів рекламної продукції та технічної документації	1	0,25
	Верстка друкованих видань	0,5	
	Додрукова підготовка макетів	0,25	
	Розміщення графіки і контенту на Інтернет сторінках	0,25	
Всього		2	

Кафедра_КОПЗ_2022рік

Складемо штатний розклад виконавців.

Таблиця 7.5 – Штатний розклад виконавців

Посада	Кількість ставок	Середньомісячний оклад, грн.	Всього за період розробки, грн.
Керівник (ІТ-менеджер)	1	14594	43782
Продакт-менеджер	0,25	14000	10500
Інженер-програміст	3,2	14500	139200
Інженер-електронщик	0,2	14000	8400
Інженер-системотехнік	0,25	14000	10500
Адміністратор мережі	0,5	14000	21000
Системний програміст	0,25	14000	10500
Дизайнер WEB	0,25	14000	10500
Інженер-верстальник	0,25	14000	10500
Бухгалтер-економіст	0,5	14000	21000
Всього за період розробки	$R_{сп} = 6,65$	-	$Ф_{роб} = 285882$

Розрахуємо середньоденну зарплату одного виконавця:

$$З_{сд} = \frac{Ф_{роб}}{R_{сп} F_{рд}}, \quad (7.8)$$

де: $Ф_{роб}$ – загальна сума зарплати за плановий період, грн.

$$З_{сд} = \frac{285882}{6,65 \cdot 60} = 716,5 \; грн.$$

7.4 Розрахунок капітальних вкладень та амортизаційних відрахувань у розробника

Балансова вартість будівель визначається з урахуванням кількості робочих місць виконавців, питомої площі на одне робоче місце, та вартості одного квадратного метра виробничої площі:

Кафедра_КбПЗ_2022рік

$$Б_{уд} = R^1_{сп} S_у Ц_{пл}, \quad (7.9)$$

де: $R^1_{сп}$ – кількість робочих місць виконавців, шт. Приймаємо 8 робочих місць;

$S_у$ – питома площа на одне робоче місце, $м^2$;

$Ц_{пл}$ – вартість одного квадратного метра площі, грн.

Згідно даних інтернет ресурсу DOM.RIA (https://dom.ria.com) ціна одного квадратного метра площі, вік якої не перевищує 30 років, по місту складає 500…1600 $у.о./м^2$. Враховуючи, що курс складає 1 у.о. = 38 грн. приймаємо для розрахунку вартість одного метра квадратного рівною 20000 грн./м^2. На кожне робоче місце у середньому потрібно 8 $м^2$. З урахуванням цього:

$$Б_{уд} = 8 \cdot 8 \cdot 20000 = 1280000 \text{ грн.}$$

Вартість передавальних пристроїв складає 10% від вартості будівель, і у даному випадку вона складе: 128000 грн.

Балансова вартість інвентарю розраховується за нормою 3500 грн. на одне робоче місце. Тобто:

$$І_{нв} = R^1_{сп} \cdot Ц_м, \quad (7.10)$$

де: $Ц_м$ – ціна меблів для одного робочого місця, грн.

$$І_{нв} = 8 \cdot 3500 = 28000 \text{ грн.}$$

Балансова вартість обчислювальної техніки визначається по оптовим цінам постачальника з врахуванням витрат на транспортування.

Специфікація на обчислювальну техніку наведена в таблиці 7.7.

Дані по оптовій ціні на обладнання та комплектуючі вибирались по прайсу фірми Комп'ютерторг за 15.10.22 – джерело

http://computorg.ua/ru/price.html

Кафедра_КОПЗ_2022рік

Таблиця 7.6 – Специфікація

Найменування комплектуючої або обладнання	Тип	Оптова ціна
Персональний комп’ютер		10947
Системний блок		7347
Процесор	INTEL CELERON G5905 (BX80701G5905 box	-
Системна плата	ASROCK H470M-HDV, підтримка 10-г покоління Intel Core i9/Core i7/Core i5/Cor i3/Pentium/Celeron, 1 x PCI-E 3.0 x1, 1 x PCI E 3.0 x16, 1 x VGA, 2 x USB 3.2 Gen 1, 1 RJ45, 4 x USB 2.0, 1 x PS/2, 1 x DVI, 1 HDMI, 3 x Audio, Ami, UEFI	-
Відеокарта	VC VTX Radeon HD6570 1GB GDDR3 128bit, 650 MHz/1334 MHz, PCI-E 2.1, DVI HDMI, VGA (VX6570 1GBK3-H)	-
Жорсткий диск	SSD: 480 Gb	-
Оперативна пам’ять	DDR4 8GB 2666 MHZ GOODRAM (GR2666D464L19S/8G)	-
DVD-привод	DVDRW Pioneer DVR-TD10RS SATA Slim Black Bulk (DVR-TD10RS)	-
Корпус	ATX Middle Tower FOXCONN Pro, 3GTLA 489, PSU 350W(FSP Brand: ATX-350PNR 12cm), black, (front bezel – black+light silver body material – 0.6mm), 80mm fan (rear) 2xUSB2.0/AUDIO/MIC, Air Duct, Tool-les chassis design,Thermally Advantaged Chassis	-

Кафедра_КПЗ_2022рік

Продовження таблиці 7.6

Найменування комплектуючої або обладнання	Тип	Оптова ціна
Кулер	–	–
Кардрідер внутрішній	USB 2.0 Card reader STORM CR-35U1A4-E int. 3.5", 1*USB2.0+AUDIO+1394, multi: A Type Cards, black	220
інше	Клавіатура, мишка	Подару-нок
Монітор	22" TFT, ASUS VW223D (5ms, 300/3000: 1 170/160, D-SUB, Wide)	3600
Принтер лазерний	Canon i-SENSYS LBP6030W	2700
Принтер струминний	Epson Stylus Photo P50 (C11CA45341) + USE cable	5500
Копіювальний апарат	Canon i-SENSYS MF217W with Wi-Fi	5965

Таблиця 7.7 – Балансова вартість обчислювальної техніки

Найменування обчислювальної техніки	Кількість, шт.	Ціна за одиницю, грн.	Витрати на транспортування, монтаж та випробовування.	Загальна вартість, грн.
Персональні комп’ютери	8	10947	8757,6	96333,6
Принтер лаз.	2	2700	540	5940
Принтер струм.	1	5500	550	6050
Сканери	-	-	-	0
Копіюв. апарат	1	5965	596,5	6561,5
Всього	–	–	–	114885,1

Кафедра_КбПЗ_2022рік

Витрати на транспорт, монтаж та випробування можуть бути прийняті в межах до 10% від оптової ціни.

Для визначення необхідної кількості капітальних вкладень складемо таблицю 7.8.

Таблиця 7.8 – Вартість основних фондів та амортизаційні відрахування розробника

Групи та види основних фондів	Балансова вартість, грн.	Амортизація	
		Норма, %	Відрахування, грн.
1	2	3	4
Група 3			
1. Будівлі	1280000	-	-
2. Передавальні пристрої	128000	-	-
Всього по групі	1408000	5	70400
Група 4			
3. Обчислювальна техніка	114885	-	-
Всього по групі	114885	50	57442,5
Група 5, 6			
4. Вимірювальні пристрої	5190	25	1297,5
5. Транспортні засоби	97500	20	19500
6. Господарський інвентар	28000	25	7000
Всього по групі	130690	-	27797,5
7. Нематеріальні активи	19000	10	1900
Разом	$К_р$ = 1672575		$А_р$ = 157540

Кафедра_КБТЗ_2022рік

Примітка: вартість автомобіля Daewoo Lanos SX 2007 взята по даним автобазару Авто-РІА, джерело https://auto.ria.com/uk/auto_daewoo_lanos_33568345.html, складає 97500 грн.

7.5 Визначення собівартості розробки та ціни програмної продукції

Визначимо основну зарплату виконавців:

$$З_о = \frac{З_{сд} \cdot Т_{пз}}{N_е}, \quad (7.11)$$

де: $N_е$ – кількість екземплярів програм, шт.

$$З_о = 716{,}5 \cdot 175/19 = 6600 \text{ грн.}$$

Визначимо додаткову зарплату (оплата відпусток, виконання державних та суспільних обов'язків) на рівні 10%:

$$З_д = З_о \cdot Н_q \cdot 0{,}01, \quad (7.12)$$

де: $Н_q$ – норматив додаткової зарплати, %.

$$З_д = 6600 \cdot 10 \cdot 0{,}01 = 660 \text{ грн.}$$

Відрахування на соціальні потреби за нормативом $Н_с = 22\%$ від суми основної та додаткової зарплати:

$$С_{оц} = 0{,}01 \cdot Н_с (З_о + З_о), \quad (7.13)$$

де: $Н_с$ – відрахування на соціальні потреби, %.

$$С_{оц} = 0{,}01 \cdot 22(6600+660) = 2686 \text{ грн.}$$

Визначимо загальногосподарські витрати (електроенергію, ремонт і утримання приміщень і т.д) за нормативом $Н_г = 15\%$ від основної зарплати:

$$Г_{осп} = З_о \cdot Н_г \cdot 0{,}01, \quad (7.14)$$

де: $Н_г$ – загальногосподарські витрати, %.

$$Г_{осп} = 6600 \cdot 15 \cdot 0{,}01 = 990 \text{ грн.}$$

Визначимо витрати на матеріали для розробки програмної продукції за нормами споживання та діючими цінами за одиницю виміру:

Кафедра_КоПЗ_2022рік

$$З_М = (З_{М1} + З_{М2} + З_{М3})/N_е, \quad (7.15)$$

де: $З_{М1}$ – вартість паперу, грн.;

$З_{М2}$ – вартість запам'ятовуючих пристроїв, грн.;

$З_{М3}$ – вартість фарби, картріджей, тонеру, грн.;

$N_е$ – кількість екземплярів програм, шт.

Згідно прийнятих норм на підприємстві $n_{вит}$ приймаємо 0,5 пачки паперу на період розробки. Тоді, враховуючи, що вартість пачки паперу складає $Ц_п$=210 грн., визначаємо вартість паперу за період розробки:

$$З_{М1} = Ц_п \cdot N_м. \quad (7.16)$$

$$З_{М1} = 210 \cdot 0,5 = 105 \text{ грн.}$$

Згідно прийнятих норм по комплектації до вартості запам'ятовуючих пристроїв входить вартість CD/DVD дисків. Їх кількість дорівнює кількості коробочних версій запропонованого продукту (приймаємо 5):

$$З_{М2} = \sum Ц_д , \quad (7.17)$$

де: $Ц_д$ – вартість дисків CD/DVD: CDR box – 21,4 грн./шт., DVD-R box – 35 грн./шт.

$$З_{М2} = 5 \cdot 21,4 = 107 \text{ грн.}$$

Згідно норм одноразовій заправці підлягають усі друкуючі пристрої і становить:

$$З_{М3} = \sum Ц_з., \quad (7.18)$$

де: $Ц_з$ – вартість розхідних матеріалів друкуючих пристроїв: відновлення та заправка картриджу для Canon i-SENSYS LBP6030W – 574 грн.; картридж для Epson Stylus Photo P50 – 558 грн.; відновлення картриджу для MF217W – 570 грн.

$$З_{М3} = 574+558+570 = 1702 \text{ грн.}$$

$$З_М = (105 + 107 + 1702)/19 = 101 \text{ грн.}$$

Визначимо витрати на освоєння нових мов програмування або операційних систем за нормативом ($Н_п$ = 15%) від основної зарплати виконавців:

$$О_п = З_о \cdot Н_п \cdot 0,01, \quad (7.19)$$

де: $Н_п$ – норматив витрат на освоєння нових мов програмування, %.

Кафедра_КбПЗ_2022рік

$$O_п = 6600 \cdot 15 \cdot 0{,}01 = 990 \text{ грн.}$$

Величини ціна підприємства, податок на додану вартість, відпускна ціна програмної продукції визначаються за формулами, приведеними в таблиці 7.9

Таблиця 7.9 – Нормативна калькуляція собівартості розробки програмного забезпечення задачі

Найменування статей витрат	Позначення	Величина, грн
1. Основна зарплата виконавців	$З_о$	6600
2. Додаткова зарплата виконавців	$З_д$	660
3. Відрахування на соціальні потреби	$С_{оц}$	2686
4. Загальногосподарські витрати	$Г_{осп}$	990
5. Витрати на матеріали	$З_М$	101
6. Освоєння нових операційних систем, мов програмування	$О_п$	990
7. Амортизація основних фондів	$А_м$	2073
8. Повна собівартість програмного забезпечення	$С_п$	14100
9. Плановий прибуток	$П_р$	7050
10.Ціна підприємства $Ц_п = С_п + П_р$	$Ц_п$	21150
11.Податок на додану вартість $ПДВ = 0.01 \cdot Н_{дв} \cdot Ц_п$	$ПДВ$	4230
12.Відпускна ціна програмної продукції $Ц = Ц_п + ПДВ$	$Ц$	25380

Визначимо витрати на амортизацію основних фондів з урахуванням загальної річної суми амортизаційних відрахувань та кількості екземплярів програм ($N_е$ = 19 прим.):

$$А_м = \frac{А_р \cdot N_{міс}}{N_е \cdot 12}, \qquad (7.20)$$

де: $А_р$ – загальна річна сума амортизаційних відрахувань, грн.

Кафедра КбПЗ 2022рік

$$A_м = 157540 \cdot 3/(19 \cdot 12) = 2073 \text{ грн.}$$

Повна собівартість ПЗ визначається як сума витрат за попередніми статтями калькуляції:

$$C_п = З_о + З_д + C_{оц} + Г_{осп} + З_м + О_п + А_м. \quad (7.21)$$

$$C_п = 6600+660+2686+990+101+990+2073 = 14100 \text{ грн.}$$

Визначимо плановий прибуток за рівнем рентабельності ($P_п$) програмної продукції, яка залежить від складності програми та ступеня новизни задачі.

Для даного програмного забезпечення рівень рентабельності складає 50%.

$$П_р = 0{,}01 \cdot P_п \cdot C_п, \quad (7.22)$$

де: $P_п$ – рівень рентабельності, %.

$$П_р = 0{,}01 \cdot 50 \cdot 14100 = 7050 \text{ грн.}$$

7.6 Визначення об'єму капітальних вкладень у споживача програмної продукції

Об'єм капітальних вкладень у споживача програмної продукції визначаємо на основі балансової вартості основних фондів, яка враховує ціну, транспортно-заготівельні витрати, вартість будівель, монтажних та пусконалагоджувальних робіт, а також витрати на випробування у виробничих умовах. Результати розрахунків зводимо у таблицю 7.10.

Таблиця 7.10 – Розрахунок об'єму капітальних вкладень у споживача програмної продукції

Найменування капітальних вкладень	Сума за варіантами, грн.	
	Базовий	Новий
Вартість програмної продукції	–	25380
Всього капітальних витрат	–	25380

Кафедра_КБПЗ_2022рік

7.7 Визначення експлуатаційних витрат

Експлуатаційні витрати у споживача програмної продукції визначаємо при умові роботи підсистеми на протязі року. Результати зводимо до таблиці 7.11.

Таблиця 7.11 – Розрахунок експлуатаційних витрат у споживача програмної продукції

Найменування статей витрат	Позначення	Сума витрат за варіантами, грн.	
		Базовий	Новий
1. Витрати на технічне обслуговування	$З_р$	107360	21472
2. Витрати на електроенергію	$З_{ел}$	20580	17640
3. Витрати на амортизацію	$З_{ам}$	0	6345
Всього витрат за рік	$І$	127940	45457

Витрати на профілактичні роботи:

$$З_р = Т_р \cdot З_г \cdot (1 + 0{,}01 \cdot Н_q) \cdot (1 + 0{,}01 \cdot Н_c), \qquad (7.23)$$

де: $Т_р$ – кількість годин обслуговування кожного комп'ютера за рік, *год.*;

$З_г$ – заробітна плата обслуговуючого персоналу, *грн / год*.

Після купівлі нового програмного забезпечення кількість профілактичних годин робіт зменшилася з 1000 годин на рік до 200 годин на рік, тому витрати на технічне обслуговування зменшилися з:

$$З_{р\,баз} = 1000 \cdot 80 \cdot 1{,}1 \cdot 1{,}22 = 107360 \text{ грн},$$

до:

$$З_{р\,нов} = 200 \cdot 80 \cdot 1{,}1 \cdot 1{,}22 = 21472 \text{ грн}.$$

Витрати на електроенергію визначаються з урахуванням споживаємої потужності ($П_{ел}$) в кіловатах, часу експлуатації технічних засобів ($Т_р$) в годинах та ціни однієї кіловат-години ($Ц_{ел}$):

Кафедра_КбПЗ_2022рік

$$З_{ел} = П_{ел} \cdot Т_р \cdot Ц_{ел}. \quad (7.24)$$

$$З_{ел\ баз} = 7 \cdot 0{,}35 \cdot 4000 \cdot 2{,}1 = 20580 \text{ грн.}$$

$$З_{ел\ нов} = 7 \cdot 0{,}3 \cdot 4000 \cdot 2{,}1 = 17640 \text{ грн.}$$

Витрати по амортизації визначаються на основі норм амортизаційних відрахувань, вартості програмної продукції і основних фондів. Для розрахунку складаємо таблицю 7.12.

Таблиця 7.12 – Розрахунок амортизаційних відрахувань

Групи основних фондів	Норма амортизації %	Балансова вартість, грн., за варіантами		Сума відрахувань, грн за варіантами	
		Базовий	Новий	Базовий	Новий
Програмна продукція	25	–	25380	–	6345
Всього відрахувань	-	–	25380	–	6345

7.8 Визначення економічної ефективності програмної продукції

Економічна ефективність програмного забезпечення визначається для виготовлювача і споживача за такими показниками.

Величина економічного ефекту при виготовленні програмної продукції, розраховуємо за формулою:

$$E_в = (Ц_п - C_п) \cdot N_е - \sum_{i=1}^{m} E_{p_m} \cdot K_{p_m}, \quad (7.25)$$

де: $K_р$ – балансова вартість основних фондів розробника, грн.; $E_р$ – розрахунковий коефіцієнт капіталовкладень.

$$Ев = (21150 - 14100) \cdot 19 - (0{,}05 \cdot 1408000 + 0{,}5 \cdot 114885 +$$
$$+0{,}25 \cdot 33190 + 0{,}2 \cdot 97500 + 0{,}1 \cdot 19000) \cdot 3/12 = 94565 \text{ грн.}$$

Визначимо період окупності додаткових капітальних вкладень у виробника програмної продукції:

$$T_в = \frac{K_p^*}{(Ц_п - C_п) \cdot N_е}, \quad (7.26)$$

Кафедра_КоПЗ_2022рік

де: K_p^* – балансова вартість основних фондів розробника без врахування вартості ОФ третьої групи, так як їх строк служби на порядок більший ніж період розробки ПЗ.

$$T_в = \frac{264575}{(21150 - 14100) \cdot 19 \cdot 12 / 3} = 0{,}5 \text{ років}.$$

Показники економічної ефективності програмної продукції зводимо до таблиці 7.13.

Таблиця 7.13 – Показники економічної ефективності програмної продукції

Найменування показників	Одиниця виміру	Величина
1. Кількість екземплярів програми	Прим.	19
2. Повна собівартість розробленої програми	Грн.	14100
3. Ціна розробленої програми	Грн.	21150
4. Плановий прибуток від реалізації розробленої програми	Грн.	7050
5. Рентабельність програмної продукції	%	50
6. Об’єм додаткових капітальних вкладень у виробника програмної продукції	Грн.	1672575
7. Загальний прибуток від реалізації програмної продукції	Грн.	133950
8. Величина економічного ефекту при виготовлені програмної продукції	Грн.	94565
9. Період окупності додаткових капітальних вкладень у виробника програмної продукції	Років.	0,5
10.Об’єм додаткових капітальних вкладень у споживача програмної продукції	Грн.	25380
11.Величина економічного ефекту у користувача програмної продукції	Грн.	76138
12.Період окупності додаткових капітальних вкладень у користувача програмної продукції	Років	0,3

Кафедра_КОПЗ_2022рік

Визначимо величину економічного ефекту у користувача програмної продукції за формулою:

$$E_{сп} = (I_б - I_н) - E_н(K_н - K_б), \qquad (7.27)$$

де: $I_б$, $I_н$ – величина експлуатаційних витрат за базовим и новим варіантом відповідно;

$K_б$, $K_н$ – об'єм капітальних вкладень за варіантами, що порівнюються.

$$E_{сп} = (127940\text{-}45457)\text{-}0{,}25 \cdot 25380 = 76138 \text{ грн.}$$

Визначимо період окупності додаткових капітальних вкладень у споживача програмної продукції за рахунок зниження експлуатаційних витрат:

$$T_{сп} = \frac{K_н - K_б}{I_б - I_н}, \qquad (7.28)$$

$$T_{сп} = \frac{25380}{127940 - 45457} = 0{,}3 \ року.$$

7.9 Висновки

Розроблена програма економічно вигідна. За рахунок впровадження програмного забезпечення досягається скорочення часу обробки інформації, підвищується культура праці, підвищення якості приймаючих управлінських рішень.

Кафедра_КбПЗ_2022рік

8 ЗАХОДИ З ОХОРОНИ ПРАЦІ ТА ТЕХНІКИ БЕЗПЕКИ

8.1 Вступ

Охорона здоров'я працівників, забезпечення безпеки умов праці, ліквідація професійних захворювань і виробничого травматизму повинна складати одну з головних завдань роботодавця.

Основою охорони праці є науковий аналіз умов праці, технологічних процесів, виробничого обладнання, робочих місць, трудових операцій, організації виробництва з метою виявлення шкідливих і небезпечних виробничих факторів, їх властивостей, особливостей виливу на організм людини. На підставі такого аналізу розробляються заходи та засоби, спрямовані на мінімізацію несприятливого впливу виробничих факторів, створення безпечних та нешкідливих умов праці.

Для того, щоб об’єктивно проаналізувати відповідність умов праці діючим нормативно-правовим актам, необхідно здійснити санітарно-гігієнічну характеристику умов праці відділу, в якому працює програміст, над розробкою даного програмного продукту.

В зв’язку з цим необхідно сконцентрувати увагу на небезпечних і шкідливих чинниках пов’язаних з постійною роботою за комп’ютером.

Електробезпека є одним із критичних питань для співробітників, що працюють із технікою, яка одержує живлення з електричної мережі. При невиконанні норм електробезпеки можлива поразка електричним струмом.

Кафедра_КОПЗ_2022рік

8.2 Аналіз санітарно-гігієнічних умов праці на робочому місці програміста

Розглянемо умови праці у приміщенні, в якому працюють програмісти. Геометричні розміри приміщення наведено у таблиці 8.1.

Таблиця 8.1 – Розміри приміщення

Найменування	Значення, м
Ширина	5
Довжина	6,4
Висота	3,2

Таблиця 8.2 – Площа та обсяг приміщення, на одного працюючого·

Геометрична характеристика	Одиниця виміру	Нормативне значення *	Фактичне значення
Площа, S	$м^2$	не менше 6.0	6,4
Обсяг, V	$м^3$	не менше 20.0	20,48

* Згідно ДСанПіН 3.3.2.007-98 (Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин) [2].

У зазначеному приміщенні працюють 5 людей. За даними, які наведено у табл. 8.1, та табл. 8.2, можна зробити висновок, що площа та об’єм приміщення у розрахунку на одно робоче місце програміста не відповідають нормативним вимогам ДСанПіН 3.3.2-007-98 «Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин» [2], але відповідають нормативним вимогам Наказу Міністерства соціальної політики України № 207, від 14.02.2018 «Про затвердження Вимог щодо безпеки та захисту здоров’я працівників під час роботи з екранними пристроями» [5] та НПАОП 0.00-1.28-10 «Правила охорони праці під час експлуатації електронно-

Кафедра_КБПЗ_2022рік

обчислювальних машин»). Таким чином можна зробити висновок, що санітарно-гігієнічні умови праці на робочому місці програміста відповідають вимогам.

Температура повітря в приміщенні визначається впливом температури зовнішнього повітря і тепловою енергією, яка виділяється всередині приміщення. Джерелами виділення теплоти в даному приміщенні є електроустаткування, освітлювальні прилади, а також люди. У світлий час доби джерелом надлишкового тепла є сонячна радіація. Згідно Постанови № 42 від 01.12.1999 Головного державного санітарного лікаря України, робота, виконувана в даному приміщенні, відноситься до категорії Іа. В цьому випадку людина витрачає енергії до 120 ккал у годину. Вологість повітря в приміщенні визначається впливом багатьох факторів, серед яких: вологість атмосферного повітря, виділення вологи людьми (при диханні та випарами з поверхні шкіри).

Мікроклімат повітряного середовища в приміщенні характеризується запиленістю та загазованістю повітря. Мікроклімат приміщення визначається діючим на організм людини поєднанням, вологості, температури, швидкості руху повітря та інтенсивності теплового випромінювання. Аналіз мікроклімату складається з визначення зазначених вище факторів і порівняння результатів із встановленими нормами.

У таблиці 8.3 наведено оптимальні та фактичні значення параметрів мікроклімату як для категорії ваги робіт Іа, так і розглянутого приміщення. У приміщеннях, де встановлено ЕОМ, рекомендується застосування тільки оптимальних значень показників мікроклімату.

Таблиця 8.3 – Оптимальні і фактичні значення параметрів мікроклімату

Пора року	Оптимальні для Іа			Фактичні		
	Температура, °С	Воло-гість,%	Швидкість повітря, м/с	Температура,°С	Воло-гість%	Швидкість повітря, м/с
Холодна	22-24	40-60	0,1	22-23	40-55	0,9
Тепла	23-25	50-70	0,1	24-25	50-65	0,1

Кафедра_КОПЗ_2022рік

Проведений аналіз показує, що показники мікроклімату в приміщенні відповідають установленим нормам. Штучне опалення застосовується у холодний період року.

В літню пору застосовується кондиціонер.

Для боротьби з пилом робляться регулярні провітрювання та вологі прибирання приміщенні.

У приміщенні знаходяться наступні джерела шуму: принтер HP 1100, електродвигуни вентиляторів ЕОМ.

Одним з найважливіших факторів, які впливають на ефективність трудової діяльності людини, та попереджають травматизм і професійні захворювання програмістів є освітлення на робочому місці.

З 2019 року діють Державні будівельні норми України “Природне і штучне освітлення” – ДБН В.2.5-28:2018 [1], у яких прописані вимоги до використання всіх освітлювальних приладів, у т.ч. світлодіодних.

Працю працівника, який постійно працює за комп’ютером, згідно ДБН В.2.5-28:2018 [1], можна віднести до роботи з малою точністю (найменший розмір об'єкта розрізнення від 1 до 5 мм) V-го розряду зорової роботи, з великою контрастністю об'єкта розрізнення (символів на екрані дисплея), з темним тлом (під розряд зорової роботи В). Приміщення можна віднести до 1-ої групи приміщень, у яких проводиться розрізнення об'єктів зорової роботи при фіксованому напрямку лінії зору того, що працює на робочу поверхню. Для такого типу приміщень і розряду зорової роботи нормоване значення коефіцієнта природної освітленості (КПО) робочої поверхні (при поєднаному, спільному освітленні), повинен становити не більше 1,5%, освітленість при штучному висвітленні повинна становити 300 Лк. [1], Крім того все поле зору повинне бути освітлено достатньо рівномірно – ця основна гігієнічна вимога. Так як яскраве світло на ділянці периферійного зору значно збільшує напруженість очей і, як наслідок, призводить до їх швидкої стомлюваності, ступінь освітлення приміщення і яскравість екрану комп'ютера повинні бути приблизно однаковими.

Кафедра_КПЗ_2022рік

8.3 Розробка заходів з умов поліпшення охорони праці

Згідно аналізу умов праці в розглянутому приміщенні, ми одержали наступні результати:

– розмірі приміщення, у розрахунку на одному працюючого, відповідають нормативам;

– мікроклімат відповідає нормативному значенню;

– акустичні умови роботи не перевищують нормативних значень.

Таким чином можна припустити, що основною причиною можливого зниження працездатності програміста є психофізіологічний фактор, тому основна пропозиція буде така: дотримання позитивної психологічної атмосфери в колективі та регламентованого режиму праці та відпочинку, організація робочого місця з урахуванням ергономічних вимог.

Рекомендовані заходи: регулярні періодичні наочні огляди персоналом шляхів для евакуації людей із приміщення, відповідно до плану евакуації (який повинен розташовуватись на видному місці у приміщенні), включення до колективного договору мінімально можливого вмісту аптечок з обов'язковою наявністю масок-клапанів, або іншого спорядження для штучного дихання. Регулярна періодична перевірка параметрів заземлення та занулення (вимірювання опору ланцюга).

Регулярна наочне знайомство персоналу із шляхами для евакуації людей із приміщення відповідно до плану евакуації, забезпечення розподільних щитів спеціальними розетками з заземлюючими контактами; організація заземлення всіх приладів і пристроїв, які працюють при напрузі вище 36 В.

Так як при ураженні електричним струмом у людини може статися фібриляція шлуночків серця, в організації бажано мати дефібрилятор і підготовлений персонал для роботи з ним.

Кафедра_КБПЗ_2022рік

8.4 Розрахункова частина

Для захисного штучного заземлення застосовуються вертикальні електроди: металевий куток 35·35·5 мм., довжиною *L=2 м.,* та горизонтальний електрод – металева полоса з перетином 40·4 мм. Напруга – 220/380 В. Розрахункова схема розташування заземлюючих електродів – у ряд.

Розрахунок проводиться за допустимим опором розтіканню струму заземлювача.

Початкові дані для розрахунку захисного заземлення: тип верхнього шару грунта – чорнозем, нижнього шару грунта – глина (питомий опір $\rho_2 = 40$ *Ом·м)*. Умовна товщина верхнього шару грунта: *H=0,6 м*. Відстань між вертикальними заземлювачами (електродами) *A*=3 *м*. Глибина закладення горизонтального контура заземлення *t=0,8 м*. Опір заземлювача, який нормується: $R_{ЗН} = 4$ *Ом*. Необхідно визначити необхідну кількість вертикальних заземлювачів та длину полоси (горизонтального заземлювача).

Розрахунок.

Відстань від центра вертикального заземлювача до поверхні землі:

$$T=t+L/2=0,8+2/2=1,8 \text{ м.}$$

Розрахунковий питомий опір грунта (з врахуванням того, що фактично вся конструкція заземлювача розташовується у нижньому шарі грунта):

$$\rho = \psi \rho_2 = 1,36 \cdot 40 = 54,5 \text{ Ом·м.}$$

де $\psi = 1,36$ – табличне значення коефіцієнта сезонності для відповідної кліматичної зони у багатошаровому грунті [6]; $\rho = 40$ *Ом·м.* – табличне значення питомого опору нижнього шару грунта (глина) [11].

Еквівалентний діаметр вертикального електрода (кутка) [11]:

$$Dв=0,95 \cdot K= 0,95 \cdot 35 = 47,5 \text{ мм.} = 0,0332 \text{м.}$$

де *K= 35 мм.* – розмір металевого кутка (заданий).

Відношення *A/L=3/2=1,5.*

Кафедра_КбПЗ_2022рік

Опір розтіканню електричного струму одного електрода вертикального заземлювача з урахуванням заглиблення заземлювача [11]:

$$R_O=0,366(\rho/L)[lg(2L/Dв)+(1/2)lg((4T+L)/(4T-L))]=$$

$$=0,366(54,5/2)[(lg(2\cdot2/0,0332)+(1/2\ )lg((4\cdot1,8+2)/(4\cdot1,8-2))]=$$

$$=21,94\ Ом.$$

Визначаємо коефіцієнт екранування вертикальних електродів $Кев=0,8$ при орієнтовній кількості вертикальних електродів, яке дорівнює 4 [11].

Визначаємо необхідну кількість вертикальних заземлювачів (без врахування горизонтального заземлювача), при $R_{ЗН} = 4\ Ом$:

$$N=R_O /(Кев\ R_{ЗН})= 21,94 / (0,8\cdot4)=6,85 \approx 7\ шт.$$

Визначаємо довжину з'єднуючої полоси:

$$L_П =1,05\cdot A\cdot N= 1,05\cdot3\cdot6,85=21,6 \approx 22\ м.$$

Опір розтіканню електричного струму з'єднуючої полоси з урахуванням кліматичного коефіцієнта питомого опору грунта $K_П$ [11]:

$$R_П =0,366(\rho_2\cdot K_П/L_П)lg(2(L_П \cdot L_П)/(B\cdot t))=$$

$$=0,366(40\cdot5/21,6)\cdot lg((2\cdot21,6^2)/(0,04\cdot0,8))=15,13\ Ом.$$

де $K_П=5$ – табличне значення кліматичного коефіцієнта питомого опору грунта для відповідної кліматичної зони для з'єднуючої полоси [11]: $B = 40\ мм. = 0,04\ м.$- ширина з'єднуючої полоси (задана).

Загальний опір розтіканню електричного струму заземлювача [11]:

$$R =(R_O\cdot R_П)/(R_O\cdot\eta_П+ N\cdot R_П\cdot Кев)=$$

$$=(21,94\cdot 15,13)/(21,94\cdot 0,75+ 6,85\cdot 15,13\cdot 0,8)=3,33\ Ом.$$

де $\eta_П = 0,75$ – табличне значення коефіцієнта екранування з'єднуючої полоси [11].

Умова $R \le R_{ЗН}$ виконується ($3,33 \le 4$).

Так як R суттєво більше $R_{ЗН}$, зменшимо кількість вертикальних електродів до 6 і виконаємо перерахунок. У результаті остаточно отримали: кількість вертикальних електродів дорівнює 6 при $R =3,8\ Ом.$

Кафедра_КОПЗ – 2022рік

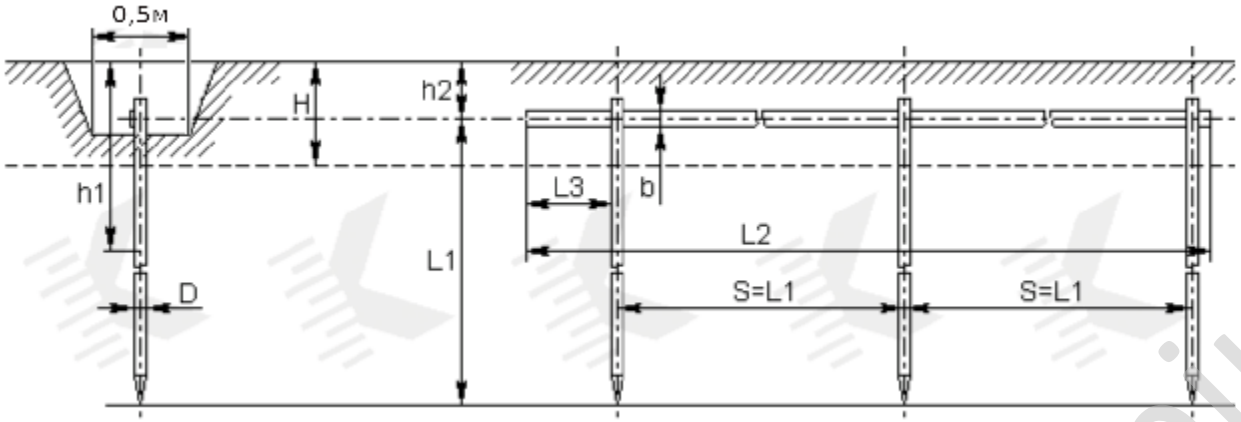

Рисунок 8.1 – Схема штучного заземлення

При необхідності можна зменшити кількість електродів заземлювача зменшивши загальний опір розтіканню електричного струму заземлювача методом зменшення питомого опору ґрунта, домішуючи у грунт безпосередньо навколи електродів заземлювача розчини солей NaCl, CaCl, сажу, соду, шлак, коксову дрібницю, або спеціальні суміші.

8.5 Висновки до розділу

Дотримання всіх необхідних умов праці не лише сприяє збереженню здоров'я працівників, а також підвищує ефективність виробництва в цілому.

З цих міркувань було здійснено аналіз приміщення, призначеного для праці програмістів, проведено розгляд небезпечних та шкідливих факторів, що негативно впливають на програмістів під час роботи. Виконано розрахунок захисного штучного заземлення, як одного з ключових факторів безпеки програміста. Розроблено заходи з охорони праці.

Кафедра_КбПЗ_2022рік

9 ОСНОВНІ ВИСНОВКИ

Програмне забезпечення, створене в результаті виконання випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти, призначено для системи даних карти тахографа.

В межах України в недостатній мірі представлені вітчизняні розробки в цій області.

У випускній кваліфікаційній роботі за другим (магістерським) рівнем вищої освіти наведені теоретичне узагальнення й рішення наукового завдання дослідження методів даних карти тахографа.

Рішення даного завдання полягало у вирішенні наступних задач:

– Був проведений огляд існуючих систем даних карти тахографа.

– Досліджена система даних карти тахографа.

– На основі отриманих результатів досліджень створена програмна реалізація системи даних карти тахографа.

Розроблені під час виконання випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти алгоритми дозволяють успішно вирішувати завдання даних карти тахографа.

Проведено аналіз предметної галузі в ході якого були виявлені об'єкти, взаємодія яких носить істотний характер для функціональної діяльності предметної галузі, і їхні основні характеристики; побудована алгоритм і вибраний середовище розробки.

Розроблене програмне забезпечення має простий, дружній та зручний інтерфейс користувача, що забезпечує легкість у освоєнні роботи програмного продукту, зручність у використанні, і не потребує особливих спеціальних знань.

При створені програмного забезпечення було використано об’єктно-орієнтований підхід, що відповідає сучасним тенденціям у галузі розробки комерційних програмних систем.

Кафедра_КОПЗ_2022рік

Програма реалізована на мові високого рівня Embarcadero Delphi. Дана мова програмування дозволяє найбільш ефективно обробляти дані. Це дозволило мінімізувати строк розробки програмного забезпечення, і, як слід, зменшити витрати на його розробку. Запропоноване програмне забезпечення ділиться на загальне програмне забезпечення, що поставляється із засобами обчислювальної техніки й спеціальне програмне забезпечення, що спеціально розроблене для даної конкретної системи й включає програми, що реалізують її функції.

Програма призначена для виконання під управлінням багатозадачної операційної системи Windows 10/11.

Даються необхідні рекомендації з установки розробленого програмного забезпечення.

Для підвищення рівня безпеки запропоновано застосовувати алгоритм PRESENT.

В цілому створене програмне забезпечення підтверджує правильність використаних проектних рішень та повністю відповідає вимогам технічного завдання. Створене програмне забезпечення має потенційну можливість для подальшого вдосконалення і застосування у різних галузях.

Розроблена програма має реальний економічний ефект від її впровадження у виробництво у сумі 76138 грн. З урахуванням вартості розробки програми та обладнання, строк окупления становить 0,3 роки.

Кафедра_КОПЗ_2022рік

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Шевчук І.І. Дослідження та програмна реалізація системи даних карти тахографа // Збірник праць молодих науковців ЦНТУ. – Вип. 13. – Кропивницький: ЦНТУ, 2022.
2. Абрамов П.И., Матвеев П.П. Защита информации в системах с использованием смарт-карт. – Расчёты и операционная работа в коммерческом банке, №1/2002, С. 42-47;
3. Абрамов П.И., Матвеев П.П. Каждому по ОСКАРу: открытая платформа разработки платёжних приложений на базе российской интеллектуальной карты. – Конфидент, №5'2001, С. 77-79;
4. Матвеев П.П. Защита программного обеспечения от несанкционированного копирования с применением интеллектуальних карт. Безопасность информационних технологий, №4’2003, С. 55-60;
5. Kovalenko O., Popereshnyak S., Grinenko S., Grinenko O., Radivilova T. «Methods for Assessing the Maturity Levels of Software Ecosystems». *CEUR Workshop Proceedings* Volume 2654, 2019, Pages 251-261. Режим доступу: https://www.scopus.com/record/display.uri?eid=2-s2.0-85083214331&origin=AuthorNamesList&txGid=5633fba897776a6e0f3d5633fbcd3fbe **(Scopus)**.
6. Kovalenko O., Drieieva H., Simakhin V., Bondar S., Drieiev O., Zhumadilova M. «Multifractal Properties of Traffic Generator Based on Markov Chains ». *CEUR Workshop Proceedings* Volume 2588, 2019, Pages 567-579. Режим доступу: https://www.scopus.com/record/display.uri?eid=2-s2.0-85083214331&origin=AuthorNamesList&txGid=176e2cada8976a6e0f3d5633fbcd3fbe **(Scopus)**.
7. Kovalenko Oleksandr Qualitative risk analysis of software development / Oleksandr Kovalenko, Jamil Al-Azzeh, Oleksii Smirnov, Anna Kovalenko, Serhii Smirnov // Asian Journal of Information Technology. – Volume 17 Issue 3. – Medwell Journals. – 2018. – P. 218-230. ISSN: 1682-3915.

Кафедра_КБПЗ_2022рік

URL: http://medwelljournals.com/abstract/?doi=ajit.2018.218.230 Doi: ajit.2018.218.230

8. Kovalenko Oleksandr, The mathematical model of the testing technology for DOM XSS vulnerabilities / O. Kovalenko, O. Smirnov, A.Kovalenko, S. Smirnov, V. Vialkova // Scientific & practical cyber security journal (SPCSJ) Volume 2 Issue 1, P. 22-28. Georgia. Tbilisi. Scientific Cyber Security Association (SCSA), 2018 ISSN: 2587-4667. URL: https://journal.scsa.ge/wp-content/uploads/2018/12/04-3-o.kovalenko-a.kovalenko-o.smirnov-s.smirnov-v.vialkova.pdf

9. Коваленко А.В. Технология тестирования DOM XSS уязвимости / А.В. Коваленко, А.С. Коваленко, А.А. Смирнов, С.А. Смирнов // Scientific & practical cyber security journal (SPCSJ) Volume 1. Issue 1. P. 38-45 Georgia. Tbilisi. Scientific Cyber Security Association (SCSA), 2017 ISSN: 2587-4667. URL: https://journal.scsa.ge/wp-content/uploads/2018/12/8-dom-xss-testing-technology-vulnerabilities.pdf

10. Коваленко О.В. Моделі та методи розроблення програмного забезпечення комп'ютерних систем для підвищення безпеки даних: монографія / О.В. Коваленко // К.: Вид. «КОД» – 2019. – 305 с.

11. Коваленко А.В. Методы качественного анализа и количественной оценки рисков разработки программного обеспечения / А.В. Коваленко, А.А. Смирнов // Информационные технологии в управлении, образовании, науке и промышленности: монография / Под редакцией профессора В.С. Пономаренко. – Х.: Видавець Рожко С.Г., 2016. – 566 с.

12. Коваленко А.В. Разработка метода управления рисками разработки программного обеспечения / А.В. Коваленко, А.А. Смирнов // Інформаційні технології: проблеми та перспективи: монографія / За загальною редакцією В.С. Пономаренка. – Х.: Видавець Рожко С.Г., 2017. – 447 с.

13. Коваленко А.В. Комплекс математических моделей технологии тестирования web-приложений / А.В. Коваленко, А.А. Смирнов // Інформаційні технології: сучасний стан та перспективи: монографія / За загальною редакцією

Кафедра_КБПЗ_2022рік

В.С. Пономаренка. – Х.: ТОВ «ДІСА ПЛЮС», 2018. – 461 с.

14. Коваленко А.В. Задачи распознавания ситуаций в ERP системах / А.В. Коваленко, А.А. Смирнов, А.С. Коваленко // Збірник наукових праць "Системи обробки інформації". – Випуск 4(120). – Х.: ХУПС – 2014. – С. 161-164.

15. Коваленко А.В. Методы качественного анализа и количественной оценки рисков разработки программного обеспечения / А.В. Коваленко, А.А. Смирнов // Збірник наукових праць "Системи обробки інформації". – Випуск 5(142). – Х.: ХУПС – 2016. – С. 153-157.

16. Коваленко А.В. Проблемы анализа и оценки рисков информационной деятельности / А.В. Коваленко, А.А. Смирнов, Н.Н. Якименко, А.П. Доренский // Збірник наукових праць "Системи обробки інформації". – Випуск 3(140). – Х.: ХУПС – 2016. – С. 40-42.

17. Коваленко А.В. Метод качественного анализа рисков разработки программного обеспечения / А.В. Коваленко, А.А. Смирнов, Н.Н. Якименко, А.П. Доренский // Наука і техніка Повітряних Сил Збройних Сил України. – Випуск 2(23). – Харків: ХУПС. – 2016. – С. 150-158.

18. Коваленко А.В. Метод количественной оценки рисков разработки программного обеспечения / А.В. Коваленко, А.А. Смирнов, Н.Н. Якименко, А.П. Доренский // Збірник наукових праць Харківського університету Повітряних Сил. Випуск 2 (47). – Харків: ХУПС. – 2016. – С. 128-133.

19. Коваленко А.В. Использование псевдобулевых методов бивалентного программирования для управления рисками разработки программного обеспечения / А.В. Коваленко, А.А. Смирнов // Системи управління, навігації та зв'язку. – Випуск 1 (37). – Полтава: ПолтНТУ. – 2016. – С. 98-103.

20. Коваленко А.В. Метод управления рисками разработки программного обеспечения / А.В. Коваленко // Системи управління, навігації та зв'язку. – Випуск 2 (38). – Полтава: ПолтНТУ. – 2016. – С. 93-100.

21. Коваленко А.В. Технология тестирования уязвимости к SQL

Кафедра_КБПЗ_2022рік

инъекциям / А.В. Коваленко // Системи управління, навігації та зв'язку. – Випуск 5 (45). – Полтава: ПолтНТУ. – 2017. – С. 66-71.

22. Коваленко А.В. Масштабирование имитационной модели технологии тестирования безопасности / А.В. Коваленко // Системи управління, навігації та зв'язку. – Випуск 6 (46). – Полтава: ПолтНТУ. – 2017. – С. 181-184.

23. Коваленко А.В. Имитационная модель технологии тестирования безопасности Web-приложений / А.В. Коваленко // Системи управління, навігації та зв'язку. – Випуск 1 (47). – Полтава: ПолтНТУ. – 2018. – С. 114-123.

24. Коваленко О.В. Методи якісного аналізу та кількісної оцінки ризиків розроблення програмного забезпечення/ О.В. Коваленко // Системи управління, навігації та зв'язку. – Випуск 3 (49). – Полтава: ПолтНТУ. – 2018. – С. 116-125.

25. Коваленко О.В. Управління ризиками розроблення програмного забезпечення за умови обмеженості коштів виділених на усунення помилок безпеки/ О.В. Коваленко // Техніка в сільськогосподарському виробництві, галузеве машинобудування, автоматизація. – Випуск 31. – Кропивницький: ЦНТУ. – 2018. – С. 128-140.

26. Коваленко О.В. GERT-мережевий синтез технології тестування на вразливість WEB-додатків/ О.В. Коваленко // Захист інформації. – Випуск 20(2) – К.: НАУ. – 2018. – С. 89-94.

27. Коваленко О.В. Імітаційна модель технології тестування безпеки на основі положень теорії масштабування / О.В. Коваленко // Безпека інформації. – Випуск 24 (2). – К.: НАУ. – 2018. – С. 110-117.

28. Коваленко О.В. Оцінка ефективності технології тестування безпеки / О.В. Коваленко // Вчені записки Таврійського національного університету імені В.І. Вернадського. Серія: Технічні науки. Том 29 (68) № 2, 2018. – С. 137-141

29. Коваленко О.В. Методи та засоби управління безпекою додатків / О.В. Коваленко // Інформаційно-керуючі системи на залізничному транспорті. №4, 2018. – С. 41-44.

Кафедра_КБПЗ_2022рік

30. Коваленко О.В. Розробка інформаційної технології передтестової компіляції та розподілу доступу / О.В. Коваленко // Системи управління, навігації та зв'язку. – Випуск 4 (50). – Полтава: ПолтНТУ. – 2018. – С. 115-119.

31. Коваленко О.В. Аналіз та дослідження інформаційних технологій розробки програмного забезпечення/ О.В. Коваленко // Вчені записки Таврійського національного університету імені В.І. Вернадського. Серія: Технічні науки. Том 29 (68) № 5, 2018. – С. 131-137.

32. Коваленко О.В. Удосконалений метод управління ризиками розроблення програмного забезпечення на основі напівмарковської моделі прийняття рішень/ О.В. Коваленко // Сучасні інформаційні системи. – Випуск 2(3). – Харків. – 2018. – С. 41-48.

33. Коваленко О.В. Математичні моделі технології тестування DOM XSS вразливості та вразливості до SQL ін'єкцій / О.В. Коваленко // Вісник Черкаського державного технологічного університету. Серія : Технічні науки №4, 2018. – С. 29-36.

34. Коваленко О.В. Математична модель технології тестування вразливості до SQL ін'єкцій / О.В. Коваленко // Системи управління, навігації та зв'язку. – Випуск 6 (58). – Полтава: ПолтНТУ. – 2019. – С. 43-47.

35. Коваленко О.В. Математична модель технології тестування комплексу DOM XSS вразливостей для аналітичної оцінки часових витрат / О.В. Коваленко // Центральноукраїнський науковий вісник. Технічні науки. № 2(33). с. 173-180, 2019.

36. Коваленко А.В. Проблемы анализа и оценки рисков информационной деятельности / А.В. Коваленко, А.А. Смирнов // Збірник наукових праць II міжнародної науково-практичної конференції «Актуальні питання забезпечення кібернетичної безпеки та захисту інформації». м. Київ. 24-27 лютого 2016 р. – Київ: Європейський університет. – 2016. – С. 138-139.

37. Коваленко А.В. Анализ и оценка рисков разработки программного обеспечения / А.В. Коваленко, А.А. Смирнов // Збірник тез «Securitea

Кафедра_КБПЗ_2022рік

informationala 2015-2016». Conferenta internationala (editia a XII-a). Chisinau. Moldova. 3 martie 2016. – Chisinau: ADSEM. – 2016. – P. 96-102.

38. Коваленко А.В. Исследование источников и причин риска разработки программного обеспечения, этапов и работ, при выполнении которых возникает риск / А.В. Коваленко, А.А. Смирнов // Збірник тез VII всеукраїнської науково-практичної конференції "Інформатика та системні науки (ІСН-2016)". м. Полтава. 10-12 березня 2016 р. – Полтава.: ПУЕТ – 2016. – С. 264-266.

39. Коваленко А.В. Оценка показателя чистой приведенной стоимости для количественной оценки рисков проекта разработки программного обеспечения / А.В. Коваленко, А.А. Смирнов // Збірник тез науково-практичної конференції “Проблеми кібербезпеки інформаційно-телекомунікаційних систем”. м. Київ. 10-11 березня 2016 р. – Київ: КНУ ім. Тараса Шевченко – 2016. – С. 81-82.

40. Коваленко А.В. Методика структурной идентификации рисков разработки программного обеспечения / А.В. Коваленко, А.А. Смирнов // Збірник тез Міжнародної науково-практичної конференції «Інформаційна безпека та комп'ютерні технології» (IS&CT). м. Кіровоград. 24-25 березня 2016 р. – Кіровоград: КНТУ. – 2016. – С. 71-72.

41. Коваленко А.В. Методы качественного анализа рисков разработки программного обеспечения / А.В. Коваленко, А.А. Смирнов // Збірник тез першої міжнародної науково-практичної конференції «Проблеми науково-технічного та правового забезпечення кібербезпеки у сучасному світі» (ПНПЗК-2016). м. Харків. 30 березня – 1 квітня 2016 р. – Харків: НТУ «ХПІ». – 2016. – С. 6-7.

42. Коваленко А.В. Структурная идентификация рисков разработки программного обеспечения / А.В. Коваленко, А.А. Смирнов // Збірник тез XVIII міжнародного науково-практичного семінару «Комбінаторні конфігурації та їх застосування». м. Кіровоград. 15-16 квітня 2016 р. – Кіровоград: КНТУ. –

Кафедра_КБПЗ_2022рік

2016. – С. 175-182.

43. Коваленко А.В. Исследование разработанной методики структурной идентификации рисков разработки программного обеспечения / А.В. Коваленко, А.А. Смирнов // Збірник тез VIII міжнародної науково-практичної конференції “Проблеми і перспективи розвитку ІТ-індустрії”. м. Харків. 28-29 квітня 2016 р. – Харків: ХНЕУ. – 2016. – С. 49.

44. Коваленко А.В. Исследование дерева рисков разработки программного обеспечения / А.В. Коваленко, А.А. Смирнов // Збірник тез III міжнародної науково-практичної конференції «Інформаційна та економічна безпека» (INFECO-2016)». м. Харків. 28-30 квітня 2016 р. – Харків: ХНМІ ДВНЗ «УБС». – 2016. – С. 174-178.

45. Коваленко А.В. Методы качественного анализа и количественной оценки рисков разработки программного обеспечения / А.В. Коваленко, А.А. Смирнов, А.С. Коваленко // Сборник тезисов XII международной конференции "Стратегия качества в промышленности и образовании". г. Варна. Болгария. 30 мая – 02 июня 2016 г – Варна. ТУВ. – 2016. – С. 585-589.

46. Коваленко А.В. Разработка метода управления рисками разработки программного обеспечения / А.В. Коваленко, А.А. Смирнов, А.С. Коваленко // Матеріали Всеукраїнської науково-практичної конференції «Кібербезпека в Україні: правові та організаційні питання». м. Одеса, 21 жовтня 2016 р. – Одеса : ОДУВС, 2016. – С.146-148.

47. Коваленко А.В. Метод управления рисками разработки программного обеспечения с использованием псевдобулевых методов бивалентного программирования / А.В. Коваленко, А.А. Смирнов // Матеріали Всеукраїнської науково-практичної конференції «Актуальні задачі та досягнення у галузі кібербезпеки». м. Кропивницький, 23-25 листопада 2016 року – Кропивницький: ЦНТУ, 2016. – С. 162.

48. Коваленко А.В. Псевдобулевые методы бивалентного программирования для управления рисками разработки программного

Кафедра_КБПЗ_2022рік

обеспечения / А.В. Коваленко, А.А. Смирнов, А.С. Коваленко, С.А. Смирнов // Збірник наукових праць III міжнародної науково-практичної конференції «Актуальні питання забезпечення кібернетичної безпеки та захисту інформації». м. Київ. 22-25 лютого 2017 р. – Київ: Європейський університет. – 2017. – С. 158-162.

49. Коваленко А.В. Метод управления рисками разработки программного обеспечения / А.В. Коваленко, А.А. Смирнов // Збірник тез II науково-практичної конференції "Проблеми кібербезпеки інформаційно-телекомунікаційних систем". м. Київ. 23-24 березня 2017 р. – Київ: КНУ ім. Тараса Шевченко – 2017. – С. 203-205.

50. Коваленко А.В. Алгоритмы анализа уязвимостей при управлении рисками разработки программного обеспечения / А.В. Коваленко, А.А. Смирнов, А.С. Коваленко // Conferenta internationala (editia a XIII-a). «Securitea informationala 2017». Chisinau. Republic of Moldova. 4-5 aprilie 2017. – Chisinau: ADSEM. – 2017. – P. 19-22.

51. Коваленко А.В. Алгоритм анализа DOM XSS уязвимости при управлении рисками разработки программного обеспечения / А.В. Коваленко, А.А. Смирнов, А.С. Коваленко // Збірник тез дев'ятнадцятого міжнародного науково-практичного семінару «Комбінаторні конфігурації та їх застосування». м. Кропивницький 7-8 квітня 2017 р. – Кропивницький: ГЛА НАУ. – 2017. – С. 125-127.

52. Коваленко А.В. Алгоритм анализа уязвимости SQL Injection для управления рисками разработки программного обеспечения / А.В. Коваленко, А.А. Смирнов, А.С. Коваленко // Збірник тез другої міжнародної науково-технічної конференції «Проблеми науково-технічного та правового забезпечення кібербезпеки у сучасному світі» (ПНПЗК-2017). м. Харків.10-12 квітня 2017 р. – Харків: НТУ «ХПІ». – 2017. – С. 27.

53. Коваленко А.В. Метод управления рисками разработки программного обеспечения на основе алгоритмов анализа

Кафедра_КБПЗ_2022рік

уязвимостей / А.В. Коваленко, А.А. Смирнов, А.С. Коваленко // Збірник тез Міжнародної науково-практичної конференції «Інформаційна безпека та комп'ютерні технології» (IS&CT). м. Кіровоград. 20-22 квітня 2017 р. Кіровоград: КНТУ. – 2017. – С. 92.

54. Коваленко А.В. Алгоритмы анализа DOM XSS уязвимости и уязвимости SQL Injection при управлении рисками разработки программного обеспечения / А.В. Коваленко, А.А. Смирнов, А.С. Коваленко // Збірник тез IX міжнародної науково-практичної конференції "Проблеми і перспективи розвитку IT-індустрії". м. Харків. 20-21 квітня 2017 р. – Харків: ХНЕУ. – 2017. – С. 61.

55. Kovalenko O.V. Method of testing the dom xss vulnerability / Kovalenko Oleksandr, Kovalenko Anna, Smirnov Oleksii, Smirnov Serhii // International Conference «information technologies, systems and networks ITSN-2017». Chisinau, Republic of Moldova. 17 – 18 October 2017. – Chisinau: Academy of Sciences of Moldova, Military Academy of Armed Forces "Alexandru cel Bun". – 2017. – P. 7.

56. Коваленко О.В. Метод тестування DOM XSS уразливості / О.В. Коваленко, О.А. Смірнов, А.С. Коваленко, С.А. Смірнов // Збірник тез всеукраїнської науково-практичної інтернет-конференції «Автоматика та комп'ютерно-інтегровані технології у промисловості, телекомунікаціях, енергетиці та транспорті». м. Кропивницький. 16-17 листопада 2017 р. – Кропивницький: ЦНТУ. – 2017. – С. 198-199.

57. Коваленко О.В. GERT-модель технології тестування DOM XSS уразливості / О.В. Коваленко, А.С. Коваленко, О.А. Смірнов, С.А. Смірнов // Збірник наукових праць IV міжнародної науково-практичної конференції «Актуальні питання забезпечення кібернетичної безпеки та захисту інформації». м. Київ. 21-24 лютого 2018 р. – Київ: Європейський університет. – 2018. – С. 65-70.

58. Коваленко О.В. Технології тестування уразливостей Web-застосунків з використанням GERT-моделі / О.В. Коваленко, А.С. Коваленко,

Кафедра_КБПЗ_2022рік

О.А. Смірнов, С.А. Смірнов // Збірник тез всеукраїнської науково-практичної конференції "Комп'ютерні інтелектуальні системи та мережі (КІСМ-2018)". м. Кривий Ріг. 21-23 березня 2018 р. – Кривий Ріг.: ДВНЗ КНУ – 2018. – С. 227-230.

59. Коваленко А.В. Тестирование уязвимости Web-приложений к атаке вида межсайтовый скриптинг / А.В. Коваленко, А.С. Коваленко, А.А. Смирнов, С.А. Смирнов // Збірник тез «Securitea informationala 2018». Conferenta internationala (editia a XIV-a). Chisinau. Moldova. 20-21 martie 2018. – Chisinau: ADSEM. – 2018. – P. 54-56.

60. Коваленко А.В. Комплекс математических моделей технологии тестирования web-приложений / А.В. Коваленко, А.С. Коваленко, А.А. Смирнов, С.А. Смирнов // Збірник тез X міжнародної науково-практичної конференції “Проблеми і перспективи розвитку IT-індустрії”. м. Харків. 19-20 квітня 2018 р. – Харків: ХНЕУ. – 2018. – С. 38.

61. Державні будівельні норми України: ДБН В.2.5-28:2018. – Режим доступу до ресурсу: https://goo.su/9AkQ

62. Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин: ДСанПІН 3.3.2-007-98. – Режим доступу до ресурсу: https://zakon.rada.gov.ua/rada/show/v0007282-98

63. Закон України «Про охорону праці» від 14.10.1992 р. № 2694-XII. – Режим доступу до ресурсу: https://zakon.rada.gov.ua/laws/show/2694-12

64. Зеркалов Д. В. Охорона праці в Галузі: Загальні вимоги: навч. посіб. Київ: Основа. 2011. 551 с.

65. Наказ Міністерства соціальної політики України 14.02.2018 № 207 «Про затвердження Вимог щодо безпеки та захисту здоров’я працівників під час роботи з екранними пристроями». – Режим доступу до ресурсу: https://zakon.rada.gov.ua/laws/show/z0508

66. Охорона праці. Ч. 1. Захисне заземлення: метод. вказ. до викон.

Кафедра_КОТЗ_2022рік

розрахунків з викор. персон. ЕОМ ІВМ сумісного типу / Кіровоград. ін-т с.-г. машинобуд.; [укл. О. В. Оришака, Є. К. Солових, В. О. Оришака]. – Кіровоград: КІСМ, 1997. – 20 с. Режим доступу до ресурсу: http://dspace.kntu.kr.ua/jspui/handle/123456789/4358

67. Постанова № 42 від 01.12.1999 Головного державного санітарного лікаря України «Санітарні норми мікроклімату виробничих приміщень ДСН 3.3.6.042-99. – Режим доступу до ресурсу: https://zakon.rada.gov.ua/rada/show/va042282-99

68. Сакулин В.П., Шептовицкий В.М. Безопасность труда при монтаже и эксплуатации электроустановок / В.П.Сакулин, В.М.Шептовицкий. – Л. : “Колос”, 1973. – 238 с.

69. Центр післядипломної освіти та підвищення кваліфікації. – Режим доступу до ресурсу: https://cpo.stu.cn.ua

70. Оришака, О. В. Основи охорони праці: навч. посіб. / О. В. Оришака, Г. П. Горбачова, К. М. Марченко; М-во освіти і науки України, Центральноукраїн. нац. техн. ун-т. – Кропивницький : ЦНТУ, 2022. – 175 с. – Режим доступу до ресурсу: http://dspace.kntu.kr.ua/jspui/handle/123456789/12161 (дата звернення 19.09.22).

Кафедра_КБПЗ_2022рік

Додаток А
(обов'язковий)

Технічне завдання

Зміст

Кафедра_КОПЗ_2022рік

					ВКРМ-123.22.0028.00.00.ТЗ		
Вим.	Арк.	№ документа	Підпис	Дата			
Розробив		Шевчук І.І.			Дослідження та програмна реалізація системи даних карти тахографа	Літ.	Аркуш 1 / Аркушів 6
Перевірив		Смірнов О.А.				М	
Н. Контр.		Гермак В.С.				ЦНТУ КІ-21М-1,4	
Затв.		Смірнов О.А.					

1 Найменування та область застосування

Це технічне завдання розповсюджується на дослідження та програмну реалізацію системи даних карти тахографа.

2 Підстава для розробки

Підставою для розробки служить завдання на випускну кваліфікаційну роботу за другим (магістерським) рівнем вищої освіти, видане на кафедрі кібербезпеки та програмного забезпечення (нак. № 19-13 від 17.08.2022 року).

3 Мета та призначення розробки

Метою випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти є дослідження та програмна реалізація системи даних карти тахографа.

4 Джерела розробки

Джерелом цієї випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти є стосовна до теми література і існуючі аналоги.

5 Технічні вимоги

5.1 Склад продукції

Складниками розробки є:

– вибір і обґрунтування методів реалізації проекту;

– розробка програмної частин системи, а також розробка взаємодії системи з ОС та з користувачем;

Кафедра_КбПЗ_2022рік

– техніко-економічне обґрунтування доцільності прийнятого до розробки програмного забезпечення;

– аналіз умов праці;

– розробка програми, що реалізує спроектовані алгоритми роботи системи.

5.2 Показники призначення

Система повинна забезпечувати:

– програмну реалізацію системи даних карти тахографа;

– цілісність даних у процесі роботи та при зберіганні;

– простий, інтуїтивно зрозумілий інтерфейс.

5.3 Вимоги до функціональних характеристик

Розроблене програмне забезпечення не повинно мати обмежень на версію драйверів та операційної системи.

5.4 Вимоги до архітектури

Компонент, що розробляється повинен використовувати системні засоби та апаратні засоби, що на даному етапі розвитку обчислювальної техніки найбільше поширені.

5.5 Вимоги до надійності

Програмні модулі написані по всім правилам, які стосуються стандартних викликів процедур, функцій, методів і форм, визначених технічною документацією на середовище розробки.

Кафедра_КбПЗ_2022рік

5.6 Умови експлуатації

Робочі місця користувачів ПЗ повинні задовольняти наступним умовам експлуатації:

– температура повітря: 19-20 град. по Цельсію;

– відносна вологість повітря до 80%;

– атмосферний тиск 107 кПа.

5.7 Вимоги до складу та параметрів технічних засобів

Програмне забезпечення повинно бути реалізоване на ПЕОМ архітектури IBM PC, працювати в ОС Windows 10/11 і з сумісними з цією платформою пристроями і прикладним програмним забезпеченням.

5.8 Вимоги до інформаційної і програмної сумісності

Переносність програмного забезпечення повинна бути забезпечена за рахунок його реалізації стандартного інтерфейсу взаємодії з ОС, що працюють під управлінням ОС Windows 10/11.

5.8.1 Обладнання

Комп’ютер Intel® Celeron/8 Mb/1.2 Gb/SVGA 14” 1Mb або сумісні з ним.

5.8.2 Мова програмування

Середовище Embarcadero Delphi.

Кафедра_КОПЗ_2022рік

5.8.3 Вхідні дані

Опис алгоритму роботи запропонованої системи.

5.8.4 Вихідні дані

Робоча програма.

6 Вимоги до програмної документації

Програмна продукція повинна бути представлена у виді опису структури даних, схем та опису алгоритму, а також текстів вихідних модулів програмного забезпечення згідно ЄСПД .

7 Економічні вимоги

7.1 Для ПЗ необхідно виробити функціонально-вартісний аналіз варіантів розробки.

7.2 Виконати розрахунок витрат показників економічного ефекту з урахуванням цін на 3 вересня 2022 року.

8 Вимоги щодо охорони праці

В частині охорони праці випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти повинна бути розглянута Розробка заходів з умов поліпшення охорони праці програміста.

Кафедра_КбПЗ_2022рік

9 Перелік документів, що розробляються

– Наукова новизна – 1 аркуш.
– Структурна схема системи – 1 аркуш.
– Функціональна схема системи – 1 аркуш.
– Діаграма процесів – 1 аркуш.
– Блок-схема алгоритму роботи програми – 2 аркуша.
– Показники економічної ефективності – 1 аркуш.
– Пояснювальна записка – 127 аркушів.

10 Етапи розробки

10.1 Збір і обробка інформації по темі випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти. Постановка задачі на виконання випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти (складання ТЗ).

10.2 Проведення досліджень або експериментальних робіт для уточнення основних положень випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти.

10.3 Розробка функціональних схем, блок схем алгоритмів роботи програмного забезпечення.

10.4 Побудова схем взаємодії даних.

10.5 Створення прототипу ПЗ.

10.6 Віднаходження ПЗ, аналіз отриманих результатів.

10.7 Робота над питанням охорони праці і техніки безпеки.

10.8 Розрахунок з техніко-економічного обґрунтування.

10.9 Оформлення пояснювальної записки і виконання робіт по графічній частині.

11 Порядок контролю та приймання

11.1 Подання випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти на попередній захист 10.12.2022 р.

11.2 Подання випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти на захист 21.12.2022 р.

Кафедра_КбПЗ_2022рік

Додаток Б

(обов’язковий)

Міністерство освіти і науки України

Центральноукраїнський національний технічний університет

ЗАТВЕРДЖУЮ

Керівник випускної кваліфікаційної роботи за

другим (магістерським) рівнем вищої освіти

_____ Смірнов О.А.

Дослідження та програмна реалізація системи даних карти тахографа

Лістинг програми

Код документу 12

Носій: CD/DVD-диск / USB-флеш-накопичувач

Загальна кількість аркушів: 29

Літера: РП

Кропивницький – 2022 року

Кафедра_КбПЗ_2022рік

Файл проекту тахографу для роботи з захищеною чіп-картою

```
program Chabak_Zahischeniy_Tachograph;

uses
  Forms,
  Unit1 in ' Zahischeniy_Tachograph_Unit1.pas' {Form1},
  Unit2 in ' Zahischeniy_Tachograph_Unit2.pas' {Form2},
  Unit3 in ' Zahischeniy_Tachograph_Unit3.pas' {Form3};
  Unit4 in ' Zahischeniy_Tachograph_Unit4.pas' {Form4};
  Unit5 in ' Zahischeniy_Tachograph_Unit5.pas' {Form5};
  Unit5 in ' Zahischeniy_Tachograph_Unit6.pas' {Form6};

{$R *.RES}

begin
    Form6:=TForm5.Create(Application);
    Form6.Show;
    Form6.Update;
Try
  Application.HintPause:=200;
  Application.HintHidePause:=7000;
  Application.HintShortPause:=25;
  Application.Initialize;
  Application.Title := 'Chabak_Zahischeniy_Tachograph';
     Application.CreateForm(TForm1, Form1);
     Application.CreateForm(TForm2, Form2);
     Application.CreateForm(TForm3, Form3);
     Application.CreateForm(TForm4, Form4);
     Application.CreateForm(TForm5, Form5);
  finally
  Form6.Free;
end;
Application.Run;
end.
```

Кафедра_КБПЗ_2022рік

Файл проекту Zahischeniy_Tachograph_Unit1.pas – організація захищеного стеку тахографа

```
//*************************************************************//
// Testuvalos`a u:  Embarcadero Delphi //
//  OS: WIN10/11                                          //
//                    Testova versiya tachographu                  //
// coded by: Shevchuk I.I.                                 //
//*************************************************************//
 unit Unit1;
 //Zahischeniy_Tachograph_Traffic;

 interface

 uses SysUtils, Windows, IPHelper, IPHLPAPI;


 type
 TZahischeniy_Tachograph_Traffic = Class;

 TNewInstanceEvent = procedure(Sender :TZahischeniy_Tachograph_Traffic) of
 object;
 TFreezeEvent = procedure(Sender :TZahischeniy_Tachograph_Traffic) of object;

 TZahischeniy_Tachograph_Traffic = Class
 private
 FIP: string;
 FMac: string;
 FInPerSec: Dword;
 FInTotal: Dword;
 FPeakInPerSec: Dword;
 FInterfaceIndex: DWord;
 FActiveCountIn: Dword;
 FSecondsActive: Cardinal;
 FPrevCountIn: DWord;
 FDescription: string;
 FOutTotal: Dword;
 FPeakOutPerSec: Dword;
 FOutPerSec: Dword;
 FPrevCountOut: DWord;
 FActiveCountOut: Dword;
 FAverageInPerSec: Dword;
 FAverageOutPerSec: Dword;
 FStartedAt: TDateTime;
 FRunning: boolean;
 FOnFreeze: TFreezeEvent;
 FOnUnFreeze: TFreezeEvent;
 FConnected: boolean;
 FFound: boolean;
 FSpeed: DWord;

 function GetIPFromIFIndex(InterfaceIndex: Cardinal): string;
 public
 property Found : boolean read FFound write FFound;
 property Connected : boolean read FConnected;
 property Running : boolean read FRunning;
 property InterfaceIndex : DWord read FInterfaceIndex;
 property IP : string read FIP;
 property Mac : string read FMac;
 property Description : string read FDescription;
 property StartedAt : TDateTime read FStartedAt;
 property SecondsActive : Cardinal read FSecondsActive;
 property Speed : DWord read FSpeed;
 property ActiveCountIn : Dword read FActiveCountIn;
 property PrevCountIn : DWord read FPrevCountIn;
 property InPerSec : Dword read FInPerSec; property AverageInPerSec : Dword read
 FAverageInPerSec;
 property InTotal : Dword read FInTotal;
```

Кафедра_КбПЗ_2022рік

```
property PeakInPerSec : Dword read FPeakInPerSec;
property ActiveCountOut : Dword read FActiveCountOut;
property PrevCountOut : DWord read FPrevCountOut;
property OutPerSec : Dword read FOutPerSec; property AverageOutPerSec : Dword
read FAverageOutPerSec;
property OutTotal : Dword read FOutTotal;
property PeakOutPerSec : Dword read FPeakOutPerSec;
procedure NewCycle(const InOctets, OutOctets, TrafficSpeed : Dword);
procedure Reset;
procedure Freeze;
procedure UnFreeze;
procedure MarkDisconnected;
function GetStatus : string;
function FriendlyRunningTime:string;
constructor Create(const AMibIfRow : TMibIfRow; OnNewInstance :
TNewInstanceEvent);
published
property OnFreeze :TFreezeEvent read FOnFreeze write FOnFreeze;
property OnUnFreeze :TFreezeEvent read FOnUnFreeze write FOnUnFreeze;
end;

function BytesToFriendlyString(Value : DWord) : string;
function BitsToFriendlyString(Value : DWord) : string;

implementation

function BytesToFriendlyString(Value : DWord) : string;
const
OneKB = 1024;
OneMB = OneKB * 1024;
OneGB = OneMB * 1024;
begin
if Value < OneKB then
Result := FormatFloat('#,##0.00 B',Value)
else
if Value < OneMB then
Result := FormatFloat('#,##0.00 KB', Value / OneKB)
else
if Value < OneGB then
Result := FormatFloat('#,##0.00 MB', Value / OneMB)
end; (*BytesToFriendlyString*)

function BitsToFriendlyString(Value : DWord) : string;
const
OneKB = 1000;
OneMB = OneKB * 1000;
OneGB = OneMB * 1000;
begin
if Value < OneKB then
Result := FormatFloat('#,##0.00 bps',Value)
else
if Value < OneMB then
Result := FormatFloat('#,##0.00 Kbps', Value / OneKB)
else
if Value < OneGB then
Result := FormatFloat('#,##0.00 Mbps', Value / OneMB)
end; (*BytesToFriendlyString*)


{ TZahischeniy_Tachograph_Traffic }
constructor TZahischeniy_Tachograph_Traffic.Create(const AMibIfRow: TMibIfRow;
OnNewInstance : TNewInstanceEvent);
var
Descr : string;
begin
inherited Create;

FRunning := True;
FConnected := True;
```

Кафедра_КбПЗ_2022рік

```
self.FInterfaceIndex := AMibIfRow.dwIndex;
self.FIP := GetIPFromIFIndex(self.InterfaceIndex);
self.FMac := MacAddr2Str(TMacAddress(AMibIfRow.bPhysAddr),
AMibIfRow.dwPhysAddrLen);

SetLength(Descr, Pred(AMibIfRow.dwDescrLen));
Move(AMibIfRow.bDescr, Descr[1], pred(AMibIfRow.dwDescrLen));
self.FDescription := Trim(Descr);

self.FPrevCountIn := AMibIfRow.dwInOctets;
self.FPrevCountOut := AMibIfRow.dwOutOctets;

self.FStartedAt := Now;


self.FSpeed := AMibIfRow.dwSpeed;

FActiveCountIn := 0;
FActiveCountOut:= 0;
FInTotal := 0;
FOutTotal:= 0;
FInPerSec:= 0;
FOutPerSec:= 0;
FPeakInPerSec := 0;
FPeakOutPerSec:=0;

if Assigned(OnNewInstance) then OnNewInstance(self);
end; (*Створення*)

procedure TZahischeniy_Tachograph_Traffic.NewCycle(const InOctets, OutOctets,
TrafficSpeed: Dword);
begin
Inc(self.FSecondsActive);

If not Running then Exit;

FSpeed := TrafficSpeed;

//IN
self.FInPerSec := InOctets - self.PrevCountIn;
Inc(self.FInTotal, self.InPerSec);
if InPerSec > 0 then Inc(FActiveCountIn);
if InPerSec > PeakInPerSec then FPeakInPerSec := InPerSec;
try
self.FAverageInPerSec := InTotal div ActiveCountIn
except
self.FAverageInPerSec := 0
end;
FPrevCountIn := InOctets;

//OUT
self.FOutPerSec := OutOctets - self.PrevCountOut;
Inc(self.FOutTotal, self.OutPerSec);
if OutPerSec > 0 then Inc(FActiveCountOut);
if OutPerSec > PeakOutPerSec then FPeakOutPerSec := OutPerSec;
try
self.FAverageOutPerSec := OutTotal div ActiveCountOut
except
self.FAverageOutPerSec := 0
end;
FPrevCountOut := OutOctets;
end; (*Новий цикл роботи з захищеною чіп-картою*)

function TZahischeniy_Tachograph_Traffic.GetIPFromIFIndex(InterfaceIndex:
Cardinal): string;
var
i:  integer;
IPArr : TMIBIPAddrArray;
```

Кафедра КбПЗ_2022рік

```
begin
Result := '!не знайдено!';
Get_IPAddrTableMIB( IpArr );
if Length(IPArr) > 0 then
for i := low(IPArr) to High(IPArr) do
if IPArr[i].dwIndex = InterfaceIndex then
begin
Result := IPAddr2Str(IParr[i].dwAddr);
BREAK;
end;
end; (*GetIPFromIFIndex*)

procedure TZahischeniy_Tachograph_Traffic.Reset;
begin
self.FPrevCountIn := InPerSec;
self.FPrevCountOut := OutPerSec;

self.FStartedAt := Now;
FSecondsActive := 0;

FActiveCountIn := 0;
FActiveCountOut:= 0;
FInTotal := 0;
FOutTotal := 0;
FInPerSec := 0;
FOutPerSec := 0;
FPeakInPerSec := 0;
FPeakOutPerSec := 0;
end; (*Перезавантаження*)

procedure TZahischeniy_Tachograph_Traffic.Freeze;
begin
FRunning := False;
if Assigned(FOnFreeze) then OnFreeze(Self);
end; (*Заблоковано*)

procedure TZahischeniy_Tachograph_Traffic.UnFreeze;
begin
FRunning := True;
if Assigned(FOnUnFreeze) then OnUnFreeze(Self);
end; (*Розблоковано*)

procedure TZahischeniy_Tachograph_Traffic.MarkDisconnected;
begin
self.FConnected := False;
self.FRunning := False;
end; (*Розірвано з'єднання*)

function TZahischeniy_Tachograph_Traffic.GetStatus: string;
begin
if  self.Connected then
Result := 'З'єднано'
else
Result := 'Роз'єднано';

if self.Running then
Result := Result + ', 1'
else
Result := Result + ', 2';
end; (*Отримання статусу роботи з захищеною чіп-картою*)

function TZahischeniy_Tachograph_Traffic.FriendlyRunningTime: string;
var
H, M, S: string;
ZH, ZM, ZS: Integer;
begin
ZH := SecondsActive div 3600;
ZM := Integer(SecondsActive) div (60 - ZH * 60);
ZS := Integer(SecondsActive) - (ZH * 3600 + ZM * 60);
```

Кафедра_КбПЗ_2022рік

```
H := Format('%.2d',[ZH]);
M := Format('%.2d',[ZM]);
S := Format('%.2d',[ZS]);

Result := H + ':' + M + ':' + S;
end; (*FriendlyRunningTime*)

{-------------------------
 Реалізація універсального циклу шифрування алгоритму
 криптографічного перетворення ДСТ 28147-89.
---------------------------
>Параметри  при  виклику:
   EAX=N1, EDX=N2;
   ESI – адреса першого елемента ключа;
   EBX – адреса таблиці замін;
   CX    – число основних кроків
>Результати на виході:
   EDX=N1, EAX=N2 для циклів 32-З, 32-Р;
   EAX=N1, EDX=N2 для циклу 16-З;
}

procedure TGOSTEncryption.Gost386;
asm
    {Внутрішній  цикл  роботи  П/П}
    {1. Початок  циклу  й  збереження  старого  N1}
@iloop:  push      EAX
    {2. Додавання  до  S  ключа  за модулем  2^32}
         add       EAX,[ESI] { додати    ключ}
         add       ESI,4     { наступний  елемент ключа}
    {3. Поблочна заміна в S з обертанням на 8 біт уліво}
         xlat                { перекодування  байта}
         ror       EAX,8     { AL <- наступний байт}
         add       ebx,256   { наступний вузол замін}

         xlat                { перекодування  байта}
         ror       EAX,8     { AL <- наступний байт}
         add       ebx,256   { наступний вузол замін}

         xlat                { перекодування  байта}
         ror       EAX,8     { AL <- наступний байт}
         add       ebx,256   { наступний вузол замін}

         xlat                { перекодування  байта}
         sub       ebx,3*256 { BX -> 1-й вузол замін}
    {4. Довернення  S  на  3  біти  вліво}
         rol       EAX,3
    {5. Обчислення  нових  значень  N1,N2}
         xor       EAX,EDX
         pop       EDX
    { Завершення  внутрішнього  циклу}
         loop      @iloop
end;

var vesi,vedi,veax,vebx,vecx,vedx:DWORD;

procedure PushAll;
asm {Збереження регістрів відповідно до угод}
    mov vesi,esi
    mov vedi,edi
    mov veax,eax
    mov vebx,ebx
    mov vecx,ecx
    mov vedx,edx
end;

procedure PopAll;
asm {Відновлення регістрів}
    mov esi,vesi
    mov edi,vedi
```

Кафедра_КСПЗ_2022рік

```
    mov eax,veax
    mov ebx,vebx
    mov ecx,vecx
    mov edx,vedx
end;

{-------------------------
 Генерація масиву криптографічної гами згідно
 криптоалгоритму ДСТ 28147-89.
----------------------------
 Зауваження:
 1. Дана процедура виробляє гаму  блоками по 8
 байтів, що відбиває властивість використаного алго-
 ритму бути блоковим шифром.
----------------------------
}

procedure TGOSTEncryption.GenerateGamma(dst:pointer;noblocks:DWORD);
var s1,s2:Item;
begin
     s1:=0;s2:=0;
     GenerateGamma(dst,noblocks,s1,s2);
end;

procedure TGOSTEncryption.GenerateGamma(dst:pointer;noblocks:DWORD;var
s1,s2:Item);

const
// визначаємо константи C1,C2 за ДСТ
    C1=$01010101;
    C2=$01010104;

var
    k:TPExtEncKey;  // адреса ключа
    c:TPExtTable;  // адреса таблиці замін
    l:DWORD;
    cur:pointer;

begin
    k:=@ExtKey;
    c:=@ExtChTab;
    l:=noblocks;
    cur:=dst;

    ExpandKey(@Key,k,8,4,KeyMask);

asm
         call      PushAll
         cmp       l,0
         je        @ex
                                  // настроювання регістрів
         mov       EBX,c          // EBX <- адреса таблиці замін
         mov       edi,s2
@circle: mov       esi,s1         // Додавання  констант
         mov       EAX,[esi]
         mov       EDX,[edi]
         add       EAX,C1         // зміна S1, s2:
         adc       EDX,C2         //    S1=(S1+C1) mod 2^32
                                  //    S2=(S2+C2) mod (2^ 32-1)
         adc       EDX,0          // якщо був перенос,
                                  // треба додати 1 до результату
         mov       [esi],EAX      // заносимо нові
         mov       [edi],EDX      // значення S1,S2 на їхнє місце
                                  // Готовимо регістри для виклику циклу 32-
3;
         mov       ESI,k          // ESI <- адреса ключа
         mov       ECX,32         // CX <- число основних кроків
         call      Gost386        // крок простої заміни
                                  // Використання згенерованого блоку гами
```

Кафедра_КОПЗ_2022рік

```
        mov     esi,cur
        mov     [ESI],EDX       // Заносимо 8-байтний блок
        mov     [ESI+4],EAX     // гами в область призначення
        add     cur,8           // корекція покажчика
        dec     l               // корекція лічильника
        jnz     @circle         // на генерацію нового блоку
@ex:    call    PopAll
end;
end;

procedure TGOSTEncryption.ApplyGamma(src:pointer;gamma:pointer;len:DWORD);
//Застосовуємо гаму
asm
        call    PushAll
        mov     esi,src
        mov     edi,gamma
        mov     ecx,len
        cmp     ecx,0
        je      @ex
@l1:    mov     al,[edi]
        xor     [esi],al
        inc     esi
        inc     edi
        loop    @l1
@ex:    call    PopAll
end;

{-------------------------
 Виробіток імітовставки для масиву даних згідно
 криптоалгоритму ДСТ 28147-89.
---------------------------
}

procedure TGOSTEncryption.GenerateImito(src:pointer;noblocks:DWORD);
begin
     GenerateImito(src,noblocks,0,0);
end;

procedure
TGOSTEncryption.GenerateImito(src:pointer;noblocks:DWORD;s1,s2:Item);

const ExtMask:DWORD=$2;

var
    l:DWORD;
    pt:TPExtTable;
    pk:TPExtEncKey;
    p1,p2:^DWORD;

begin
    ExpandKey(@Key,@ExtKey,8,2,ExtMask);
    l:=noblocks;
    pt:=@ExtChTab;pk:=@ExtKey;
    p1:=@I1;p2:=@I2;
asm
        call    PushAll
        mov     EAX,s1          // Завантаження початкового значення
контрольної комбінації
        mov     EDX,s2
                                // Початкове завантаження покажчиків
        mov     EBX,pt  // EBX <- адреса ТЗ
        mov     EDI,src         // EDI <- адреса даних
        cmp     l,0
        je      @ex
                                // Додаємо блок даних
@circle: xor    EAX,[EDI]       // Гаммуємо
        xor     EDX,[EDI+4]     // 8-байтный
                                // Виклик простої заміни
        mov     ECX,16          // CX <- число основних кроків
```

Кафедра_КбПЗ_2022рік

```
        mov     ESI,pk          // ESI <- адреса ключа
        call    Gost386
                                // Організація циклу
        add     EDI,8           //корекція адреси призначення
        dec     l
        jnz     @circle         // робимо цикл
@ex:    mov     esi,p1
        mov     [esi],eax       // заносимо нові значення
        mov     esi,p2
        mov     [esi],edx
        call    PopAll
end;
end;

{-------------------------
 Побудова розширеного ключа шифрування з однократ-
 ного ключа для шифру типу ДСТ 28147-89.
 ** Схема розширення задається маскою розширення.
 ** Маска розширення може мати розмір 1..32767 біт.
---------------------------
}
procedure TGOSTEncryption.ExpandKey
(src:TPEncKey;dst:TPExtEncKey;KeyLength:DWORD;
                KeyRepeat:DWORD;RepeatMask:DWORD);
asm
        call    PushAll
                                // Настроювання регістрів
        mov     ESI,src           // ESI=адреса джерела ключа
        mov     EDI,dst           // EDI=адреса призначення
        mov     EBX,KeyLength   // BX <- адреса кінця ключа
        shl     EBX,2
        add     ebx,esi
        sub     ebx,4
        xor     EAX,EAX         // AX=0
        cld
                                // Перевірка умови завершення циклу
@iloop: cmp         EAX,KeyRepeat  // потрібне число зроблене ?
        jge         @ex            // якщо так, вихід
        mov         ECX,KeyLength  // CX=довжина ключа
        bt          RepeatMask,AX  // CF=черговий біт маски
        jc         @Rev            // CF=1 -> реверс
                                // Повторення елементів ключа в прямому
порядку
        mov         ESI,src         // ESI=зсув ключа
        rep     movsd           // копіюємо ключ
        jmp     @Incr           // на рахунок повтор. ключа
                                // Повторення елементів ключа у зворотному
порядку
@Rev:   mov     ESI,EBX         // SI=адреса кінця ключа
@rl:    movsd                         // слова ключа
        sub           ESI,8           // до попереднього елемента
        loop    @rl             // організація циклу
@Incr:  inc     AX              // лічильник повторів ключа
        jmp     @iloop
@ex:    call    PopAll
end;

{-------------------------
 Шифрування масиву даних у режимі простої заміни
 згідно криптоалгоритму ДСТ 28147-89.
---------------------------
 Зауваження:
 1.Зашифрування або  розшифрування задається передачею
 відповідної  адреси  ключа – порядок  елементів
 у   ключах  зашифрування  й  розшифрування   взаємно
 зворотний.
2.Відповідно  ДО ДСТУ 28147-89  цей модуль може використовуватися
 тільки  для шифрування  ключової інформації
 (і  синхропосилки  для  гаммування).
```

Кафедра_КбПЗ_2022рік

```
}

procedure TGOSTEncryption.simple(src,dst:pointer;noblocks:DWORD);
// Шифрування простою заміною

var l:DWORD;
    pt:TPExtTable;
    pk:TPExtEncKey;

begin
    l:=noblocks;
    pt:=@ExtChTab;
    pk:=@ExtKey;
asm
         call      PushAll
                                 //Початкове завантаження покажчиків
         mov       ebx,pt// BX <- адреса ТЗ
         mov       esi,src       //source address
         mov       edi,dst       // address призначення
                                 // Завантаження блоку даних
@circle: mov       EAX,[esi]     // EAX <- S1
         mov       EDX,[esi+4]   // EDX <- S2
                                 // Виклик простої заміни;
         mov       ecx,32        // CX <- число основних кроків
         push      esi
         mov       esi,pk// ESI <- адреса ключа
         call      Gost386       //     <крок простої заміни>
         pop       esi
                                 // Запис  результату  на  місце
         mov       [edi+4],EAX   // Заносимо результат
         mov       [edi],EDX     //   на його місце
                                 // Організація циклу
         add       edi,8         // корекція адреси призначення
         add       esi,8         //go to next element in the source
         dec       l             // корекція лічильника  б.
         jnz       @circle       // робимо цикл
         call      PopAll
end;
end;

{-------------------------
 Побудова розширеної таблиці замін (1024 байт) з
 таблиці замін (128 байт) алгоритму  ДСТ 28147-89.

---------------------------
}

procedure TGOSTEncryption.Set128Table(p:TP128Table);
var pt:TPExtTable;
begin
    pt:=@ExtChTab;
asm
        call      PushAll
        mov       ESI,p          // ESI->  джерело
        mov       EDI,pt         // EDI->  приймач
        cld                      // Цикл  по  блоках  розширеної  таблиці
замін
        mov       ecx,4          // CX <- число блоків великий ТЗ
@blocks:mov       EBX,ESI        // BX<-адреса  початку  лінії
        add       EBX,10h        //     старших байтів блоку
                                 // Цикл  по  лініях  блоку
        push      eCX            // зберегти  лічильник блоків
        mov       eCX,16         // завантажити  лічильник  ліній
@lines: push      ESI            // збереження показника поточного  блоку
        mov       AH,[EBX]       // AH <- старший    напівбайт
        push      eCX            // зберегти   лічильник ліній
        mov       CL,4           // зрушення напівбайта  на місце
        shl       AH,CL          //     старшого  напівбайта
                                 // Цикл  по  байтах  лінії
```

Кафедра КБПЗ 2022рік

```
        mov     ECX,16          // завантажити лічильник  байтів
@bytes: lodsb                   // завантажити черговий  байт
        or      AL,AH           // додати старший напівбайт
        stosb                   // ...  і записати результат
        loop    @bytes          // цикл   по  байтах   лінії
                                // Перевірка  циклу  по  лініях
        pop     eCX             // відновлюємо лічильник  ліній
        pop     ESI             // відновлюємо вказівник тек. блоку
        inc     EBX             // просунути покажчик байт
        loop    @lines          // цикл  по рядках  таблиці
                                // Перевірка  циклу  по  блоках
        pop     eCX             // відновлюємо лічильник блоків
        add     ESI,20h         // просунути  вказівник блоку
        loop    @blocks         // цикл по блоках
        call    PopAll
end;
end;


end.
```

Кафедра_КбПЗ_2022рік

Файл проекту Zahischeniy_Tachograph_Unit2.pas – перевірка апаратної частини

```
//*************************************************************//
// Testuvalos`a u:  Embarcadero Delphi //
//  OS: WIN10/11                                          //
//                    Testova versiya tachographu                  //
// coded by: Shevchuk I.I.                                 //
//*************************************************************//

unit Unit2;
//Zahischeniy_TachographSysInfoU;
interface
uses
Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms,
Dialogs,ComCtrls, StdCtrls, Buttons, ImgList, Commctrl, ShellAPI, Menus;

type
TForm1 = class(TForm)
PageControl1: TPageControl;
TabSheet2: TTabSheet;
TabSheet1: TTabSheet;
ListView1: TListView;
StatusBar1: TStatusBar;
ListView2: TListView;
ImageList1: TImageList;
PopupMenu1: TPopupMenu;
Details1: TMenuItem;
KillProcess1: TMenuItem;
RefreshList1: TMenuItem;
PopupMenu2: TPopupMenu;
CloseWindow1: TMenuItem;
RefreshList2: TMenuItem;
SpeedButton1: TSpeedButton;
procedure PageControl1Change(Sender: TObject);
procedure FormCreate(Sender: TObject);
procedure ListView1Click(Sender: TObject);
procedure ListView1DblClick(Sender: TObject);
procedure ListView1KeyUp(Sender: TObject; var Key: Word;
Shift: TShiftState);
procedure Details1Click(Sender: TObject);
procedure KillProcess1Click(Sender: TObject);
procedure RefreshList1Click(Sender: TObject);
procedure CloseWindow1Click(Sender: TObject);
procedure RefreshList2Click(Sender: TObject);
procedure SpeedButton1Click(Sender: TObject);
private
{ Private declarations }
public
{ Public declarations }
end;
var
Form1: TForm1;
implementation
{$R *.DFM}
uses
TlHelp32, AdditU;
procedure ListProcesses;
var
hSnapShot: THandle;
lppe: TProcessEntry32;
hIcon: THandle;
Count: Integer;
procedure _FillList;
begin
with Form1.ListView1.Items.Add, lppe do
begin
hIcon:= ExtractIcon(hInstance, lppe.szExeFile, 0);
if hIcon = 0 then
```

Кафедра КБПЗ_2022рік

```
hIcon:= LoadImage(0, IDI_WINLOGO, IMAGE_ICON, LR_DEFAULTSIZE,
LR_DEFAULTSIZE, LR_DEFAULTSIZE or LR_DEFAULTCOLOR or LR_SHARED);
ImageIndex:= ImageList_AddIcon(Form1.ImageList1.Handle, hIcon);
Caption:= ExtractFileName(szExeFile);
SubItems.Add(Format('$%x', [Th32ProcessID]));
SubItems.Add(Format('$%x', [Th32ParentProcessID]));
case pcPriClassBase of
4: SubItems.Add(Format('%d (Idle)', [pcPriClassBase]));
8: SubItems.Add(Format('%d (Norm)', [pcPriClassBase]));
13: SubItems.Add(Format('%d (High)', [pcPriClassBase]));
24: SubItems.Add(Format('%d (Real)', [pcPriClassBase]));
else SubItems.Add(Format('%d', [pcPriClassBase]));
end;
SubItems.Add(Format('%d', [cntThreads]));
SubItems.Add(Format('%d', [cntUsage]));
SubItems.Add(szExeFile);
end;
Inc(Count);
end;
begin
hSnapShot:= CreateToolHelp32Snapshot(TH32CS_SNAPPROCESS, 0);
if hSnapShot <> INVALID_HANDLE_VALUE then
begin
Form1.ListView1.Items.Clear;
lppe.dwSize:= SizeOf(lppe);
Count:= 0;
if Process32First(hSnapShot, lppe) then _FillList;
while Process32Next(hSnapShot, lppe) do _FillList;
Form1.StatusBar1.Panels[1].Text:= 'Total: ' + IntToStr(Count);
CloseHandle(hSnapShot);
end
else MessageBox(Form1.Handle, 'Внутрішня помилка', 'Помилка', MB_OK or
MB_ICONERROR);
end;
procedure ListWindows;
function EnumWindowsProc(hWnd: THandle; lParam: Integer): Boolean; stdcall;
var
Text, PName: string;
hIcon: THandle;
lpdwPID: PDWORD;
i: Integer;
begin
New(lpdwPID);
GetWindowThreadProcessID(hWnd, lpdwPID);
SetLength(Text, 255);
if GetWindowText(hWnd, PChar(Text), 255) <> 0 then
with Form1.ListView2.Items.Add, Form1.ListView1 do
begin
hIcon:= GetClassLong(hWnd, GCL_HICON);
ImageIndex:= ImageList_AddIcon(Form1.ImageList1.Handle, hIcon);
for i:= 0 to Items.Count - 1 do
if Cardinal(StrToInt(Items[i].SubItems[0])) = lpdwPID^ then
PName:= Items[i].Caption;
Caption:= Text;
SubItems.Add(Format('%d', [hWnd]));
SubItems.Add(Format('$%x', [lpdwPID^]));
SubItems.Add(PName);
Dispose(lpdwPID);
end;
Result:= true;
end;
begin
Form1.ListView2.Items.Clear;
EnumWindows(@EnumWindowsProc, 0);
end;

procedure ListModules(OwnerID: Cardinal);
var
hSnapShot: THandle;
```

Кафедра_КбПЗ_2022рік

```
lpme: TModuleEntry32;

procedure _FillList;
begin
with Form2.ListView1.Items.Add, lpme do
begin
Caption:= ExtractFileName(szModule);
SubItems.Add(Format('%d', [modBaseSize]));
SubItems.Add(Format('$%p', [modBaseAddr]));
SubItems.Add(Format('%d', [ProccntUsage]));
SubItems.Add(Format('%d', [GlblcntUsage]));
SubItems.Add(szExePath);
end;
end;

begin
hSnapShot:= CreateToolHelp32Snapshot(TH32CS_SNAPMODULE, OwnerID);
if hSnapShot <> INVALID_HANDLE_VALUE then
begin
Form2.ListView1.Items.Clear;
lpme.dwSize:= SizeOf(lpme);
if Module32First(hSnapShot, lpme) then _FillList;
while Module32Next(hSnapShot, lpme) do _FillList;
CloseHandle(hSnapShot);
end
else MessageBox(Form1.Handle, 'Внутрішня помилка', 'Помилка', MB_OK or
MB_ICONERROR);
end;

procedure ListThreads(OwnerID: Cardinal);
var
hSnapShot: THandle;
lpte: TThreadEntry32;

procedure _FillList;
begin
if lpte.Th32OwnerProcessID =
Cardinal(StrToInt(Form1.ListView1.Selected.SubItems[0])) then
with Form2.ListView2.Items.Add, lpte do
begin
Caption:= Format('$%x', [Th32ThreadID]);
case TpBasePri of
4: SubItems.Add(Format('%d (Idle)', [TpBasePri]));
8: SubItems.Add(Format('%d (Norm)', [TpBasePri]));
13: SubItems.Add(Format('%d (High)', [TpBasePri]));
24: SubItems.Add(Format('%d (Real)', [TpBasePri]));
else SubItems.Add(Format('%d', [TpBasePri]));
end;
case TpDeltaPri of
-15: SubItems.Add(Format('%d (Idle)', [TpDeltaPri]));
-2: SubItems.Add(Format('%d (Lowest)', [TpDeltaPri]));
-1: SubItems.Add(Format('%d (Low)', [TpDeltaPri]));
0: SubItems.Add(Format('%d (Normal)', [TpDeltaPri]));
1: SubItems.Add(Format('%d (High)', [TpDeltaPri]));
2: SubItems.Add(Format('%d (Highest)', [TpDeltaPri]));
15: SubItems.Add(Format('%d (Time Critical)', [TpDeltaPri]));
else SubItems.Add(Format('%d', [TpDeltaPri]));
end;
SubItems.Add(Format('%d', [cntUsage]));
end;
end;

begin
hSnapShot:= CreateToolhelp32Snapshot(TH32CS_SNAPTHREAD, OwnerID);
if hSnapShot <> INVALID_HANDLE_VALUE then
begin
Form2.ListView2.Items.Clear;
lpte.dwSize:= SizeOf(lpte);
if Thread32First(hSnapShot, lpte) then _FillList;
```

Кафедра_КбПЗ_2022рік

```
while Thread32Next(hSnapShot, lpte) do _FillList;
CloseHandle(hSnapShot);
end
else MessageBox(Form1.Handle, 'Внутрішня помилка', 'Помилка', MB_OK or
MB_ICONERROR);
end;

procedure ListHeaps(OwnerID: Cardinal);
var
hSnapShot: THandle;
lphl: THeapList32;
lphe: THeapEntry32;

procedure _FillList;
begin
if Heap32First(lphe, lphl.Th32ProcessID, lphl.Th32HeapID) then
repeat
with Form2.ListView3.Items.Add, lphe do
begin
Caption:= Format('$%x', [Th32HeapID]);
SubItems.Add(Format('%d', [dwBlockSize]));
SubItems.Add(Format('$%x', [dwAddress]));
case dwFlags of
LF32_FIXED: SubItems.Add('Незмінний');
LF32_FREE: SubItems.Add('Вільний');
LF32_MOVEABLE: SubItems.Add('Змінний')
else SubItems.Add('Невідомий');
end;
end
until not Heap32Next(lphe);
end;

begin
try
Form1.Caption:= 'Системна інформація: ЗАПИТ ПО ЗАХИЩЕНІЙ ЧІП-КАРТІ У ПРОЦЕСІ
ОБРОБКИ... ЗАЧЕКАЙТЕ';
hSnapShot:= CreateToolhelp32Snapshot(TH32CS_SNAPHEAPLIST, OwnerID);
if hSnapShot <> INVALID_HANDLE_VALUE then
begin
Form2.ListView3.Items.Clear;
lphl.dwSize:= SizeOf(lphl);
lphe.dwSize:= SizeOf(lphe);
if Heap32ListFirst(hSnapShot, lphl) then _FillList;
while Heap32ListNext(hSnapShot, lphl) do _FillList;
CloseHandle(hSnapShot);
end
else MessageBox(Form1.Handle, 'Внутрішня помилка', 'Помилка', MB_OK or
MB_ICONERROR);
finally
Form1.Caption:= 'Системна інформація';
end;
end;

procedure ListThreadWindows;
var
i: Integer;
function EnumThreadWindowsProc(hWnd: THandle; lParam: Integer): Boolean;
stdcall;
var
Text: string;
hIcon: THandle;
begin
SetLength(Text, 255);
if GetWindowText(hWnd, PChar(Text), 255) <> 0 then
with Form2.ListView4.Items.Add do
begin
hIcon:= GetClassLong(hWnd, GCL_HICON);
ImageIndex:= ImageList_AddIcon(Form1.ImageList1.Handle, hIcon);
Caption:= Text;
```

Кафедра_КБПЗ_2022рік

```
SubItems.Add(Format('%d', [hWnd]));
end;
Result:= true;
end;

begin
Form2.ListView4.Items.Clear;
for i:= 0 to Form2.ListView2.Items.Count - 1 do
EnumThreadWindows(Cardinal(StrToInt(Form2.ListView2.Items[i].Caption)),@EnumThr
eadWindowsProc, 0);
end;

{-----------------------------------------------------------------------}
procedure TForm1.PageControl1Change(Sender: TObject);
begin
ListProcesses;
end;

procedure TForm1.FormCreate(Sender: TObject);
begin
ListProcesses;
ListWindows;
end;

procedure TForm1.ListView1Click(Sender: TObject);
begin
with (Sender as TListView) do
if Selected <> nil then
StatusBar1.Panels[0].Text:= Selected.SubItems[5] else
StatusBar1.Panels[0].Text:= '';
end;

procedure TForm1.ListView1DblClick(Sender: TObject);
begin
Details1Click(self);
end;

procedure TForm1.ListView1KeyUp(Sender: TObject; var Key: Word;
Shift: TShiftState);
begin
if flag = true then flag:= false
else
begin
if Key = 13 then ListView1DblClick(ListView1);
if Key <> 0 then ListView1Click(ListView1);
end;
end;

procedure TForm1.Details1Click(Sender: TObject);
begin
with ListView1, Form2 do
if Selected <> nil then
begin
ListModules(StrToInt(Selected.SubItems[0]));
ListThreads(StrToInt(Selected.SubItems[0]));
ListHeaps(StrToInt(Selected.SubItems[0]));
ListThreadWindows;
Caption:= 'Більш детально  ' + Selected.Caption;
Show;
end
'Information', MB_OK or MB_ICONINFORMATION);
end;

procedure TForm1.KillProcess1Click(Sender: TObject);
var
hProcess: THandle;
S: string;
ID: Cardinal;
begin
```

Кафедра_КбПЗ_2022рік

```
if ListView1.Selected <> nil then
begin
S:= ListView1.Selected.Caption;
ID:= StrToInt(ListView1.Selected.SubItems[0]);
hProcess:= OpenProcess(PROCESS_ALL_ACCESS, false, ID);
if hProcess <> INVALID_HANDLE_VALUE then
begin
if not TerminateProcess(hProcess, 0) then
MessageBox(0, PChar('Неможливо завершити процес: ' + S),
'Помилка', MB_ICONWARNING or MB_OK);
CloseHandle(hProcess);
Sleep(500);
ListProcesses;
end;
end
MB_OK or MB_ICONINFORMATION);
end;

procedure TForm1.RefreshList1Click(Sender: TObject);
begin
ListProcesses;
end;

procedure TForm1.CloseWindow1Click(Sender: TObject);
begin
if ListView2.Selected <> nil then
begin
PostMessage(StrToInt(ListView2.Selected.SubItems[0]), WM_CLOSE, 0, 0);
Sleep(500);
ListWindows;
end
end;

procedure TForm1.RefreshList2Click(Sender: TObject);
begin
ListWindows;
end;

end.
```

Кафедра_КбПЗ_2022рік

Файл проекту Zahischeniy_Tachograph_Unit3.pas – Організація часових проміжків, системні функції

```
//*************************************************************//
// Testuvalos`a u:  Embarcadero Delphi //
//  OS: WIN10/11                                          //
//                    Testova versiya tachographu                  //
// coded by: Shevchuk I.I.                                  //
//*************************************************************//
 unit Unit3;
 //FormUnit;
 interface
 uses
 Windows, Graphics, ExtCtrls, Controls, StdCtrls, Buttons, Tabs,
 ComCtrls, Classes, SysUtils, Forms, dialogs,
 Zahischeniy_TachographUnit;
 type
 TMainForm = class(TForm)
 pnlMain: TPanel;
 pnlBottom: TPanel;
 pc: TPageControl;
 tsAbout: TTabSheet;
 tsTraffic: TTabSheet;
 ExitButton: TButton;
 TrafficTabs: TTabSet;
 GroupBox: TGroupBox;
 ledAdapterDescription: TLabeledEdit;
 UnFreezeButton: TBitBtn;
 FreezeButton: TBitBtn;
 ClearCountersButton: TBitBtn;
 ledMACAddress: TLabeledEdit;
 gbIN: TGroupBox;
 ledOctInSec: TLabeledEdit;
 ledAvgINSec: TLabeledEdit;
 ledPeakINSec: TLabeledEdit;
 ledTotalIN: TLabeledEdit;
 gbOUT: TGroupBox;
 ledOctOUTSec: TLabeledEdit;
 ledAvgOUTSec: TLabeledEdit;
 ledPeakOUTSec: TLabeledEdit;
 ledTotalOUT: TLabeledEdit;
 Timer: TTimer;
 gbTime: TGroupBox;
 ledStartedAt: TLabeledEdit;
 ledActiveFor: TLabeledEdit;
 RemoveInactiveButton: TBitBtn;
 StatusText: TStaticText;
 cbOnTop: TCheckBox;
 Panel3: TPanel;
 ProductName: TLabel;
 lblURL: TLabel;
 Label3: TLabel;
 ProgramIcon: TImage;
 StaticText1: TStaticText;
 ledSpeed: TLabeledEdit;
 procedure TimerTimer(Sender: TObject);
 procedure ClearCountersButtonClick(Sender: TObject);
 procedure cbOnTopClick(Sender: TObject);
 procedure FormDestroy(Sender: TObject);
 procedure TrafficTabsChange(Sender: TObject; NewTab: Integer;
 var AllowChange: Boolean);
 procedure ExitButtonClick(Sender: TObject);
 procedure FormCreate(Sender: TObject);
 procedure FreezeButtonClick(Sender: TObject);
 procedure UnFreezeButtonClick(Sender: TObject);
 procedure RemoveInactiveButtonClick(Sender: TObject);
 procedure lblURLClick(Sender: TObject);
 procedure StaticText1Click(Sender: TObject);
```

Кафедра_КбПЗ_2022рік

```
procedure pcChange(Sender: TObject);
procedure ledAdapterDescriptionChange(Sender: TObject);
private
procedure HandleNewAdapter(ATraffic : TZahischeniy_Tachograph_Traffic);
procedure HandleFreeze(ATraffic : TZahischeniy_Tachograph_Traffic);
procedure HandleUnFreeze(ATraffic : TZahischeniy_Tachograph_Traffic);
function LocateTraffic(AdapterIndex : DWord) : TZahischeniy_Tachograph_Traffic;
procedure ProcessMIBData;
procedure ClearDisplay;
procedure RefreshDisplay;
public
{ Public declarations }
end;
var
MainForm: TMainForm;
ActiveTraffic : TZahischeniy_Tachograph_Traffic;

implementation
{$R *.dfm}
procedure TMainForm.ClearDisplay;
var
j:integer;
begin
TrafficTabs.Tabs.Clear;
StatusText.Caption:='';
for j:= 0 to GroupBox.ControlCount-1 do
begin
if GroupBox.Controls[j] is TCustomEdit then
TCustomEdit(GroupBox.Controls[j]).Text := '';
end;
end;

procedure TMainForm.TimerTimer(Sender: TObject);
begin
Timer.Enabled := False;
ProcessMIBData;
Timer.Enabled := True;
end; (*TimerTimer*)

procedure TMainForm.ClearCountersButtonClick(Sender: TObject);
begin
ActiveTraffic.Reset;
RefreshDisplay;
end;

procedure TMainForm.cbOnTopClick(Sender: TObject);
begin
if cbOnTop.Checked = true then
FormStyle := fsSTAYONTOP
else
FormStyle := fsNORMAL;
end;

procedure TMainForm.FormDestroy(Sender: TObject);
var
i : integer;
begin
Timer.OnTimer := nil;
ActiveTraffic := nil;
for i:= 0 to -1 + TrafficTabs.Tabs.Count do
TrafficTabs.Tabs.Objects[i].Free;
end;
procedure TMainForm.TrafficTabsChange(Sender: TObject; NewTab: Integer; var
AllowChange: Boolean);
begin
if NewTab = -1 then
ActiveTraffic := nil
else
```

Кафедра_КБПЗ_2022рік

```
ActiveTraffic :=
TZahischeniy_Tachograph_Traffic(TrafficTabs.Tabs.Objects[NewTab]);
RefreshDisplay;
end;

procedure TMainForm.ExitButtonClick(Sender: TObject);
begin
Close;
end;

procedure TMainForm.FormCreate(Sender: TObject);
begin
Timer.Interval := 1000;
ClearDisplay;
ActiveTraffic := nil;
pcChange(Sender);
Timer.Enabled := True;
end;


procedure TMainForm.RefreshDisplay;
begin
if not Assigned(ActiveTraffic) then
begin
ClearDisplay;
Exit;
end;

with ActiveTraffic do
begin

FreezeButton.Visible := Connected;
UnFreezeButton.Visible := Connected;
ClearCountersButton.Visible := Connected;
RemoveInactiveButton.Visible := not Connected;

FreezeButton.Enabled := Running;
UnFreezeButton.Enabled := not Running;

ledAdapterDescription.Text := Description;
ledMACAddress.Text := MAC;

ledSpeed.Text := BitsToFriendlyString(Speed);

ledOctInSec.Text := BytesToFriendlyString(InPerSec);
ledPeakInSec.Text := BytesToFriendlyString(PeakInPerSec);
ledAvgINSec.Text := BytesToFriendlyString(AverageInPerSec);
ledTotalIN.Text := BytesToFriendlyString(InTotal);

ledOctOUTSec.Text := BytesToFriendlyString(OutPerSec);
ledPeakOUTSec.Text := BytesToFriendlyString(PeakOutPerSec);
ledAvgOUTSec.Text := BytesToFriendlyString(AverageOutPerSec);
ledTotalOUT.Text := BytesToFriendlyString(OutTotal);

self.ledStartedAt.Text := DateTimeToStr(StartedAt);
self.ledActiveFor.Text := FriendlyRunningTime;

StatusText.Caption := GetStatus;
end;
end; (*Оновлення зображення на дисплеї*)

procedure TMainForm.ProcessMIBData;
var
MibArr : IpHlpAPI.TMIBIfArray;
i : integer;
ATraffic : TZahischeniy_Tachograph_Traffic;
begin
Get_IfTableMIB(MibArr);
```

Кафедра_КОПЗ_2022рік

```
for i:= 0 to -1 + TrafficTabs.Tabs.Count do
begin
ATraffic := TZahischeniy_Tachograph_Traffic(TrafficTabs.Tabs.Objects[i]);
if ATraffic.Connected then ATraffic.Found := False;
end;
ATraffic := nil;

//процес роботи
if Length(MibArr) > 0 then
begin
for i := Low(MIBArr) to High(MIBArr) do
begin
ATraffic := LocateTraffic(MIBArr[i].dwIndex);
if Assigned(ATraffic) then
begin
ATraffic.NewCycle(MIBArr[i].dwInOctets, MIBArr[i].dwOutOctets,
MIBArr[i].dwSpeed);
end
else
begin
ATraffic := TZahischeniy_Tachograph_Traffic.Create(MIBArr[i],
HandleNewAdapter);
ATraffic.Found := True;
ATraffic.OnFreeze := HandleFreeze;
ATraffic.OnUnFreeze := HandleUnFreeze;
end;
end;
end;

for i:= 0 to -1 + TrafficTabs.Tabs.Count do
if NOT TZahischeniy_Tachograph_Traffic(TrafficTabs.Tabs.Objects[i]).Found then
TZahischeniy_Tachograph_Traffic(TrafficTabs.Tabs.Objects[i]).MarkDisconnected;

RefreshDisplay;
end; (*Обробка даних з Бази Даних*)

function TMainForm.LocateTraffic(AdapterIndex : DWord):
TZahischeniy_Tachograph_Traffic;
var
j : cardinal;
ATraffic : TZahischeniy_Tachograph_Traffic;
begin
Result := nil;
if TrafficTabs.Tabs.Count = 0 then Exit;

for j:= 0 to -1 + TrafficTabs.Tabs.Count do
begin
ATraffic := TZahischeniy_Tachograph_Traffic(TrafficTabs.Tabs.Objects[j]);
if ATraffic.InterfaceIndex = AdapterIndex then
begin
Result := ATraffic;
Result.Found := True;
Break;
end;
end;
end;

procedure TMainForm.HandleNewAdapter(ATraffic:
TZahischeniy_Tachograph_Traffic);
begin
TrafficTabs.Tabs.AddObject(ATraffic.IP, ATraffic);
TrafficTabs.TabIndex := -1 + TrafficTabs.Tabs.Count;
end; (*Заголовок нового пристрою*)

procedure TMainForm.FreezeButtonClick(Sender: TObject);
begin
ActiveTraffic.Freeze;
end;
```

Кафедра_КБПЗ_2022рік

```
procedure TMainForm.UnFreezeButtonClick(Sender: TObject);
begin
ActiveTraffic.UnFreeze;
end;

procedure TMainForm.HandleFreeze(ATraffic: TZahischeniy_Tachograph_Traffic);
begin
self.FreezeButton.Enabled := ATraffic.Running;
self.UnFreezeButton.Enabled := not ATraffic.Running;
end;

procedure TMainForm.HandleUnFreeze(ATraffic: TZahischeniy_Tachograph_Traffic);
begin
self.FreezeButton.Enabled := ATraffic.Running;
self.UnFreezeButton.Enabled := not ATraffic.Running;
end;

procedure TMainForm.RemoveInactiveButtonClick(Sender: TObject);
begin
If not ActiveTraffic.Connected then
begin
ActiveTraffic.Free;
ActiveTraffic := nil;
TrafficTabs.Tabs.Delete(TrafficTabs.TabIndex);
TrafficTabs.SelectNext(False);
end;

RefreshDisplay;
end;

procedure TMainForm.pcChange(Sender: TObject);
begin
pnlBottom.Visible := pc.ActivePage = tsTraffic;
end;

procedure TMainForm.ledAdapterDescriptionChange(Sender: TObject);
begin
ledAdapterDescription.Hint := ledAdapterDescription.Text;
ledAdapterDescription.ShowHint := Canvas.TextWidth(ledAdapterDescription.Text)
> ledAdapterDescription.ClientWidth;
end;

end.
```

Кафедра_КбПЗ_2022рік

Файл проекту Zahischeniy_Tachograph_Unit4.pas – заголовки внутрішніх зміних бібліотеки IPHLPAPI

```
//*************************************************************//
// Testuvalos`a u:  Embarcadero Delphi //
//  OS: WIN10/11                                          //
//                    Testova versiya tachographu                  //
// coded by: Shevchuk I.I.                                  //
//*************************************************************//

unit Unit4
//IPHLPAPI;

interface
uses
  Windows, winsock;
const
  VERSION       = '1.3';
const
  ANY_SIZE      = 1;
  MAX_ADAPZAHISCHENIY_TACHOGRAPH_DESCRIPTION_LENGTH = 128; // arb.
  MAX_ADAPZAHISCHENIY_TACHOGRAPH_NAME_LENGTH = 256; // arb.
  MAX_ADAPZAHISCHENIY_TACHOGRAPH_ADDRESS_LENGTH = 8; // arb.
  DEFAULT_MINIMUM_ENTITIES = 32; // arb.
  MAX_HOSTNAME_LEN = 128; // arb.
  MAX_DOMAIN_NAME_LEN = 128; // arb.
  MAX_SCOPE_ID_LEN = 256; // arb.

  BROADCAST_NODETYPE = 1;
  PEER_TO_PEER_NODETYPE = 2;
  MIXED_NODETYPE = 4;
  HYBRID_NODETYPE = 8;
ATypes  : array[0..8] of string[20] =
( 'Невідомий', 'BROADCAST', 'PEER_TO_PEER', '', 'MIXED', '', '', '', 'HYBRID');
  IF_OTHER_ADAPTERTYPE = 0;
  IF_ETHERNET_ADAPTERTYPE = 1;
  IF_TOKEN_RING_ADAPTERTYPE = 2;
  IF_FDDI_ADAPTERTYPE = 3;
  IF_PPP_ADAPTERTYPE = 4;
  IF_LOOPBACK_ADAPTERTYPE = 5;
  IF_SLIP_ADAPTERTYPE = 6;
 //
  AdaptTypes    : array[0..6] of string[10] =
( 'other','ethernet','tokenring','FDDI','PPP','loopback','SLIP' );
  MAX_INTERFACE_NAME_LEN = 256; { mrapi.h }
  MAXLEN_PHYSADDR = 8; { iprtrmib.h }
  MAXLEN_IFDESCR = 256; { --"---     }
//-------------------------------------------------------------
type
  TMacAddress = array[1..MAX_ADAPZAHISCHENIY_TACHOGRAPH_ADDRESS_LENGTH] of byte;

  PTIP_ADDRESS_STRING = ^TIP_ADDRESS_STRING;
  TIP_ADDRESS_STRING = array[0..15] of char;
  PTIP_ADDR_STRING = ^TIP_ADDR_STRING;
  TIP_ADDR_STRING = packed record // використовується для списку з'єднань
    Next: PTIP_ADDR_STRING;
    IpAddress: TIP_ADDRESS_STRING;
    IpMask: TIP_ADDRESS_STRING;
    Context: DWORD;
  end;

  PTFixedInfo = ^TFixedInfo;
  TFixedInfo = packed record
    HostName: array[0..MAX_HOSTNAME_LEN + 4] of char;
    DomainName: array[0..MAX_DOMAIN_NAME_LEN + 4] of char;
    CurrentDNSServer: PTIP_ADDR_STRING;
    DNSServerList: TIP_ADDR_STRING;
    NodeType: UINT;
    ScopeID: array[0..MAX_SCOPE_ID_LEN + 4] of char;
```

Кафедра_КбПЗ_2022рік

```
    EnableRouting: UINT;
    EnableProxy: UINT;
    EnableDNS: UINT;
  end;

  PTMibIfRow = ^TMibIfRow;
  TMibIfRow = packed record
    wszName: array[1..MAX_INTERFACE_NAME_LEN] of WCHAR;
    dwIndex: DWORD;
    dwType: DWORD;
    dwMTU: DWORD;
    dwSpeed: DWORD;
    dwPhysAddrLen: DWORD;
    bPhysAddr: array[1..MAXLEN_PHYSADDR] of byte;
    dwAdminStatus: DWORD;
    dwOperStatus: DWORD;
    dwLastChange: DWORD;
    dwInOctets: DWORD;
    dwInUcastPkts: DWORD;
    dwInNUCastPkts: DWORD;
    dwInDiscards: DWORD;
    dwInErrors: DWORD;
    dwInUnknownProtos: DWORD;
    dwOutOctets: DWORD;
    dwOutUCastPkts: DWORD;
    dwOutNUCastPkts: DWORD;
    dwOutDiscards: DWORD;
    dwOutErrors: DWORD;
    dwOutQLen: DWORD;
    dwDescrLen: DWORD;
    bDescr: array[1..MAXLEN_IFDESCR] of char; //байт;
  end;

  TMIBIfArray = array of TMIBIFRow;

  PTMibIfTable = ^TMIBIfTable;
  TMibIfTable = packed record
    dwNumEntries: DWORD;
    Table: array[0..ANY_SIZE - 1] of TMibIfRow;
  end;

  TTIME_T = array[1..325] of byte;

  PTIP_ADAPZAHISCHENIY_TACHOGRAPH_INFO = ^TIP_ADAPZAHISCHENIY_TACHOGRAPH_INFO;
  TIP_ADAPZAHISCHENIY_TACHOGRAPH_INFO = packed record
    Next: PTIP_ADAPZAHISCHENIY_TACHOGRAPH_INFO;
    ComboIndex: DWORD;
    AdapterName: array[1..MAX_ADAPZAHISCHENIY_TACHOGRAPH_NAME_LENGTH + 4] of
char;
    Description: array[1..MAX_ADAPZAHISCHENIY_TACHOGRAPH_DESCRIPTION_LENGTH + 4]
of char;
    AddressLength: UINT;
    Address: array[1..MAX_ADAPZAHISCHENIY_TACHOGRAPH_ADDRESS_LENGTH] of byte;
    Index: DWORD;
    aType: UINT;
    DHCPEnabled: UINT;
    CurrentIPAddress: PTIP_ADDR_STRING;
    IPAddressList: TIP_ADDR_STRING;
    GatewayList: TIP_ADDR_STRING;
    DHCPServer: TIP_ADDR_STRING;
    HaveWINS: BOOL;
    PrimaryWINSServer: TIP_ADDR_STRING;
    SecondaryWINSServer: TIP_ADDR_STRING;
    LeaseObtained: TTIME_T;
    LeaseExpires: TTIME_T;
  end;

  PTMibTCPRow = ^TMibTCPRow;
  TMibTCPRow = packed record
```

Кафедра_КбПЗ_2022рік

```
    dwState: DWORD;
    dwLocalAddr: DWORD;
    dwLocalPort: DWORD;
    dwRemoteAddr: DWORD;
    dwRemotePort: DWORD;
  end;
  //
  PTMibTCPTable = ^TMibTCPTable;
  TMibTCPTable = packed record
    dwNumEntries: DWORD;
    Table: array[0..0] of TMibTCPRow;
  end;
  //
  PTMibTCPStats = ^TMibTCPStats;
  TMibTCPStats = packed record
    dwRTOAlgorithm: DWORD;
    dwRTOMin: DWORD;
    dwRTOMax: DWORD;
    dwMaxConn: DWORD;
    dwActiveOpens: DWORD;
    dwPassiveOpens: DWORD;
    dwAttemptFails: DWORD;
    dwEstabResets: DWORD;
    dwCurrEstab: DWORD;
    dwInSegs: DWORD;
    dwOutSegs: DWORD;
    dwRetransSegs: DWORD;
    dwInErrs: DWORD;
    dwOutRsts: DWORD;
    dwNumConns: DWORD;
  end;

  PTMibUDPRow = ^TMibUDPRow;
  TMibUDPRow = packed record
    dwLocalAddr: DWORD;
    dwLocalPort: DWORD;
  end;
 //
  PTMibUDPTable = ^TMIBUDPTable;
  TMIBUDPTable = packed record
    dwNumEntries: DWORD;
    UDPTable: array[0..ANY_SIZE - 1] of TMibUDPRow;
  end;
 //
  PTMibUdpStats = ^TMIBUdpStats;
  TMIBUdpStats = packed record
    dwInDatagrams: DWORD;
    dwNoPorts: DWORD;
    dwInErrors: DWORD;
    dwOutDatagrams: DWORD;
    dwNumAddrs: DWORD;
  end;

  PTMibIPNetRow = ^TMibIPNetRow;
  TMibIPNetRow = packed record
    dwIndex: DWord;
    dwPhysAddrLen: DWord;
    bPhysAddr: TMACAddress;
    dwAddr: DWord;
    dwType: DWord;
  end;

  PTMibIPNetTable = ^TMibIPNetTable;
  TMibIPNetTable = packed record
    dwNumEntries: DWORD;
    Table: array[0..ANY_SIZE - 1] of TMibIPNetRow;
  end;

  PTMibIPStats = ^TMibIPStats;
```

Кафедра_КбПЗ_2022рік

```
TMibIPStats = packed record
  dwForwarding: DWORD;
  dwDefaultTTL: DWORD;
  dwInReceives: DWORD;
  dwInHdrErrors: DWORD;
  dwInAddrErrors: DWORD;
  dwForwDatagrams: DWORD;
  dwInUnknownProtos: DWORD;
  dwInDiscards: DWORD;
  dwInDelivers: DWORD;
  dwOutRequests: DWORD;
  dwRoutingDiscards: DWORD;
  dwOutDiscards: DWORD;
  dwOutNoRoutes: DWORD;
  dwReasmTimeOut: DWORD;
  dwReasmReqds: DWORD;
  dwReasmOKs: DWORD;
  dwReasmFails: DWORD;
  dwFragOKs: DWORD;
  dwFragFails: DWORD;
  dwFragCreates: DWORD;
  dwNumIf: DWORD;
  dwNumAddr: DWORD;
  dwNumRoutes: DWORD;
end;
//
PTMibIPAddrRow = ^TMibIPAddrRow;
TMibIPAddrRow = packed record
  dwAddr: DWORD;
  dwIndex: DWORD;
  dwMask: DWORD;
  dwBCastAddr: DWORD;
  dwReasmSize: DWORD;
  Unused1,
    Unused2: WORD;
end;

TMibIPAddrArray = array of TMIBIPAddrRow;

PTMibIPAddrTable = ^TMibIPAddrTable;
TMibIPAddrTable = packed record
  dwNumEntries: DWORD;
  Table: array[0..ANY_SIZE - 1] of TMibIPAddrRow;
end;

PTMibIPForwardRow = ^TMibIPForwardRow;
TMibIPForwardRow = packed record
  dwForwardDest: DWORD;
  dwForwardMask: DWORD;
  dwForwardPolicy: DWORD;
  dwForwardNextHop: DWORD;
  dwForwardIFIndex: DWORD;
  dwForwardType: DWORD;
  dwForwardProto: DWORD;
  dwForwardAge: DWORD;
  dwForwardNextHopAS: DWORD;
  dwForwardMetric1: DWORD;
  dwForwardMetric2: DWORD;
  dwForwardMetric3: DWORD;
  dwForwardMetric4: DWORD;
  dwForwardMetric5: DWORD;
end;
//
PTMibIPForwardTable = ^TMibIPForwardTable;
TMibIPForwardTable = packed record
  dwNumEntries: DWORD;
  Table: array[0..ANY_SIZE - 1] of TMibIPForwardRow;
end;
```

Кафедра_КбПЗ_2022рік

```
  PTMibICMPStats = ^TMibICMPStats;
  TMibICMPStats = packed record
    dwMsgs: DWORD;
    dwErrors: DWORD;
    dwDestUnreachs: DWORD;
    dwTimeEcxcds: DWORD;
    dwParmProbs: DWORD;
    dwSrcQuenchs: DWORD;
    dwRedirects: DWORD;
    dwEchos: DWORD;
    dwEchoReps: DWORD;
    dwTimeStamps: DWORD;
    dwTimeStampReps: DWORD;
    dwAddrMasks: DWORD;
    dwAddrReps: DWORD;
  end;

  PTMibICMPInfo = ^TMibICMPInfo;
  TMibICMPInfo = packed record
    InStats: TMibICMPStats;
    OutStats: TMibICMPStats;
  end;
//------------------+----------------------------------------
function GetAdaptersInfo( pAdapterInfo: PTIP_ADAPZAHISCHENIY_TACHOGRAPH_INFO;
  pOutBufLen: PULONG ): DWORD;
stdcall; external 'IPHLPAPI.DLL';

function GetNetworkParams( FixedInfo: PTFixedInfo; pOutPutLen: PULONG ): DWORD;
stdcall; external 'IPHLPAPI.DLL';

function GetTcpTable( pTCPTable: PTMibTCPTable; pDWSize: PDWORD;
  bOrder: BOOL ): DWORD;
stdcall; external 'IPHLPAPI.DLL';

function GetTcpStatistics( pStats: PTMibTCPStats ): DWORD;
stdcall; external 'IPHLPAPI.DLL';

function GetUdpTable( pUdpTable: PTMibUDPTable; pDWSize: PDWORD;
  bOrder: BOOL ): DWORD;
stdcall; external 'IPHLPAPI.DLL';

function GetUdpStatistics( pStats: PTMibUdpStats ): DWORD;
stdcall; external 'IPHLPAPI.DLL';

function GetIpStatistics( pStats: PTMibIPStats ): DWORD;
stdcall; external 'IPHLPAPI.DLL';

function GetIpNetTable( pIpNetTable: PTMibIPNetTable;
  pdwSize: PULONG;
  bOrder: BOOL ): DWORD;
stdcall; external 'IPHLPAPI.DLL';

function GetIpAddrTable( pIpAddrTable: PTMibIPAddrTable;
  pdwSize: PULONG;
  bOrder: BOOL ): DWORD;
stdcall; external 'IPHLPAPI.DLL';

function GetIpForwardTable( pIPForwardTable: PTMibIPForwardTable;
  pdwSize: PULONG;
  bOrder: BOOL ): DWORD;
stdCall; external 'IPHLPAPI.DLL';

function GetIcmpStatistics( pStats: PTMibICMPInfo ): DWORD;
stdCall; external 'IPHLPAPI.DLL';

function GetRTTAndHopCount( DestIPAddress: DWORD; HopCount: PULONG;
  MaxHops: ULONG; RTT: PULONG ): BOOL;
stdCall; external 'IPHLPAPI.DLL';
```

Кафедра_КОПЗ_2022рік

```
function GetIfTable( pIfTable: PTMibIfTable; pdwSize: PULONG;
  bOrder: boolean ): DWORD;
stdCall; external 'IPHLPAPI.DLL';

function GetIfEntry( pIfRow: PTMibIfRow ): DWORD;
stdCall; external 'IPHLPAPI.DLL';

implementation

end.
```

Кафедра_КбПЗ_2022рік