Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Прикладних інформаційних технологій та електроінженерії
(повна назва факультету)
Комп'ютерно-інтегрованих технологій
(повна назва кафедри)

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього рівня

магістр
(назва освітнього ступеня)

на тему: Розробка та дослідження автоматизованого комплексу управління потоком товарів електронними реєстраторами розрахункових операцій.

Виконав: студент ***6*** курсу, групи ***КТм-61***
спеціальності *151 – Автоматизація та комп'ютерно-інтегровані технології*
(шифр і назва спеціальності)

(підпис) ***Новіков О.В.*** (прізвище та ініціали)

Керівник (підпис) ***Митник М.М.*** (прізвище та ініціали)

Нормоконтроль (підпис) ***Левицький В.В.*** (прізвище та ініціали)

Завідувач кафедри (підпис) ***Микитишин А.Г.*** (прізвище та ініціали)

Рецензент (підпис) ***Дмитрів О.Р.*** (прізвище та ініціали)

Тернопіль
2022

Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет прикладних інформаційних технологій та електроінженерії
(повна назва факультету)

Кафедра комп'ютерно-інтегрованих технологій
(повна назва кафедри)

ЗАТВЕРДЖУЮ
Завідувач кафедри
Микитишин А.Г.
(підпис) (прізвище та ініціали)
« __ » _____ 20*22* р.

З А В Д А Н Н Я

НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього рівня **магістр**
(назва освітнього ступеня)

за спеціальністю 151 Автоматизація та комп'ютерно-інтегровані технології
(шифр і назва спеціальності)

студенту *Новікову Олегу Валерійовичу*
(прізвище, ім'я, по батькові)

1. Тема роботи Розробка та дослідження автоматизованого комплексу управління потоком товарів електронними реєстраторами розрахункових операцій

Керівник роботи ***Митник Микола Мирославович, к.т.н., доцент***
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Затверджені наказом ректора від «*30*» *вересня* 20*22* року № *4/7-815*

2. Термін подання студентом завершеної роботи *21.12.2022 р.*

3. Вихідні дані до роботи *Технічна та експлуатаційна документація обладнання для розрахункових операцій.*

4. Зміст роботи (перелік питань, які потрібно розробити)

Вступ, 1. Аналітична частина, 1.1 Аналіз відомих програмних додатків та комплексів для вирішення вибраного напрямку розробки, 1.2 Огляд відомих технічних рішень з питання обліку реалізації товарів та послуг, 2. Технологічна частина, 2.1 Характеристика обладнання для здійснення розрахункових операцій, 2.2 Принципи роботи з електронним реєстратором розрахункових операцій типу MINI-600ME, 2.3 Опис протоколу взаємодії ЕРРО типу MINI-600 з ПК, 3 Конструкторська частина, 3.1 Підключення електронного реєстратора розрахункових операцій до ПК, 3.2 Налаштування інтерфейсу RS-232 для взаємодії ПК з ЕРРО, 3.3 Розробка термінального касового сервера, 4 Науково-дослідницька, 4.1 Дослідження багатозадачності в термінальному сервері, 5. Спеціальна частина, 5.1 Особливості використання мови С++, 5.2 Ефективність і структура С++, 5.3 Реалізація моделі об'єктно-орієнтованого програмування в С++, 5.4 Вибір компілятора для реалізації проекту, 5.5 Розробка програмного забезпечення для роботи з штрих-кодами, 5.6 Робочі файли термінального сервера, 6 Охорона праці та безпека в надзвичайних ситуаціях, 6.1 Організація охорони праці при роботі з системою управління, 6.2 Електробезпека, 6.3 Розрахунок заземлення, основні висновки дипломної роботи, Бібліографія

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Охорона праці	*Тотосько О.В.*		
Безпека в надзвичайних ситуаціях	*Клепчик В.М.*		

7. Дата видачі завдання *30.09.2022 р.*

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Термін виконання етапів роботи	**Примітка**
1.	*Аналіз постановки завдання і огляд методів*	*05.10.2022*	
2.	*Порівняльна оцінка обладнання*	*07.10.2022*	
3.	*Вибір структурної схеми системи керування*	*28.10.2022*	
4.	*Вибір та розрахунок блоків живлення*	*10.11.2022*	
5.	*Розробка програмного забезпечення*	*19.11.2022*	
6.	*Безпека життєдіяльності*	*12.12.2022*	
7.	*Охорона праці*	*14.12.2022*	
8.	*Оформлення графічної частини*	*17.12.2022*	
9.	*Захист кваліфікаційної роботи*	*21.12.2022*	

Студент _____ ***Новіков О.В.***
(підпис) (прізвище та ініціали)

Керівник роботи _____ ***Митник М.М.***
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Дипломна робота складається з пояснювальної записки та графічної частини (ілюстративний матеріал – слайди).

Об’єм графічної частини дипломної роботи становить 7 слайдів.

Об’єм пояснювальної записки складає 66 друкованих сторінок формату А4 (210×297), об’єм додатків – 0друкованих сторінок формату А4.

Дипломна робота складається з восьми розділів, в яких нараховується 10 рисунків та 5 таблиць з даними.

В роботі використано 9 літературних джерел.

Запропоновано варіант автоматизованого комплексу, розроблений на базі термінального касового серверу. Вдосконалення існуючої системи автоматизації направлене на покращення якості обслуговування, збільшення обсягу реалізації товарів та економії трудових ресурсів. Кінцевою метою дослідження та вдосконалення системи автоматизації є отримання додаткового прибутку торгівельним підприємством.

В рамках даної випускної кваліфікаційної роботи були отримані наступні результати:

- Розроблено архітектуру системи.
- Розробка алгоритму роботи програми.
- Отримання даних з реєстраторів виділено в базу даних.

  Реалізована система для інтеграції алгоритмів аналізу.
- Проведено тестування.

Ключові слова: АЛГОРИТМ, РЕЄСТРАТОР, АДАПТЕР, КОМУТАТОР, ІНФОРМАЦІЙНА СИСТЕМА.

ЗМІСТ

ВСТУП

Дана робота присвячен розробці і дослідженню автоматизованої системи для обліку реалізації товарів та послуг на основі обробки даних з електронного реєстратора розрахункових операцій.

Основними позитивними наслідками впровадження розробленої автоматизованої системи є:

- об’єктивність контролю та керування;
- виключення впливу суб’єктивних факторів;
- можливість централізації керування торгівельними точками і цілими комплексами практично без обмеження відстані;
- ефективність використання торгівельного обладнання, шляхом підвищення продуктивності торгівельних точок за рахунок оптимізації технологічних процесів шляхом зменшення часу обслуговування;
- отримання адекватної інформації про стан торгівельних вузлів;
- надійність роботи комплексу, можливість відновлення системи після виникнення і розвитку аварійних ситуацій;
- підвищення продуктивності та покращення умов праці, зростання кваліфікації кадрів, зменшення кількості обслуговуючого персоналу.

Запропонований варіант автоматизованої системи оснований на розробленому програмному забезпеченні для платформи Windows, яке дозволяє полегшити ведення обліку, інтегрувати фізичне обладнання для торгівлі з системами обліку вищого рівня. Кінцевою метою вдосконалення автоматизованої системи є отримання додаткового прибутку підприємством.

Застосування систем управління з використанням програмних засобів управління на основі комп’ютерної техніки обумовлено універсальністю, високою надійністю в експлуатації, можливістю зміни логіки функціонування. Вартість таких систем нижча вартості аналогічних систем, які створені на основі традиційних технічних засобів автоматичного управління.

1. АНАЛІТИЧНА ЧАСТИНА

1.1 Аналіз відомих програмних додатків та комплексів для вирішення вибраного напрямку розробки

Схожі задачі до запропонованої системи дозволяє частково вирішити програма MINI600 від виробника електронних реєстраторів розрахункових операцій НПФ „ЮНИСИСТЕМ”.

Програма призначена для програмування електронних реєстраторів розрахункових операцій (ЕРРО) типу MINI600ME, що підключаються до комп'ютера через послідовний порт COM1, COM2, …, COM8, а також для зчитування звітних даних про продаж товарів з електронних реєстраторів розрахункових операцій.

Під програмуванням електронних реєстраторів розрахункових операцій, у даному випадку, розуміється запис нормативно-довідкової інформації в електронні реєстратори розрахункових операцій, що необхідна для їх подальшої експлуатації в автономному режимі.

Дані, що програмуються в електронні реєстратори розрахункових операцій:

інформація про товари і послуги: артикул, штрих-код, назва, ціна, залишок, відділ, ознака одиночного продажу;

інформація про відділи: номер відділу, назва, ціна, податкова група, ознака одиночного продажу;

- інформація про податки: вид податку (ПДВ, що додається ПДВ, податок з обороту); процентні ставки по податкових групах;
- інформація про націнки: дозволені/заборонені; процентна ставка;
- інформація про знижки: дозволені/заборонені; процентна ставка;
- повідомлення на чеку: початкове повідомлення; кінцеве повідомлення; альтернативні повідомлення;
- інші параметри касових апаратів: сервісний код доступу до команд

електронних реєстраторів розрахункових операцій; ознака передачі копії касової стрічки на ПК; ознака включення/відключення принтера електронних реєстраторів розрахункових операцій; ознака автоматичного вимикання касових апаратів.

Звітні дані, що зчитуються з електронних реєстраторів розрахункових операцій:

− дані про товари, зареєстровані в ЕРРО до видачі Z-звітів: продана кількість, продано на суму;

− дані про відділи, зареєстровані в ЕРРО до видачі Z-звітів: продано на суму.

Програма дозволяє виконувати наступні функції:

− зчитати запрограмовані і звітні дані з ЕРРО або з каталогу;

− відкоригувати і доповнити запрограмовані дані;

− роздрукувати на принтері запрограмовані і звітні дані;

− записати дані для програмування в цей же або інший ЕРРО;

− виконати експорт даних (зберегти дані для програмування і звітні дані в окремому каталозі у форматі DBF-файлів);

− виконати імпорт даних (зчитати дані зі збережених DBF-файлів).

Реалізація такої технології дозволяє:

− попередньо підготувати дані на комп'ютері не підключаючи ЕРРО;

− виконувати експорт/імпорт даних між різними комп'ютерами і ЕРРО;

− виконувати підготовку або наступну обробку DBF-файлів за допомогою зовнішніх додатків;

− підключати ЕРРО до комп'ютера тільки для виконання операції зчитування/запису даних.

Вимоги до комп'ютера й операційного середовища:

1. Операційне середовище:

- Windows серії.

2. Конфігурація комп'ютера:

- монітор (роздільна здатність не менше 800x600) ;
- принтер (при необхідності).

Програмний комплекс „KasSer” призначений для автоматизації технології обслуговування покупців у торговельних підприємствах усіх типів. „KasSer” дозволяє збільшити пропускну здатність розрахункових касових вузлів торговельного залу з мінімальними витратами на додаткове устаткування.

„KasSer” автоматизує практично усі функції керування торговельного підприємства: товарообіг, рух товарів, торговельну статистику і кількісний облік. „KasSer” забезпечує роботу торговельних підприємств із використанням штрих-кодових технологій, підтримуються усі формати штрих-кодових етикеток.

„KasSer” дозволяє проводити розрахунки по кредитних картках через касові апарати. Windows-версія „KasSer” написана в сучасному середовищі розробки, тому можливий обмін даними з будь-якими системами торговельного і бухгалтерського обліку.

Програмний комплекс „KasSer”, встановлений на одному комп'ютері, керує роботою підключених до нього касових апаратів або фіскальних реєстраторів, що працюють у режимі on-line, off-line або режимі фіскального принтера.

„KasSer” дозволяє обслуговувати різні типи касових апаратів і фіскальних реєстраторів одночасно. Це можуть бути: касові апарати MINI-600, MINI-500, фіскальні реєстратори MINI-ФП – у будь-яких сполученнях, що підключені до одного комп'ютера.

Підтримується робота з електронними вагами типу VE15 і електронними вагами з принтером штрих-кодових етикеток типу LP15 або ВП15.

Штрих-кодові етикетки надруковані на вагах LP15 або ВП15 обробляє будь-яка каса підключена до комп'ютера з встановленим програмним комплексом „KasSer”.

„KasSer” реєструє і накопичує дані про продажі товарів через усі підключені до нього касові апарати або фіскальні принтери в докладному асортименті.

Мінімальний набір устаткування необхідний для роботи „KasSer” містить у собі: комп'ютер, касовий апарат або фіскальний реєстратор , сканер штрих-кодових етикеток підключений у розрив клавіатури комп'ютера, зв'язок по локальній мережі з базою даних товарів.

Складові частини програмного комплексу „KasSer”:

1. Блок зв'язку з торговельним устаткуванням:

− система підтримує зв'язок з касовими апаратами типу SAMSUNG, MINI, SILEX. Зв'язок здійснюється через COM-порти комп'ютера (COM1, COM2) або розширювач COM-портів – мультиплексор (COM3 – COM 99);

− одночасно підтримує зв'язок з ЕРРО різних типів: COM1 – MINI, COM2 –SAMSUNG, COM3 – SILEX і т.д. , підключених до одного комп'ютера. Каси при цьому можуть працювати в режимі on-line (запитують атрибути товарів у комп'ютера, коди яких вводить касир із клавіатури касового апарату або зчитує за допомогою сканера штрихів-кодів, у відповідь одержують від комп'ютера інформацію для рядка чека) або в режимі фіскального принтера;

− зчитувачі штрих-кодових етикеток різних типів, підключених через COM-порти, через ЕРРО або в розрив клавіатури комп'ютера;

− термінали, підключені через COM-порти або комп'ютерну мережу;

− платіжні термінали;

− ваги з принтером штрих-кодових етикеток, підключені через COM-порти, через термінали.

2. Система прийому і підготовки бази даних товарів для реалізації:

− експорт даних з різних систем бухгалтерського і торговельного обліку: АККИС-„Торг система”, 1С – торгівля і т.д, SMarket;

− використовуються джерела ODBC-Windows або текстові файли для експорту-імпорту даних по наявних товарах;

− для каси в режимі фіскального принтера можливий експорт чеків у текстовому форматі і використанням файлу-прапорця. Кодування тексту як в OEM-таблицях так і ANSI.

3. Система формування вихідних документів для клієнтів:

− товарна накладна;

− податкова накладна.

4. Система корпоративного обміну інформацією:

− кілька торговельних об'єктів одержують товари з одного центрального складу. У кожній точці встановлений комп'ютер та модем. Накладні надходять з центрального офісу через модем. Звіти по реалізації з кожної торговельної точки надходять у центральний офіс, а також через модем.

Програмні продукти серії „АБАК” складаються з мережевих програм для торгівлі АБАК/Торгівля і для бухгалтерського обліку АБАК/Універсальна бухгалтерія. Дані з програми АБАК/Торгівля експортуються в програму АБАК/ Універсальна бухгалтерія.

Програми мають інтерфейс в дусі Microsoft® Office.

Програми реалізовані засобами СУБД Microsoft® Visual FoxPro 7.0.

Програми працюють в режимах файл-сервер і клієнт сервер. Кожна з програм складається з бази даних і робочих місць адміністратора системи, загального і спеціалізованих робочих місць. Прикладом спеціалізованого робочого місця є робоче місце касира.

Програми забезпечують обмін даними між робочими місцями з різних локальних мереж.

Для режиму файл-сервер на ПК має бути встановлена одна з операційних систем Microsoft Windows. Режим файл-сервер доцільно використовувати при 1-4 робочих місць в локальній мережі.

Режим роботи клієнт-сервер реалізовано на базі технології DCOM. На сервері,, тобто ПК з базою даних, встановлюється одна з операційних систем: Windows. Для підтримки роботи технології багатопоточної DLL в середовищі Windows необхідно встановити пакет Microsoft® Transaction Server, а в деяких Windows ця підтримка здійснюється вбудованими в цю ОС засобами COM+. Зауважимо, що робоче місце касира, на відміну від інших, працює тільки в режимі клієнт-сервер.

Програми серії АБАК мають уніфіковані засоби для формування звітів і форм первинних документів, для зміни їх форми і змісту. На базі шаблонів формується документ в форматі RTF, який або роздруковується безпосередньо, або передається для подальшої обробки в текстовий процесор Microsoft Word. Якщо на комп'ютері не встановлено Microsoft Word, то документ передається в Microsoft WordView. Microsoft WordView не потребує ліцензії на використання.

Програма АБАК/Торгівля застосовується на підприємствах гуртової та роздрібної торгівлі, в тому числі на супермаркетах. Використовується для автоматизації процесів торгівлі, обліку руху товарів, розрахунків за товар, маркетингового аналізу.

В програму входять робочі місця: основне, адміністратора системи і спеціалізоване робоче місце касира.

Програма підтримує роботу з касовими апаратами, сканерами штрих-коду, електронними вагами, кредитними картками.

Програма АБАК/Торгівля забезпечує облік:

- руху товару: прихід, внутрішні переміщення, реалізація, залишки,

резервування товару по рахунках та замовленнях, списання, інвентаризація, переоцінка, застосування знижок/націнок різних типів, замовлення товару;

− ПДВ: різні варіанти обчислення, формування податкових накладних, ведення книг обліку придбання та продажу;

− розрахунків за товар: прихід/розхід грошей і бартерні операції, дебітори, кредитори, дані про непроплачений товар;

− наданих послуг.

Підсистема звітності надає велику кількість звітів про рух товару й грошей на будь-яку дату або за довільний період часу в різних розрізах.

Можливості програми АБАК/Торгівля не обмежуються вище зазначеними і відповідають поняттю “Програма для торгівлі”.

Програми серії АБАК мають драйвери для фіскальних принтерів „Марія-301”, „Марія-301МТМ” ФП 3530Т; для касових апаратів типу MINI-500, MINI-600 та MINI-ФП.

1.2 Огляд відомих технічних рішень з питання обліку реалізації товарів та послуг

Сервер касових апаратів призначений для організації комп’ютерно-касових мереж на базі пасивних електронних реєстраторів розрахункових операцій.

Сервер є сполучною ланкою між програмним забезпеченням, що реалізує функції товарного обліку, і касовими апаратами, що працюють у торговельному залі.

Загальна схема роботи виглядає наступним чином.

Програмне забезпечення товарного обліку формує прайс-лист, що містить інформацію про найменування, ціну, код, штрих-код, кількості і т.д. всієї номенклатури, що знаходиться в торговельному залі.

У процесі роботи касові апарати запитують сервер про параметри того або іншого товару по коду або штриховому-кодові, сервер повертає касам запитану інформацію і робить у журналі продажів запис про продаж товару. Так само обробляються операції ануляції, повернення, виконання знижки або надбавки.

Програмне забезпечення товарного обліку по команді оператора або автоматично зчитує журнал продажів і здійснює операції обліку руху товару й оформлення відповідних документів. Наприклад, списує проданий товар зі складу й оформляє видаткові накладні.

Сервер не дозволяє створювати і редагувати прайс-лист, програма використовує вже створений файл.

До складу комп’ютерно-касової мережі входять касові апарати, комп’ютер, до якого ці каси підключаються і кабельна система, що забезпечує електричне з'єднання комп’ютера з касовими апаратами.

Касові апарати, що входять до складу комп’ютерно-касової мережі можуть бути наступного типу:

- ОРАНТ-121-Ф;
- ЭЛВЕС 01-03Ф;
- ШТРИХ М850Ф;
- ЕЛКА 82МБ-Ф;
- Samsung 250,350, Datex 500.

Для забезпечення можливості роботи в складі комп’ютерно-касової мережі, у касові апарати повинна бути встановлена інтерфейсна плата. Крім можливості зв'язку з комп'ютером, така плата дозволяє підключити до касового апарату сканер штрих-коду, електронні ваги або додаткові клавіатури.

Касові апарати підключаються до послідовного COM-порта комп'ютера по кабельних лініях зв'язку. Для кабельної системи торговельного підприємства може бути використаний плоский

чотирьохжильний телефонний кабель або екранована кручена пара 3-ой категорії. Вибір конкретного типу кабелю визначається топологією мережі, відстанню між касовими апаратами і ПК, наявністю зовнішніх завад на лінії зв'язку і т.п.

Кілька касових апаратів підключаються паралельно за топологією „загальна шина”. При цьому якщо кількість апаратів більше 2 або довжина кабелю більше 30м, то між касовими апаратами і комп'ютером включається спеціальний підсилювач.

Для підключення касового апарату до магістрального кабелю можна використовувати будь-які телефонні розетки або розгалужувачі.

Без використання підсилювача до одного порту комп'ютера можна підключити два касових апарати, відстань між ними і комп'ютером не повинна перевищувати 30м.

З використанням підсилювача кількість апаратів, що одночасно підключаються, може бути збільшена до 10, а відстань – до 1000 м.

Комп'ютер, до якого підключені касові апарати, сам по собі може бути включений до складу локальної обчислювальної мережі.

2. ТЕХНОЛОГІЧНА ЧАСТИНА

2.1 Характеристика обладнання для здійснення розрахункових операцій

Електронний реєстратор розрахункових операцій типу MINI-600ME

MINI-600ME – електронний контрольно-касовий апарат (рис. 2.1), що живиться від вмонтованої акумуляторної батареї, яка заряджається від напруги змінного струму 220В через додатковий блок живлення (адаптер). Функціональні можливості цього апарату відповідають вимогам фінансової звітності, що стосуються підприємств торгівлі, громадського харчування та сфери послуг.

Рисунок 2.1 – MINI-600ME

ЕККА має фіскальну пам'ять, що задовольняє технічним вимогам до електронних контрольно-касових апаратів, які затверджені Постановою Кабінету Міністрів України №199 від 18.02.2002 р.

Сканери штрих-кодових етикеток

Ручний лазерний сканер штрих-кодів MS 9520 Voyager (рис. 2.2) використовується в касових терміналах у невеликих роздрібних магазинах, для обробки облікових документів тощо.

У комплект входить стенд, що дозволяє використовувати сканер як стаціонарний пристрій. Сканер містить вбудований інфрачервоний сенсор, що забезпечує включення сканера з появою об'єкта в області зчитування. Максимальна ширина захоплення сканера – 297 мм.

Рисунок 2.2. Ручний сканер штрих-кодових етикеток MS 9520 Voyager

Основне призначення даної моделі – зчитування штрих-кодів типу EAN/UPC зі стандартних упаковок товарів, тобто використання в невеликих магазинах на робочому місці продавця, касира і комірника. Даний сканер може бути підключений до будь-якого касового апарата, що має можливість підключення сканера.

Сфери застосування:

- магазини;
- маркети.

Високошвидкісний сканер MS 860i-17 з можливістю вбудовування (рис. 2.4) розроблений спеціально для використання на невеликих робочих місцях, оскільки він майже в 2 рази менше і легше, ніж більшість сканерів штрих-кодів, що вбудовуються.

Рисунок 2.4 – Сканер штрих-кодових етикеток MS 860 i-17

Ваги з можливістю друку штрих-кодових етикеток

Ваги тупу ВП є вагами середнього класу точності (сертифікат RU.C.28.001.A, №6243, зареєстровані в державному реєстрі під № 18341-99) з вбудованим принтером для друку етикеток та дозволені для проведення торгівельних та облікових операцій на торгівельних підприємствах, громадського харчування, в харчовій промисловості, а також в інших галузях народного господарства. Ваги можуть використовуватись як автономно, наприклад, при операціях фасування, так і в складі автоматизованих торгівельних і виробничих комплексів.

Можливості ваги та основні технічні характеристики:

- цифрове відображення маси, ціни і вартості товару, що зважується, як зі сторони продавця, так і зі сторони покупця;
- відображення назви товару на алфавітно-цифровому дисплеї;
- облік маси тари при зважуванні товару;
- розрахунок вартості штучних товарів, сумування покупок одного покупця з роздруком найменувань всіх товарів, їх маси, ціни, вартості і результату розрахунку з покупцем;
- запам’ятовування цін і основних характеристик до 999 товарів;
- зберігання пам'яті ваги і працездатність вбудованого годинника при виключенні живлення;

- швидкий виклик характеристик товарів з пам'яті при допомозі 63-х клавіш товарів, а також по номеру товару чи назві;
- швидкий виклик з пам'яті ваги десяти значень маси тари;
- оперативне корегування цін товарів, що зберігаються в пам'яті;
- підведення підсумків за день по кожному товару, по групі товарів, загальних підсумків;
- друк різних варіантів етикеток, які можуть містити наступні характеристики товарів:
  - вага товару, що зважується;
  - ціну за кілограм товару;
  - вартість товару, що зважується;
  - різноманітні варіанти штрих-коду;
  - назва товару (два рядки по 24 символи);
  - дату фасування товару;
  - термін придатності товару;
  - склад товару (до 12 рядків по 48 символів);
  - знаки відповідності при обов’язковій сертифікації;
  - рекламну інформацію про фірму-продавця чи виробника (два рядки по 24 символи);
  - виготовлення різних типів термопаперу (з липким шаром і без нього) у вигляді окремих етикеток та суцільним;
  - друк в автоматичному режимі (по закінченню зважування), в ручному (по команді оператора) або відключення друку;
  - друк декількох однакових етикеток;
  - прийом та передавання інформації по інтерфейсу RS-232, що дозволяє:
    - підключити до одного комп’ютера до 99 ваг;

- вводити в пам'ять ваги характеристики товарів не тільки з клавіатури ваги, але і з комп’ютера;
- копіювати запрограмовані дані з однієї ваги на іншу;
- підключати вагу до контрольно-касових апаратів;
- створювати локальні торгові сітки.

2.2 Принципи роботи з електронним реєстратором розрахункових операцій типу MINI-600ME

Програмування електронних реєстраторів розрахункових операцій

Під програмуванням електронних реєстраторів розрахункових операцій розуміється запис нормативно-довідкової інформації в електронні реєстратори розрахункових операцій, що необхідна для їх подальшої експлуатації в автономному режимі.

Програмування назви або коду товару здійснюється за такою схемою:

611 ПС 12321 ПС *Назва товару*

У назві товару може бути до 20 символів.

Після того, як була введена команда, на індикаторі з’являється запит: „? Apt”. Цей запит номера товару, дані для якого будуть програмуватись, властивий для всієї послідовності команд програмування товару „611, 612, 613, 614, 615, 617”. У відповідь необхідно ввести номер товару, який програмується.

Перехід до наступної команди у вищезгаданій послідовності здійснюється при досягненні максимального номера товару або в результаті натискання клавіш „КОД ТОВ”. При цьому номер товару залишається таким, який був введений спочатку . Перехід до наступного товару здійснюється до відповідного коду товару після введення послідовності „код” „КОДТОВ”.

Для 611 команди ця функція не здійснюється.

Програмування ціни товару проводиться наступним чином:

612 ПС 12321 ПС *Ціна товару*

Максимальна ціна товару може становити 9.999.999,99. Ціна товару вводиться в копійках, а два знаки після коми встановлюються автоматично. Якщо для товару запрограмована нульова ціна, то такий товар вважається товаром із „відкритою ціною”. Це зручно у випадку, коли у переліку товарів є однотипні товари. Можна проводити продаж такого товару, вказуючи ціну товару безпосередньо у операції продажу.

Програмування належності до відділу здійснюється так:

613 ПС 12321 ПС НОМЕР ПС *Належність до відділу*

По цій команді вводиться НОМЕР – номер відділу від 1 до 16. Через належність до відділу товар пов’язують з податковою ставкою, тобто товар відноситься до тієї податкової групи, яка запрограмована для відповідного відділу (603 командою).

Програмування одиночної позиції товару виконується за наступною схемою:

614 ПС 12321 ПС ПАРАМ ПС *Одиночний продаж товару*

У випадку, коли ПАРАМ встановлений одиниці, то після продажу такого товару чек закривається, тобто здійснюється одиночний продаж.

Програмування товарної наявності здійснюється наступним чином:

615 ПС 12321 ПС *Товарна наявність*

Якщо товарна наявність запрограмована, то у випадку, коли ця кількість вичерпана, на індикаторі виводиться повідомлення „ПОМИЛКА17”. Якщо ж дані про товарну наявність на програмуються, то у звітах реєструється реалізована кількість даного товару як від’ємна величина.

Програмування штрих-коду товару виконується так:

617 ПС 12321 ПС *Штрих-код*

Штрих-код – це цифрова інформація. Може приймати значення від 1

до 9999999999999. На індикаторі відображається старша половина введеного значення. Для того щоб побачити молодшу половину штрих-коду, необхідно натиснути клавішу „АН/ЦІНА”.

Перепрограмування параметрів командами „611”, „613”, „617” для товарів, по яким проводився продаж, можливе після виконання звітів з обнуленням „501” та „503”. Командами „612”, „614” та „615” перепрограмування параметрів можна виконувати в будь-який момент.

Виконання звітів реєстратора розрахункових операцій

ЕККА виконує три групи звітів:

- X-звіти без обнулення: „101” – „103”, „105”, „106”;
- Z-звіти з обнуленням „501” – „503”, „505” – „507” ;
- звіти з фіскальної пам’яті: „701” – „705”.

Всі звіти з обнуленням, окрім денного „501” звіту, не є фіскальними і виконують допоміжні функції.

Денний звіт з обнуленням „501” є фіскальним, при його виконанні інформація з оперативної пам'яті записується у фіскальну пам'ять.

Після закінчення робочої зміни необхідно виконати звіт „501”. У випадку, коли тривалість робочої зміни (від часу видачі першого чеку до зняття „501” звіту) перевищує 24 години, ЕККА блокується „ПОМИЛКА18” і може бути розблокований виконанням звіту „501”.

Денні звіти з обнуленням нумеруються з першого від моменту фіскалізації і підклеюються до касового журналу.

При використанні інформації із звітів „502” та „503” необхідно врахувати те, що це звіти з накопиченням.

Порядок роботи із іншими зовнішніми пристроями:

Для роботи ЕККА з ПК необхідно виконати такі дії:

- підключити ПК до ЕККА (порт 1, роз’єм „ПК”) за допомогою інтерфейсного кабелю;

- запрограмувати режим роботи послідовного порту за допомогою команди 618.

До ЕККА можна підключати сканер штрих-кодів.

Для роботи ЕРРО зі сканером необхідно виконати такі дії:

- підключити сканер до ЕККА (порт 3, роз'їм „СКАНЕР");
- запрограмувати режим роботи послідовного порту за допомогою команди 614;
- включити сканер та ЕККА згідно з експлуатаційними документами на них.

Параметри сканеру необхідно запрограмувати згідно експлуатаційним документами на нього, в тому числі:

- настроїти набір штрих-кодів „Start Group 1", „Set to default 1", „Exit Group 1";
- настроїти параметри порту RS 232 „Start Group 5", „8 Bits None", „120", „CR", „CTS/RTS", „Exit Group 5".

Порядок роботи при виконанні функцій продажу товарів:

Виконання касових операцій починається з реєстрації касира. Реєстрація здійснюється введенням номера касира та, якщо необхідно, пароля касира:

номер касира ПС

Наприклад, реєструється восьмого касиру: 8 ПС

Якщо після виконання цієї операції на індикаторі з'являється запитання, необхідно ввести пароль касира. При відсутності пароля на індикаторі з'являється „0", що вказує на готовність ЕККА до проведення касових операцій.

Розреєстрація виконується тоді, коли апарат залишається без нагляду або коли до роботи приступає інший касир.

2.3 Опис протоколу взаємодії ЕРРО типу MINI-600 з ПК

Взаємодія реєстратора розрахункових операцій з комп'ютером відбувається по інтерфейсу RS-232 (сигнали – TxD і RxD).

Протокол обміну: 8 біт, 1 стоп біт, без контролю парності, швидкість обміну – 19200, 9600, 4800, 2400 бод.

У режимі роботи з ПК відбувається обмін блоками інформації між ЕРРО і ПК з ініціативи ЕРРО. Блок – послідовність символів, що закінчується обмежувачем блоку (код 0DH).

ЕРРО передає блок інформації і очікує відповідь від ПК у виді блоку, максимальної довжини 128 символів, включаючи обмежувач блоку.

Обробка прийнятого блоку в ЕРРО відбувається після повного прийому блоку.

Розрив зв'язку настає якщо:

- на протязі 1сек не отриманий жоден байт – відсутність зв'язку;
- визначено розрив зв'язку – таймаут усередині блоку, переповнення буфера приймача.

Для запобігання розриву зв'язку під час обробки прийнятого блоку, ПК повинний підтримувати зв'язок з ЕРРО передачею символу '@' (не більш 50 одночасно).

При розриві зв'язку на індикатор ЕРРО виводиться повідомлення "? ЗАПР. РС", а на ПК передається ідентифікаційний рядок. У випадку збою зв'язку, перед передачею ідентифікаційного рядка буде переданий символ 'G'. При одержанні відповіді від ПК, у виді блоку нульової довжини, зв'язок відновлюється.

ЕРРО передає на ПК коди натиснутих клавіш, значення штрих-коду, обмеженого символами '"' або порожній блок, одержує від ПК командний рядок і відпрацьовує його як натискання клавіш. У командному рядку також

може міститись назва товару у виді стрічки, що обмежена символами “”. Максимальна довжина назви товару – 63 символи.

ЕРРО передає вміст діапазону пам'яті при одержанні команди ДАМП ПАМ'ЯТІ НА ПК або команди ПЕРЕДАЧА ДАНИХ НА ПК. Формат: байт 7СН, кількість байт (у форматі молодший байт, старший байт), байти пам'яті, обмежувач блоку. Діапазон кількості байт: 1-65535 байт.

ЕРРО передає копію друку при встановленому параметрі РЕЗЕРВ загальних параметрів. Передається тип рядка (2 – одинарні висоти, 4 – подвійні висоти) і рядок або байт 08Н – переклад рядка, 03 – обмежувач друку.

Обмежувач блоку передається після завершення друку всіх рядків.

ЕРРО передає номер помилки у форматі: символ ‘Е’, номер помилки в символьному виді, обмежувач блоку.

При передачі даних між ПК і ЕРРО використовуються коди спеціальних (службових) символів.

3 КОНСТРУКТОРСЬКА ЧАСТИНА

3.1 Підключення електронного реєстратора розрахункових операцій до ПК

Касові апарати підключаються до порта комп'ютера RS232 за допомогою спеціалізованої плати, що змонтована у касовому апараті. Для підключення використовується спеціальний кабель.

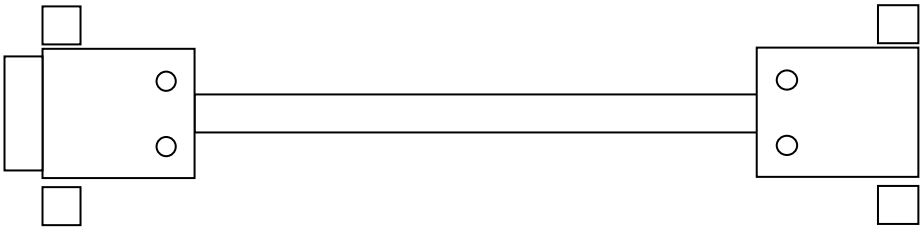
Інтерфейсна плата касового апарату може містити наступні інтерфейсні порти:

1. порт для підключення ваги;
2. порт для підключення сканера штрих-коду. В платі живлення на сканер не подається, тому з нею будуть працювати сканери, що мають зовнішнє або автономне живлення;
3. порт для підключення ПК;
4. порт для підключення терміналу.

Касові апарати підключаються до послідовного (COM) порта комп'ютера по кабельних лініях зв'язку за топологією зірка. Для кабельної системи торговельного підприємства може бути використаний плоский чотирьохжильний телефонний кабель або екранована кручена пара 3-ой категорії. Вибір конкретного типу кабелю визначається відстанню між касовими апаратами і ПК, наявністю зовнішніх завад на лінії зв'язку і т.п.

Не допускається перекручування жил у парах при розпаюванні кабелю.

Інтерфейсний шнур для під’єднання ЕРРО до ПК приведений на рис. 3.1.



До комп'ютера
розетка DB-9F

До касового апарату
вилка DB-9M

Рисунок 3.1 – Інтерфейсний шнур для під’єднання ЕРРО до ПК

Вищеописаний тип підключення може бути реалізованим при невеликій відстані касового апарату від серверного ПК. При далеких відстанях використовується прокладена лінія з телефонними розетками під конектор RJ6.

Схема підключення наступна (рис. 3.2):

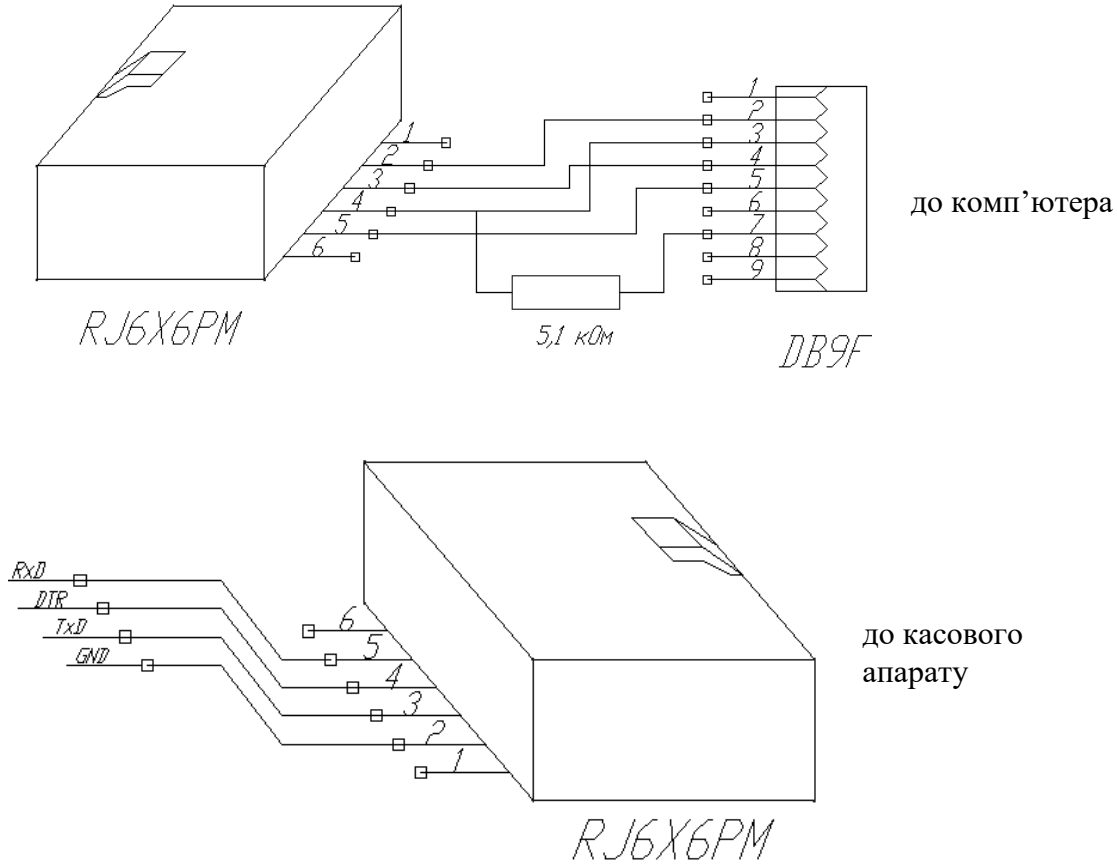


Рисунок 3.2 – Схема підключення сигнальних ліній від комп’ютера та касового апарату до конекторів RJ6.

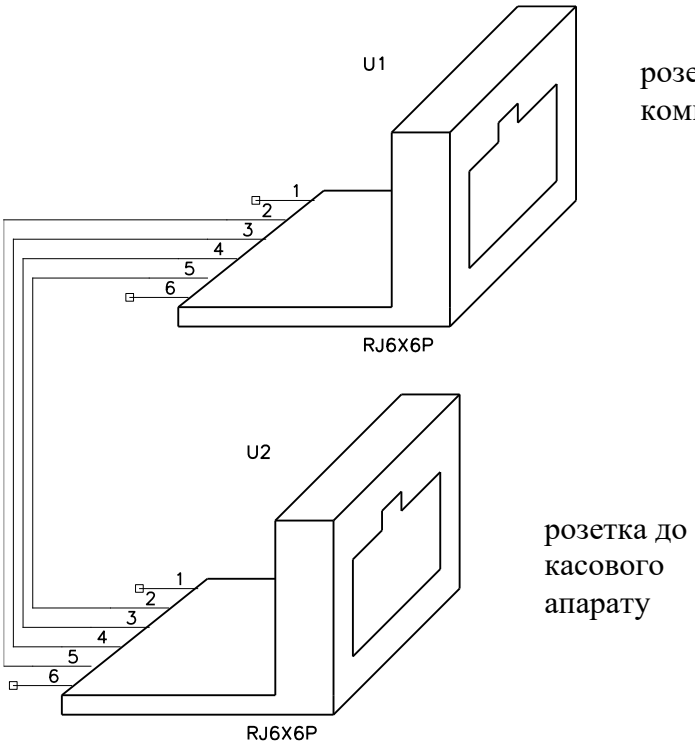


Рисунок 3.3 – Схема комутації сигнальних ліній між розетками RJ6

Сигнал	Контакт від комп’ютера		Контакт до розетки
RXD	2	⟷	2
TXD	3	⟷	4
DTR	4	⟷	3
GND	5	⟷	5

Рисунок 3.4 – Схема контактів від комп’ютера до розетки.

Підєднання сигнальних ліній від комп’ютера та EPPO до розеток показано на рисунках 3.3 - 3.5.

Контакт від розетки		Контакт до касового апарату	Сигнал
3	⟷	3	RxD
2	⟷	1	GND
5	⟷	2	TxD

Рисунок 3.5 – Схема контактів від розетки до касового апарату (або ваги).

Інтерфейс RS-232C/CCITT V24 є широко розповсюдженим стандартом послідовного зв’язку між мікропроцесорними системами та периферійними пристроями. Інтерфейс, що визначений стандартом Асоціацією електронної промисловості (EIA), передбачає наявність обладнання двох видів: термінального DTE та зв'язкового DCE.

Рівні сигналів при передачі даних від ЕРРО до ПК по інтерфейсу RS-232 зображені на рис. 3.6.

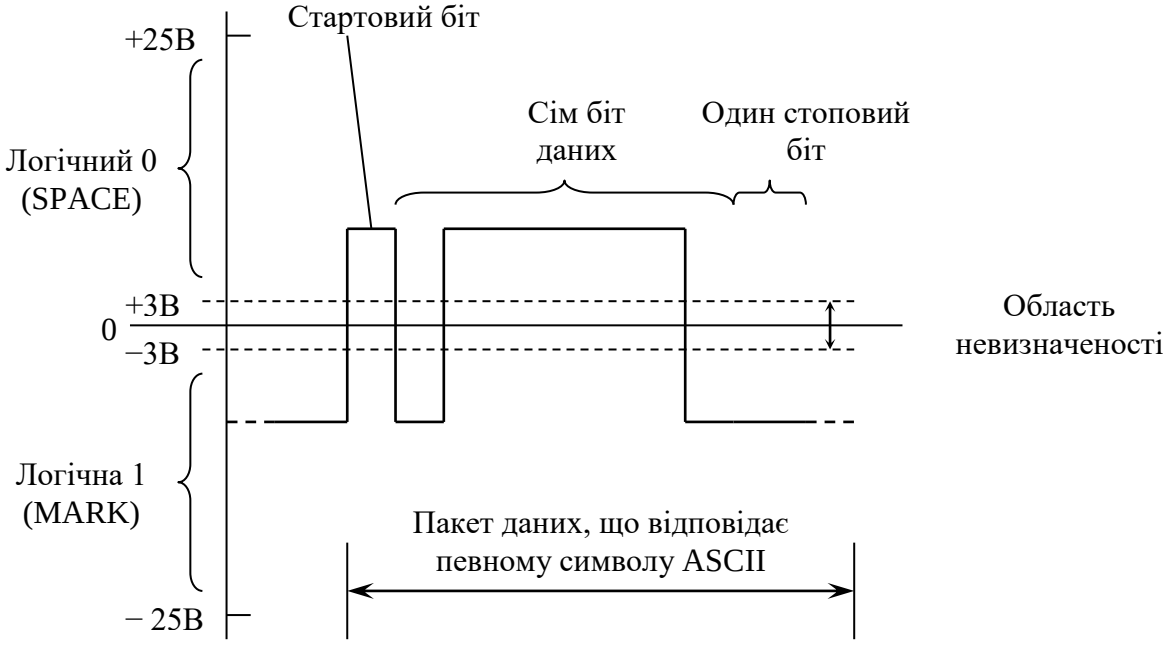


Рисунок 3.6 – ТТЛ-сигнал при передачі даних по інтерфейсу RS-232C

Термінальне обладнання може передавати та приймати дані по послідовному інтерфейсу. Воно ніби то закінчує (terminate) послідовну шину. Під зв’язковим обладнанням розуміються пристрої, які можуть спростити послідовну передачу даних разом з термінальним обладнанням.

Сигнали інтерфейсу RS-232C поділяються на наступні класи:

- послідовні дані
- керуючі сигнали квітіювання
- сигнали синхронізації

В більшості систем, що використовують інтерфейс RS-232C, дані передаються асинхронно, тобто в послідовності пакетів даних. Кожен пакет містить один символ коду ASCII (американський стандартний код для обміну інформацією), інформація в пакеті достатня для його декодування без окремого сигналу синхронізації.

Символи коду ASCII становлять сім бітів та для того, щоб передати їх по інтерфейсу RS-232C, необхідно ввести додаткові біти, що позначають початок та кінець пакету. Крім того, виникає необхідність ввести додатковий біт для контролю помилок по паритету (парності).

При передачі даних між ЕРРО та комп’ютером використовується формат, що включає в себе один стартовий біт, один стоповий біт, без контролю паритету.

При підключенні до COM-порта комп'ютера касового апарата необхідно враховувати наступне:

- кожен касовий апарат підключаються до комп'ютера на один COM-порт;
- якщо відстань між комп'ютером і касовим апаратом перевищує 30 метрів необхідно скористатися спеціальним підсилювачем (адаптером).
- Ця довжина може змінюватись в залежності від якості кабелю, і

зовнішніх умов (завад).

Схема спеціального підсилювача наведена в графічній частині.

Комп'ютер, до якого підключені касові апарати, сам по собі може бути включений до складу локальної обчислювальної мережі.

З використанням підсилювача відстань може бути збільшена до 100м.

Адаптер інтерфейсу RS-232 забезпечує:

- гальванічну розв'язку сигнальних ліній TxD, RxD через транзисторні оптрони;
- переведення порту в режим „токової петлі”. Сприяє підвищенню завадозахищеності, що дозволяє підвищити лінію передачі до 100 метрів.

Електрична принципова схема, розміщення елементів та друкована плата адаптера інтерфейсу RS-232 зображена в графічній частині дипломного проекту.

3.2 Налаштування інтерфейсу RS-232 для взаємодії ПК з ЕРРО

У літературі, найчастіше, керування послідовним і паралельним портами описується на рівні регістрів цих портів, причому приклади програм приводяться мовою Assembler. Послідовний порт досить повільний та специфічний пристрій. Тому в програмах, що працюють з портами використовуються переривання. Паралельний порт швидший, але теж повільний і не менш специфічний. Він має можливість працювати в двох напрямках, так ще і з використанням ПДП (DMA).

Написання програм, що керує пристроєм через COM порт, для MS-DOS простіше ніж для Win32. Це пов’язано з тим, що прямо працювати з регістрами портів не можна, Windows це не дозволяє, але перевагою є

незалежність від різних реалізацій, не потрібно працювати з обробкою переривань.

Windows вимагає точного дотримання апаратного потоку обміну з зовнішніми пристроями. Наприклад, не можливо керувати, пристроєм, підключеним лише до одного з виходів паралельного порту. Система буде вимагати обробки також сигналів STROBE і ACK.

З послідовними і паралельними портами в Win32 працюють як з файлами. Починати треба з відкриття порту як файлу. Використовувати звичні функції open і fopen при цьому не можна, необхідно скористатися функцією CreateFile.

3.3 Розробка термінального касового сервера

Вибір методу синтаксичного аналізу:

Існує декілька наступних методів синтаксичного аналізу:

- спадний синтаксичний аналіз, що виконується рекурсивним спуском (top-down recursive descent);
- метод зштовхування ізольованих ділянок (island collision);
- метод LALR;
- метод простого слідування;
- та деякі інші.

Деякі з вказаних вище методів описані в класичній книзі Principles of Compiler Design („Принципи побудови компіляторів”) Альфреда В.Ахо (Alfred V.Aho) та Джеффри Д.Ульмана (Jeffry D.Ullman) видавництва Addison-Wesley. Метод, використаний в дипломному проекті є модифікованим спадним синтаксичним аналізом, що виконується рекурсивним спуском з попереднім переглядом і початковим лексичним аналізом.

При зростаючому синтаксичному аналізі перегляд починається найменших частин синтаксису, тобто частин, що в більшій степені відповідають вхідному тексту, а потім поступово з простіших структур створюються більш складніші. Якщо при такому порядку аналізу у вхідному тексті міститься помилка і не відповідає можливому синтаксису, тоді цей факт виявляється на ранніх етапах аналізу.

При спадному синтаксичному аналізі (top-down parsing) перегляд починається з тої частини синтаксису, яку необхідно в кінцевому випадку розпізнати (що і є кінцевою ціллю аналізу), а потім ми поступово переходимо в напрямку окремих символів тексту. Іноді такий підхід значить перегляд всього вхідного тексту тільки для того, щоб визначити невідповідність, а після чого доводиться повертатись назад і вибирати іншу альтернативу. Спадний синтаксичний аналіз може зайняти набагато більше часу, ніж зростаючий аналіз, однак такий синтаксичний аналіз набагато простіше написати. Якщо в аналізованому тексті (кодах клавіш, натиснутих на ЕРРО) міститься помилка (невідповідність синтаксису), то впевненості в цьому не буде до тих пір, поки не будуть перевірені всі можливі варіанти аналізу.

Можна скористатися перевагою простоти написання і уникнути недоліку повільного виконання, ввівши в аналіз дві наступних властивості: лексичний аналіз і попередній перегляд. Лексичний аналіз швидше нагадує деякій зростаючий аналіз. При цьому вибираються синтаксичні класи і приймається рішення про те, що вони настільки розповсюджені і легко розпізнаються незалежно від контексту, що має місце проаналізувати саме їх. Це значить, що пошук таких елементів виконується тільки один раз.

Коли в правилі синтаксису міститься декілька альтернатив, вони можуть бути розглянуті в будь-якому порядку. Найкращим використовуваним порядком буде той, при якому спочатку проглядається найдовша з можливих альтернатив.

До кожної альтернативи може бути додано дві розріджені таблиці

типу „Повинно містити” і „З чого може починатися”. В таблиці „Повинно містити” вказується, що саме необхідно визначити як знайдене на попередніх етапах синтаксичного або лексичного аналізу, якщо дана конкретна альтернатива дасть бажаний результат.

Таблиця „З чого може починатися” також є розріджений масив, в ньому вказуються ті символи і синтаксичні класи, які можуть зустрічатися на початку будь-якої альтернативи. Це значить, що альтернативу можна швидко переглянути, якщо перші декілька символів вказують на імовірність того, що дана альтернатива не дасть бажаного результату. Таблиця „З чого може починатися” однаково корисна і при зростаючому синтаксичному аналізі. Вона може бути створена автоматично із самого синтаксису так само, як і таблиця „Повинно містити”.

Побудова синтаксичного аналізатора:

Необхідно побудувати синтаксичний аналізатор виразів, який буде отримувати в якості вхідних даних коди натиснутих клавіш на касовому апараті, ґрунтуючись на тому припущенні, що вони складають закінчену операцію. В якості вихідного результату він буде виводити наступне:

- Опис форми виразу (його синтаксису) або повідомлення про помилку;
- Запис в базу даних про операцію: сума продажу товару, його кількість, код та штрих-код.

Крім того, до обробки помилок, узгодженню в часі та точності виконуваних операцій синтаксичного аналізатора висовуються наступні вимоги:

- Обробка помилок.
- Узгодження в часі.
- Точність.

Розглянемо правила, які описують синтаксис (послідовність кодів клавіш, що може прийти з ЕРРО) і процес синтаксичного аналізу (з’ясування

змісту цих послідовностей).

Сформуємо синтаксичну нотацію. Використаємо правила, що описують синтаксис, а також смислові описи різноманітних правил. Привила синтаксису описані в нормальній формі.

Оператор * значить натискання клавіші *, оператор *L* значить натискання клавіші “АН / ЦІНА”, а оператор *I* означає натискання клавіші “КОД ТОВ.”, згідно протоколу обміну касового апарата з ПК.

Така нотація (запис) називається нормальною формою або формою Бекуса-Наура (БНФ – Backus-Naur Form). Вона вперше була використана при описі мови програмування Algol 60. БНФ допускає декілька різноманітних способів запису, однак в дипломному проекті використаємо нотацію, що дещо відрізняється від тої, яка використовується при визначенні K&R (визначення, що застосовувалось при визначенні мови програмування С Керніканом і Річі). Синтаксичне позначення використано в дипломному проекті є більш компактним, ніж позначення БНФ, що застосовується в офіційному визначенні (K&R). Кожне правило синтаксису має своє ім’я. Визначене ім'я записується першим в стрічці і відділяється від свого визначення вдома двокрапками та знаком дорівнює (::=). Далі в стрічці вказується альтернатива. Кожна альтернатива складається із послідовності елементів, які в свою чергу, є або символом (кодом клавіші, що натискається на ЕРРО), або іменем (це означає, що в вхідних даних повинен бути присутнім деякий опис аналізованого елементу). Альтернативи розділяються вертикальною рискою (|), і розміщуються в одній стрічці, або якщо не вдається їх так розмістити, то вони займають декілька стрічок. Кожне правило синтаксису завершується крапкою з комою (;).

Немає великої різниці в тому яку з форм запису БНФ використовувати. Будь-яка з них повинна відображати посилання на інші описи синтаксису, термінальні символи та альтернативи. Застосовувана компактна форма БНФ виявляється набагато ближче до початкової форми,

що використовувалась в звіті по мові програмування Algol 60 (Algol 60 Report), так що для її застосування існують історичні обґрунтування.

Синтаксис роботи електронного реєстратора розрахункових операцій міститься в окремому файлі “syntax.lex”, вміст якого приведений в додатку А.

Клас one_digit визначає одну цифру, натиснуту на клавіатурі касового апарата. Сукупність класів one_digit (до тринадцяти в групі) утворюють штрих-код, або код товару (клас kod). Клас kil визначає кількість товару, що продається; вона є натуральним числом з трьома цифрами після коми.

Послідовність синтаксичних класів one_digit складає також клас price, який є ціною товару, що продається. Ціна товару на ЕРРО набирається в копійках, тобто кому набирати не потрібно.

Синтаксичні класи register_kil, register_price, kod утворюють клас реєстрації одного товару - register_towar_type. Описати величину націнки в грошових одиницях, націнки у відсотках, скидання в грошових одиницях, скидання у відсотках дозволяють синтаксичні класи nazinka_abs, nazinka_widn, skidannja_abs, skidannja_widn.

Обробити помилки, визначити реєстрації товарів, що необхідно проігнорувати, відмови товарів дозволяють синтаксичні класи ignore_error_liter, ignore_error_one_of_all, ignore_error_all_of_all, error_body, widmowa, register_widmowa_error, anul_oper. Визначити анулювання всього чеку дозволяє клас anul_check.

На вершині ієрархії синтаксичних класів знаходиться синтаксичний клас job, який описує будь-яку поведінку касового апарата. Його задача полягає на основі визначених класів визначити реєстрації товарів та ігнорувати операції, які знаходяться до реєстрації, і які не описані синтаксичними класами. Це дозволяє визначити синтаксичному аналізатору операції реалізації товарів на ЕРРО навіть якщо до цього виконувалось декілька операцій, що не мають відношення до продажу (виконання звітів на

ЕРРО та деякі інші операції).

Формування результатів синтаксичного аналізу:

Семантика, що реалізована в дипломному проекті полягає у виконанні операції вставки записів в таблицю бази даних про реалізований через ЕРРО товар або послугу.

В результаті зчитування синтаксису створюється декілька відповідних структур. Кожне правило синтаксису дозволяє отримати структуру наступного вигляду:

Вказівники на значення NULL, вказують на кінець списку. Крім того при зчитуванні синтаксису проглядаються структури для заповнення наступних таблиць:

Таблиця синтаксичних імен. Кожному визначенню синтаксису із вхідного синтаксису в цій таблиці відповідає одна позиція. В кожній позиції записується ім'я синтаксичної одиниці, вказівник на структуру синтаксичного класу, порядковий номер. В процесі побудови таблиці використовується ряд флагів. Індекс (номер) кожної позиції таблиці є номером, що в подальшому використовується для звертання до даного визначення синтаксису. Ця таблиця складається з LEXEMLIMIT елементів, де LEXEMLIMIT – це максимальне число визначень синтаксису, що підтримується даною програмою.

Для зчитування синтаксису використовується функція InstallLex з класу LexReader, основна задача якого є формування внутрішніх структур до початку роботи термінального сервера. Клас InstallLex також виконує розбір буферу кодів клавіш натиснутих на ЕРРО в залежності від синтаксису і формує двозв’язний список результатів синтаксичного аналізу (список проаналізованих синтаксичних класів).

Клас RunOperation розроблений для формування результатів синтаксичного аналізу (виконання функцій обробки із двозв’язного списку проаналізованих синтаксичних класів). RunOperation містить самі функції

обробки синтаксичних класів, що використані у файлі синтаксису syntax.lex.

Клас RunOperation має стільки екземплярів, на скільки ЕККА розрахований термінальний сервер. В даному випадку чотири екземпляри. Кожен об’єкт класу RunOperation пов'язаний з екземпляром класу InstallLex, який для них, в даному випадку, є спільним і є репозитарієм синтаксису ЕРРО типу MINI600ME.

4 НАУКОВО-ДОСЛІДНИЦЬКА

4.1 Дослідження багатозадачності в термінальному сервері

Термінальний касовий сервер ґрунтується на багатозадачних принципах сучасних операційних систем.

Обробка повідомлень є невід'ємною частиною будь-якого Windows-додатка. 32-розрядна платформа в силу своєї структури надає розробникам нові можливості.

Ці можливості – можливості багатозадачності. Багатозадачність в термінальному сервері потрібна для того, щоб кілька ділянок коду виконувалися одночасно при роботі з декількома касовими апаратами. Ці дії виконуються у фоновому режимі, на протязі яких, програма продовжує реагувати на дії користувача. Для взаємодії з кожним касовим апаратом використовується окремий потік.

Поняття процеси і потоки необхідні для того, щоб описати принципи багатозадачності на 32-розрядній платформі Windows. Процесом (process) називається екземпляр програми, що завантажений в пам'ять. Цей екземпляр може створювати потоки (thread), що являють собою послідовність інструкцій на виконання. Виконуються не процеси, а саме потоки. Причому будь-який процес має хоча б один потік. Цей потік називається головним (основним) потоком додатка.

Тому що практично завжди потоків набагато більше, ніж фізичних процесорів для їхнього виконання, то потоки насправді виконуються не одночасно, а по черзі. Але переключення між ними відбувається так часто, що здається начебто вони виконуються паралельно.

У залежності від ситуації потоки можуть знаходитися в трьох станах. По-перше, потік може виконуватися, коли йому виділений процесорний час, тобто він може знаходитися в стані активності. По-друге, він може бути

неактивним і очікувати виділення процесора, тобто бути в стані готовності. По-третє, потік може бути в стані блокування. Коли потік заблокований, йому взагалі не виділяється час. Звичайно блокування ставиться на час чекання якої-небудь події. При виникненні цієї події потік автоматично переводиться зі стану блокування в стан готовності. Якщо один потік використовує певний ресурс, а інший чекає звільнення доступу до цього ж ресурсу. Другий міг би використовувати цикл очікування, але під час виконання цього циклу процесор зайнятий на 100% (це називається активним чеканням). Таких циклів уникаємо при побудові термінального касового сервера., в чому нам надає допомогу механізм блокування. Другий потік може заблокувати себе доти, поки перший не встановить подію, що сигналізує про звільнення ресурсу.

У системі виділяються два види потоків – інтерактивні, що повторюють свій цикл обробки повідомлень (такі, як головний потік додатка), і робітники, що представляють собою просту функцію. В другому випадку потік завершується після завершення виконання цієї функції.

Слід приділити увагу також способу організації черговості потоків. Можна було б обробляти всі потоки по черзі, але такий спосіб далеко не найефективніший. Набагато ефективніше ранжувати усі потоки по пріоритетах. Пріоритет потоку позначається числом від 0 до 31, і визначається виходячи з пріоритету процесу, що породив потік, і відносного пріоритету самого потоку. Таким чином, досягається найбільша гнучкість, і кожен потік в ідеалі одержує стільки часу, скільки йому необхідно.

Іноді пріоритет потоку може змінюватися динамічно. Так інтерактивні потоки, що мають звичайно клас пріоритету Normal, система обробляє по-іншому і дещо підвищує фактичний пріоритет таких потоків, коли процес, що їх породив, знаходиться на передньому плані (foreground). Це зроблено для того, щоб додаток, з яким у даний момент працює користувач, швидше реагував на його дії.

У Windows виконуються не процеси, а потоки. При створенні процесу автоматично створюється його основний потік. Цей потік у процесі виконання може створювати нові потоки, що, у свою чергу, теж можуть створювати потоки і т.д. Процесорний час розподіляється саме між потоками, і виходить, що кожен потік працює незалежно.

Усі потоки, що належать одному процесові, розділяють деякі загальні ресурси – такі, як адресний простір оперативної пам'яті або відкриті файли. Ці ресурси належать усьому процесові, а виходить, і кожному його потокові. Отже, кожен потік може працювати з цими ресурсами без яких-небудь обмежень.

В Windows реалізована витісняюча багатозадачность – у будь-який момент система може перервати виконання одного потоку і передати керування іншому.. Завжди, коли два або більше потоків використовують який-небудь загальний ресурс, виникає проблема сумісного доступу до ресурсу. Це відбудеться коли один потік ще не закінчив працювати з яким-небудь загальним ресурсом, а система переключилася на інший потік, що використовує той же ресурс. Результат роботи цих потоків може значно відрізнятися від задуманого. Такі конфлікти можуть виникнути і між потоками, що належать різним процесам.

Саме тому необхідний механізм, що дозволяє потокам погоджувати свою роботу із загальними ресурсами. Цей механізм одержав назву механізму синхронізації потоків (thread synchronization).

Для керуванням потоком, що працює з певним реєстратором розрахункових операцій в розроблюваному термінальному сервері використовується функція process_rro. На цю функцію покладено ряд задач:

- керування зв’язком з ЕРРО (встановлення та підтримка зв’язку);
- перевірка копії термінального сервера щодо легальності;
- керування програмуванням товарів та послуг в ЕРРО;

- виконання термінального ехо-режиму.

Під ехо-режимом розуміється така робота касового сервера при якій він повторює кожен код клавіші, що надходить з електронного реєстратора розрахункових операцій, відсилаючи його назад. Касовий апарат сприймає коди клавіш, що були отримані від ПК, так ніби вони були натиснуті на клавіатурі ЕРРО.

Функція, що реалізовує ехо-режим, виконує окрім повторення кодів клавіш накопичення їх в буфері до тих пір, поки не зустрінеться код клавіші ОПЛАТА. Після його отримання буфер аналізується синтаксичним аналізатором та очищається.

Блок-схеми функцій головного потоку програми, потоків касових апаратів, ехо-режиму зображені в графічній частині дипломного проекту.

5. СПЕЦІАЛЬНА ЧАСТИНА

5.1 Особливості икористання мови С++

Мова програмування служить двом зв'язаним між собою цілям: вона дає програмісту апарат для завдання дій, що повинні бути виконані, і формує концепції, якими користується програміст, міркуючи про те, що робити. Першій меті ідеально відповідає мова, що настільки "близька до машини", що всіма основними машинними аспектами можна легко і просто оперувати досить очевидним для програміста образом. Другій меті ідеально відповідає мова, що настільки "близька до розв'язуваної задачі", що концепції її рішення можна було виражати прямо і коротко. З таким наміром попередньо задумувалися засоби, додані до С для створення С++.

Зв'язок між мовою, на якій ми думаємо/програмуємо, і задачами і рішеннями, що ми можемо представляти у своїй уяві, дуже близька. З цієї причини обмежувати властивості мови тільки цілями виключення помилок програміста в кращому випадку небезпечно. Як і у випадку з природними мовами, є величезна користь бути принаймні двомовним. Мова надає програмісту набір концептуальних інструментів; якщо вони не відповідають задачі, то їх просто ігнорують. Наприклад, серйозні обмеження концепції покажчика змушують програміста застосовувати вектора і цілу арифметику, щоб реалізувати структури, покажчики і т.п. Гарне проектування і відсутність помилок не може гарантуватися тільки за рахунок мовних засобів.

С++ – універсальна мова програмування, задумана так, щоб зробити програмування більш приємним для серйозного програміста. Крім можливостей, що дає С, С++ надає гнучкі й ефективні засоби визначення нових

типів. Використовуючи визначення нових типів, що точно відповідають концепціям додатка, програміст може розділяти розроблювальну програму на частини, що легко піддаються контролю. Такий метод побудови програм часто називають абстракцією даних. Інформація про типи міститься в деяких об'єктах типів, визначених користувачем. Такі об'єкти прості і надійні у використанні в тих ситуаціях, коли їхній тип не можна установити на стадії компіляції. Програмування з застосуванням таких об'єктів часто називають об’єктно-орієнтованим. При правильному використанні цей метод дає більш короткі, більш зрозумілі і легше контрольовані програми.

Ключовим поняттям С++ є клас. Клас - це тип, обумовлений користувачем. Класи забезпечують приховання даних, гарантовану ініціалізацію даних, неявне перетворення типів для типів, визначених користувачем, динамічне завдання типу, контрольоване користувачем керування пам'яттю і механізми перевантаження операцій. С++ надає набагато кращі, чим у С, засобу вираження модульності програми і перевірки типів. У мові є також удосконалення, не зв'язані безпосередньо з класами, що включають у себе символічні константи, inline- підстановку функцій, параметри функції за замовчуванням, перевантажені імена функцій, операції керування вільною пам'яттю і посилальний тип. У С++ збережені можливості мови С по роботі з основними об'єктами апаратного забезпечення (біти, байти, слова, адреси і т.п.). Це дозволяє дуже ефективно реалізовувати типи, обумовлені користувачем.

С++ і його стандартні бібліотеки спроектовані так, щоб забезпечувати переносимість. Наявна на сучасний момент реалізація мови буде йти в більшості систем, що підтримують С. З С++ програм можна використовувати С бібліотеки, і з С++ можна використовувати велику частину інструментальних засобів, що підтримують програмування на С.

5.2 Ефективність і структура C++

C++ була розвинута з мови програмування C і деякими виключеннями зберігає C як підмножину. Базова мова, C підмножина C++, спроектована так, що є дуже близька відповідність між її типами, операціями й операторами і комп'ютерними об'єктами, з якими безпосередньо приходиться мати справу: числами, символами й адресами. За винятком операцій вільної пам'яті new і delete, окремі вирази й оператори C++ звичайно не потребують схованої підтримки під час виконання чи підпрограмах.

У C++ використовуються ті ж послідовності виклику і повернення з функцій, що й у C. У тих випадках, коли навіть цей досить ефективний механізм є занадто дорогим, C++ функція може бути підставлена inline, задовольняючи, таким чином, угоді про запис функцій без додаткових витрат часу виконання.

Одним з первісних призначень C було застосування її замість програмування на асемблері в самих насущних задачах системного програмування. Коли проектувалася C++, були прийняті міри, щоб не ставити під погрозу успіхи в цій області. Розходження між C і C++ виявляється в першу чергу в кількості уваги, що приділяється типам і структурам. C виразна і поблажлива. C++ ще більш виразна, але щоб досягти цієї виразності, програміст повинний приділити більше уваги типам об'єктів. Коли відомі типи об'єктів, компілятор може правильно обробляти вирази, тоді як у противному випадку програмісту довелося б задавати дії з болісними подробицями. Знання типів об'єктів також дозволяє компілятору виявляти помилки, що у противному випадку залишилися б до тестування. Використання системи типів для того, щоб одержати перевірку параметрів функцій, захистити дані від випадкової зміни, задати нові операції і т.д., саме по собі не збільшує витрат на часом виконання і використання пам'яті.

Особлива увага, що приділена при розробці C++ структурі, відбилася на зростанні масштабу програм, написаних з часу розробки C. Маленьку програму (менше 1000 рядків) можна змусити працювати за допомогою грубої сили, навіть порушуючи всі правила гарного стилю. Для програм великих розмірів це не зовсім так. Якщо програма в 10 000 рядків має погану структуру, то можна помітити, що нові помилки з'являються так само швидко, як зникають старі. C++ була розроблена так, щоб дати можливість розумним чином структурувати великі програми таким чином, щоб для однієї людини не було непомірним справлятися з програмами в 25 000 рядків. Існують програми набагато великих розмірів, однак ті, котрі працюють, у цілому, як виявляється, складаються з великого числа майже незалежних частин, кожна з який набагато нижче зазначених меж.

Не кожна частина програми, може бути добре структурована, незалежна від апаратного забезпечення, легко читається і т.п. C++ має можливості, призначені для того, щоб безпосередньо й ефективно працювати з апаратними засобами, не турбуючись про безпеку чи простоту розуміння. Вона також має можливості, що дозволяють ховати такі програми за елегантними і надійними інтерфейсами.

5.3 Реалізація моделі об’єктно-орієнтованого програмування в C++

Об'єкт – це абстрактна сутність, наділена характеристиками об'єктів навколишнього нас реального світу. Програма звичайно використовує змінні для збереження інформації про різні реально існуючі сутності, наприклад про службовців, книги і навіть файли. При об’єктно-орієнтованому програмуванні увага зосереджується на предметах, що утворять систему, і операціях, що повинні виконувати над цими предметами. Наприклад, для об'єкта-файлу можна мати операції, що друкують, відображають чи змінюють вміст файлу. У C++ використовується клас для визначення своїх

об'єктів. Мета полягає в тому, щоб включити в клас стільки інформації про об'єкт, скільки потрібно. Виходячи з цього, можна підібрати клас, створений для однієї програми, і використовувати його в декількох різних програмах.

Створення об'єктів і маніпулювання ними – це зовсім не привілей мови C++, а скоріше результат методології програмування, що втілює в кодових конструкціях опису об'єктів і операції над ними. Кожен об'єкт програми, як і будь-який реальний об'єкт, відрізняється власними атрибутами і характерним поводженням. Об'єкти можна класифікувати по різних категоріях.

Кожен клас займає визначене місце в ієрархії класів. Таким чином, будь-який клас визначає деяку категорію об'єктів, а всякий об'єкт є екземпляр деякого класу.

Об’єктно-орієнтоване програмування (ООП) — це методика, що концентрує основну увагу програміста на зв'язках між об'єктами, а не на деталях їхньої реалізації. Основними принципами ООП є інкапсуляція, успадкування, поліморфізм, створення класів і об'єктів.

Інкапсуляція є об'єднання в єдиному об'єкті даних і коду, що оперує з цими даними. У термінології ООП дані називаються членами даних або записами об’єкта, а код – методами об'єктними чи функціями-членами.

Інкапсуляція дозволяє в максимальному ступені ізолювати об'єкт від зовнішнього оточення. Вона істотно підвищує надійність розроблювальних програм, тому що локалізовані в об'єкті функції обмінюються з програмою порівняно невеликими обсягами даних, причому кількість і тип цих даних звичайно ретельно контролюються. У результаті заміна чи модифікація функцій і даних, інкапсульованих в об'єкт, як правило, не спричиняє наслідків, що погано прослідкуються, для програми в цілому (з метою підвищення захищеності програм в ООП майже не використовуються глобальні змінні).

Іншим наслідком інкапсуляції є легкість обміну об'єктами, переносу їх з однієї програми в іншу.

Однією з особливостей інкапсуляції є перевантаження методів класу. Говорять, що метод перевантажений, якщо він асоціюється з більш ніж однією однойменною функцією, тобто визначає декілька реалізацій методів, що відрізняються між собою типами і кількістю параметрів та типом результату, але мають однакове ім’я.

Однієї із самих чудових особливостей живої природи є її здатність породжувати потомство, що володіє характеристиками, подібними з характеристиками попереднього покоління. Запозичена в природи ідея спадкування вирішує проблему модифікації поводження об'єктів і додає ООП виняткову силу і гнучкість. Спадкування дозволяє, практично без обмежень, послідовно будувати і розширювати класи. Починаючи з найпростіших класів, можна створювати похідні класи по зростаючій складності, що не тільки легкі в налагодженні, але і прості за внутрішньою структурою.

Послідовне проведення в життя принципу спадкування, особливо при розробці великих програмних проектів, добре узгоджується з технікою спадного структурного програмування (від загального до частки), і багато в чому стимулює такий підхід. При цьому складність коду програми в цілому істотно скорочується. Похідний клас (нащадок) успадковує усі властивості, методи і події свого базового класу (батька) і всіх його попередників в ієрархії класів.

При спадкуванні базовий клас обростає новими атрибутами й операціями. У похідному класі звичайно з'являються нові члени даних, властивості і методи. При роботі з об'єктами звичайно підбирають найбільш придатний клас для рішення конкретної задачі і створюють одного чи декількох нащадків від нього, що здобувають здатність робити не тільки те, що закладено в батьку. Дружні функції дозволяють похідному класу одержати доступ до всіх членів даних зовнішніх класів.

Крім того, похідний клас може перевизначати наслідувані методи, тобто визначати для них іншу реалізацію, в тому випадку, коли їхня робота в базовому класі не підходить нащадку.

Слово поліморфізм від грецьких слів *poly* (багато) і *morphos* (форма) означає множинність форм. Поліморфізм – це властивість родинних об'єктів (тобто об'єктів, класи яких є похідними від одного батька) поводитися по-різному в залежності від ситуації, що виникає в момент виконання програми. У рамках ООП програміст може впливати на поводження об'єкта тільки побічно, змінюючи вхідні в нього методи і додаючи нащадкам відсутні в батька специфічні властивості.

Для зміни методу необхідно перевантажити його в нащадку, тобто оголосити в нащадку однойменний метод і реалізувати в ньому потрібні дії. У результаті в об'єкті-батьку й об'єкті-нащадку будуть діяти два однойменних методи, що мають різну кодову реалізацію і, отже, що додають об'єктам різне поводження. Наприклад, в ієрархії родинних класів геометричних фігур (крапка, пряма лінія, квадрат, прямокутник, окружність, еліпс і т.д.) кожен клас має метод Draw, що відповідає за належний відгук на подію з вимогою намалювати цю фігуру.

Завдяки поліморфізму, нащадки можуть перевантажувати загальні методи батька для того, щоб реагувати специфічним образом на те саму подію.

Клас не має фізичної сутності, його найближчою аналогією є оголошення структури. Пам'ять виділяється тільки тоді, коли клас використовується для створення об'єкта. Цей процес також називається створенням *екземпляра класу (class instance).*

Звичайно, об'єкт знаходиться в деякому унікальному стані, обумовленому поточними значеннями його атрибутів. Функціональність об'єктного класу визначається можливими операціями над екземпляром цього класу.

Визначення класу в мові C++ містить інкапсуляцію членів даних і методів, що оперують із членами даних і визначають поводження об'єкта, тобто говорячи термінології ООП, методи модифікують стан об'єкта шляхом зміни членів даних.

5.4 Вибір компілятора для реалізації проекту

У зв'язку з тим, що сьогодні рівень складності програмного забезпечення дуже високий, розробка додатків Windows з використанням тільки якої-небудь мови програмування значно ускладнюється. Потрібно затратити масу часу на рішення стандартних задач по створенню багатовіконного інтерфейсу. Реалізація технології зв'язування і вбудовування об'єктів-OLE – вимагає ще більш складної роботи.

Щоб полегшити роботу практично всі сучасні компілятори з мови C++ містять спеціальні бібліотеки класів. Такі бібліотеки містять у собі практично весь програмний інтерфейс Windows і дозволяють користуватися при програмуванні засобами більш високого рівня, чим звичайні виклики функцій. За рахунок цього значно спрощується розробка додатків, що мають складний інтерфейс користувача, полегшується підтримка технології OLE і взаємодія з базами даних.

C++Builder призначений для швидкої розробки додатків (RAD), побудованих на сучасному фундаменті об’єктно-орієнтованого програмування (ООП). C++Builder у корені змінює процес розробки: наскільки легше і швидше можна одержувати працюючі і надійні програми для операційних систем Windows, чим при використанні традиційних інтерфейсних оболонок інших систем.

5.5 Розробка програмного забезпечення для роботи з штрих-кодами

Для кодування товарів із змінною кількістю або вагою використовують штрих-коди обмеженої циркуляції у яких разом з кодом товару у деяких розрядах закодована вага або вартість товару. Касовий апарат не може працювати з такими кодами на пряму. Касовий сервер може обробляти такі штрих-коди. Для цього формується послідовність клавіш, які необхідно натиснути на клавіатурі касового апарату для здійснення реєстрації товару з кодом та кількістю (вагою), що вказана у ваговому штрих-коді. Ця послідовність відсилається касовому апарату у відповідь на зчитування вагового штрих-коду. ЕРРО сприймає цю послідовність і виконує реєстрацію товару.

В розробленому комплексі обліку реалізації товарів використана наступна схема вагових штрих-кодів (табл. 5.1):

Таблиця 5.1 – Призначення розрядів вагових штрих-кодів

Розряди	Зміст
0-1	Ідентифікатор коду (рівні “0” “2”)
2-6	Код товару
7-11	Вага (кількість) товару в грамах
12	Контрольний розряд

Якщо буде зчитаний штрих-код, що підходить під такий шаблон, то касовий сервер виконає реєстрацію товару з кодом, що міститься в розрядах 2-6, та кількістю, що міститься в розрядах 7-11. Штрих-коди, що не мають префіксу „02”, не сприймаються як вагові і передаються безпосередньо в

ЕРРО. Ці штрих-коди товарів можуть бути як внутрішніми, роздрукованими на клейкому папері за допомогою розробленого програмного забезпечення, так і штрих-кодами від виробника продукту.

Для обробки вагових штрих-кодів в функції ехо-повторювача використовується наступний код:

```
     if(s.length() == 14 && s[0] == '2' && s[1] == '2'){
        char resstr[30];
        sprintf(resstr,"K%.3f*%dI",StrToInt(
s.substr(7,5).c_str())/1000.0,StrToInt(s.substr(2,5).c_
str()));
        ma[i]->sourse += s = resstr;
     }
```

Для повної автоматизації торгівельних операцій всі товари повинні містити штрих-кодові етикетки. Товари, що не мають штрих-кодові етикетки від виробника, постачаються етикетками з внутрішніми штрих-кодами. Для друку штрих-кодових етикеток розроблена програма, яка використовує таблицю бази даних формату dBaseIII, що створюється програмним обліковим забезпеченням. Формування зображення штрих-кодових етикеток відбувається в Microsoft Word. Вигляд програми зображений в графічній частині дипломного проекту.

5.6 Робочі файли термінального сервера

Термінальний касовий сервер та програмне забезпечення для друку штрих-кодових етикеток використовує інтерфейс з обліковим програмним забезпеченням на рівні файлів формату dBaseIII.

При необхідності програмування товарів в ЕРРО облікове програмне забезпечення створює файл формату dBaseIII в папці *in* робочого каталогу

термінального сервера. Розроблений термінальний сервер зчитує параметри товарів та виконує програмування відповідного реєстратора розрахункових операцій. Структура файлу приведена в таблиці 5.2.

Таблиця 5.2 – Прайс-лист товарів та послуг

Поле	Тип	Розмір	Значення
Code	N	4	Код товару
Bar	C	13	Штрих-код товару
Name	C	20	Назва товару
Price	N	9.2	Ціна товару
Amount	N	7.3	Кількість (приріст) товару
Dep	N	1	Номер відділу (прив’язка до номеру відділу, а через нього до ПДВ)
Nalog	N	1	Номер податкової групи (не використовується)
Ecr	C	2	Номер ККМ, на який дозволений продаж товару (не використовується)
Check_amnt	N	1	Флаг контролю кількості (не використовується)
Piece	N	1	Флаг штучного товару (не використовується)
Disc	N	1	Флаг запису о дисконтній картці (не використовується)

При реалізації товарів записи про видані чеки робляться в журналі продажів, який є файлом формату dBaseIII. Структура файлу приведена в таблиці 5.3.

Таблиця 5.3 – Журнал продажів

Поле	Тип	Розмір	Значення
Code	N	4	Код товару
Price	N	9.2	Ціна товару
Weight	N	7.3	Кількість
Oper	N	2	Код операції (1=возврат, 2=продажа) (не використовується)
Kassa	N	2	Номер каси
Date1	D	8	Дата операції
Time1	C	8	Час операції
Bar	C	13	Штрих-код товару
Dep	N	1	Номер секції
Disc	N	9.2	Сума скидки на операцію (не використовується)
Card	N	8	Номер дисконтної картки, по якій проведене скидання (не використовується)
Rec	N	4	Номер чеку (наскрізна нумерація в межах поточної дати)
Z	N	4	Номер поточної зміни (не використовується)

Термінальний сервер веде внутрішню таблицю для кожного ЕРРО, в якій міститься інформація про параметри закодованих товарів та обороти в грошах та кількості.

При реалізації товарів у внутрішній таблиці фіксується збільшення оборотів в грошах та кількості по товарам. При необхідності виконання звіту (503-ого) внутрішня таблиця копіюється в вихідну папку термінального

сервера з назвою, що вказує дату та час операції. Внутрішня таблиця є файлом формату dBaseIII. Структура файлу приведена в таблиці 5.4.

Таблиця 5.4 – Структура внутрішньої таблиці та таблиці аналогу звіту

Поле	Тип	Розмір	Значення
Code	N	4	Код товару
Bar	C	13	Штрих-код товару
Name	C	20	Назва товару
Price	N	9.2	Ціна товару
Amount	N	7.3	Кількість (приріст) товару
Dep	N	1	Номер відділу (прив’язка до номеру відділу, а через нього до ПДВ)
Nalog	N	1	Номер податкової групи (не використовується)
Ecr	C	2	Номер ККМ, на який дозволений продаж товару (не використовується)
Check_amnt	N	1	Флаг контролю кількості (не використовується)
Piece	N	1	Флаг штучного товару (не використовується)
Disc	N	1	Флаг запису о дисконтній картці (не використовується)
Suma	N	9.2	Сума продажів
Kil	N	7.3	Кількість продажів
Amount1	N	7.3	Залишок (наявність)

6 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ

6.1 Організація охорони праці при роботі з системою управління

Охорона праці розглядає проблеми забезпечення здорових і безпечних умов праці. Виявляє і вивчає можливі причини нещасних випадків, професійних захворювань, аварій, вибухів, пожеж і розробляє систему заходів і вимог з метою виключення цих причин і створення безпечних і сприятливих для людини умов праці.

Завдання охорони праці є зведення до мінімуму імовірності пошкодження або захворювання працівників з одночасним забезпеченням комфорту при максимальній продуктивності праці.

Навчання працівників безпеці праці проводять відповідно до вимог ГОСТ 12. 0.004 - 79, який встановлює порядок і види навчання. На всіх підприємствах і в організаціях незалежно від характеру і ступеню небезпеки виробництва навчання працівників проводять при підготовці нових робітників, проведенні різноманітних видів інструктажів і підвищенні кваліфікації.

Контроль за своєчасним і якісним навчанням виконує відділ охорони праці чи інженер з охорони праці, або ІТП, на якого наказом керівника підприємства покладено ці обов'язки. Ті, що вперше поступають на роботу, навчання проходять згідно з "Типовим положенням про підготовку і підвищення кваліфікації робітників". В журналі обліку навчальної роботи реєструють навчальну тему, за якою проводилось навчання.

Інструктаж працюючих поділяють на вступний, початковий, на робочому місці, повторний, позаплановий і початковий.

Вступний інструктаж з усіма, хто поступає на роботу незалежно від їх освіти і стажу роботи по даній професії, проводить інженер з охорони праці за програмою, затвердженою головним інженером підприємства, про проведення вступного інструктажу з обов'язковим підписом того, хто проводив інструктаж і того, хто його отримував.

Початковий інструктаж на робочому місці, повторний, позаплановий і поточний проводить керівник робіт.

Початковий інструктаж на робочому місці проводять при прийомі на роботу нових робітників за інструкцією з охорони праці, розробленою для окремих професій або видів робіт. Всі робітники після цього інструктажу і перевірки знань 2-5 змін (залежно від навичок і стажу роботи) працюють під наглядом бригадира чи майстра, потім оформляється допуск до їх самостійної праці.

Повторний інструктаж проходять всі працівники незалежно від кваліфікації, освіти і стажу роботи через три місяці. Його проводять з метою перевірки знання робітниками правил і норм з охорони праці.

Позаплановий інструктаж проводять коли змінилися правила охорони праці або технологічний процес, обладнання, інструмент та інші фактори, що впливають на безпеку праці; коли працівники порушують правила охорони праці, що можуть призвести чи призвели до травм, аварій чи пожежі, вибуху. Його проводять індивідуально чи з групою робітників однієї професії за програмою початкового інструктажу на робочому місці. При його реєстрації вказують причину, яка спричинила його проведення.

Умови праці мають велике значення практично для всіх виробничих показників - продуктивності праці, якості робіт, безпеки працівників та інше.

Санітарно-гігієнічні умови праці характеризуються показниками виробничого середовища - рівнем освітлення, мікрокліматичними

параметрами, загазованістю і запиленістю повітряного середовища, рівнем шуму і вібрації, наявністю іонізуючого випромінювання та інше.

6.2 Електробезпека

Електричні установки, з якими доводиться мати справу практично всім працюючим по встановленню та налагодженню засобів автоматизації, виявляють для людини велику потенційну небезпеку, яка збільшується у зв'язку з тим, що органи чуття людини не можуть на відстані виявити присутність електричної напруги на обладнанні.

Степінь ураження електричним струмом залежить від цілого ряду факторів: значення сили струму, електричного опору тіла людини та тривалості протікання через неї струму, виду та частоти струму, індивідуальних властивостей людини та умов навколишнього середовища.

Конструкція електроустановок має відповідати умовам їх експлуатації та забезпечувати захист персоналу від дотику з струмоведучими та рухомими частинами, а обладнання - від попадання всередину посторонніх твердих тіл та води.

Конструкція, вид виконання, спосіб встановлення, клас ізоляції застосовуваних провідників, кабелів, пристроїв та іншого електрообладнання відповідають вимогам електробезпеки. За ступенем ураження людей електричним струмом котельня відноситься згідно ПУЕ 1.1.13 до категорії приміщень з підвищеною небезпекою (висока температура, можливість одночасного дотику до металевих елементів технологічного обладнання або металоконструкцій будинку та металевих корпусів електрообладнання).

У нормальному режимі роботи обладнання - можливість ураження працівників електричним струмом виключена. Але на випадок аварії для запобігання ураження струмом людей передбачене захисне заземлення.

Згідно ПУЕ 1.7.65 допустимий опір заземлення повинен бути не більшим 10 Ом.

При виконанні монтажних робіт використовуються переносні електроінструменти (електродрилі, електрошліфувальні установки, тощо). Для забезпечення безпечної праці корпуси однофазних електроприймачів повинні занулюватись.

Захист людини від ураження електричним струмом в мережах з зануленням здійснюється тим, що при замиканні одної з фаз на занулений корпус в ланці цієї фази виникає струм короткого замикання, що діє на струмовий захист (плавкий запобіжник, автомат), в результаті чого відбувається відключення аварійної ділянки від мережі. Крім того, ще до спрацювання захисту струм короткого викликає перерозподіл напруги в мережі, що приводить до зниження напруги корпусу відносно землі. Таким чином, занулення зменшує напругу дотику та обмежує час, на протязі якого людина, що доторкнулася до корпусу, може попасти під дію напруги.

Для того, щоб забезпечити швидке (на протязі декількох секунд) відключення аварійної ділянки, струм короткого замикання повинен бути достатньо великим. Відповідно до вимог ПУЕ струм короткого замикання повинен не менше ніж в три рази перевищувати номінальний струм плавкої вставки найближчого запобіжника або номінальний струм нерегульованого розчеплювача автоматичного вимикача. При використанні автоматичних вимикачів, що мають тільки електромагнітний розчіплювач (відсічку), струм короткого замикання повинен перевищувати значення струму встановлення миттєвого спрацювання в 1,25-1,4 рази в залежності від номінального струму.

В однофазних електроприймачів, що включені між фазним та нульовим робочим проводами, занулення корпусів слід виконувати з допомогою окремого (третього) провідника, який повинен з'єднувати корпус електроприймача з нульовим захисним проводом. В таких випадках

під'єднувати корпуси електроприймачів для забезпечення електробезпеки до нульового робочого проводу недопустимо, оскільки при його розриві (перегоранні запобіжника) всі під'єднані до нього корпуси виявляться під фазною напругою відносно землі.

В мережі з зануленням недопустимо використовувати заземлення окремих електроприймачів, не під'єднавши їх перед цим до нульового захисного провідника. В цьому випадку при замиканні фази на заземлений, але не приєднаний до нульового захисного провідника корпус створюється коло струму через заземлення цього корпусу та заземлення нейтралі джерела струму. Такий випадок небезпечний, оскільки засоби захисту не зможуть відключити такий електроприймач через мале значення струму і тому небезпечна напруга на всіх корпусах може зберігатися тривалий період, поки заземлений приймач не буде відключений вручну.

Важливо відмітити, що якщо занулений корпус одночасно заземлений, то це тільки покращує умови безпеки, оскільки забезпечує додаткове заземлення нульового захисного проводу.

Для ізоляції людини від частин електроустановок, що знаходяться під напругою, використовуються основні та допоміжні ізолюючі засоби, а саме слюсарно-монтажний інструмент з ізольованими ручками, коврики, ізолюючі підставки, тощо.

У приміщеннях, де знаходяться вимірювальні прилади, необхідно забезпечити виконання заходів по боротьбі з статичною електрикою (тобто прилади повинні бути заземлені). Найпростішим засобом є підтримка відносної вологості повітря на рівні 50 - 60 % за допомогою побутового електрозволожувача.

Підлогу слід виконувати відповідно до ГОСТ 12.4.124-83, використовуючи антистатичне покриття на проходах і біля робочих місць.

Робітникам рекомендовано носити одежу з природних матеріалів або з комбінованих - природних і штучних волокон. Для зняття електростатичних зарядів з одежі слід використовувати антистатики побутового призначення.

Оскільки корпуси приладів виконані з металу, то для усунення небезпеки ураження людини електричним струмом (можливий пробій на корпус приладу) використовується захисне заземлення.

6.3 Розрахунок заземлення

Розрахуємо систему заземлення для електроустаткування, яке працює від напруги 220 В.

$$R_{заз} \le \frac{U}{I_p} = \frac{220}{66} = 3.3 \le 4 Ом$$

Визначаємо опір грунту: $\rho = \kappa_н * \rho_н$ = 2*200 = 400 Ом м,

де $\kappa_н$ - коефіцієнт підсилення;

$\rho_н$ — питомий опір грунту (вибирається з довідкової літератури).

Визначаємо опір одиночного вертикального заземлювача:

$$R_B = \frac{\rho}{2\pi l}(\ln\frac{21}{d} + \frac{1}{2} * \frac{4t+1}{4t-1})$$

де t - відстань від середини заземлювача до поверхні грунту, м;

l,d - довжина і діаметр стержня заземлювача, м;

R_B = 96 Ом.

Визначаємо опір сталевої полоси, що з'єднує стержневі заземлювачі:

$$R_П = (\rho / 2\pi l) * \ln(l^2 / dt) = 61 \text{Ом.}$$

Визначаємо орієнтовне чмсло стержневих заземлювачів:

$$n = R_B / [r_B]\eta_B = 96 / 4 * 1 = 24 \text{шт;}$$

r_B - допустимий по нормам опір заземляючого пристрою,

$\eta_в$- коефіцієнт використання вертикальних заземлювачів (для орієнтовного розрахунку приймається рівним 1).

Приймаємо розміщення вертикальних заземлювачів по контуру з відстанню між стальними заземлювачами рівним 21. З довідкової літератури визначаємо $\eta_в$= 0,66 і $\eta_г$ = 0,39.

Визначаємо необхідну кількість вертикальних заземлювачів

$$n = R_в /[r_в]\eta_в = 96/(4*0.66) = 36$$

Розраховуємо загальний розрахунковий опір аземлюючого пристрою R з врахуванням з'єднувальної полоси

$$R = R_в R_п /(R_в \eta_г + R_п \eta_в n) = 3.9 \text{ Ом.}$$

Розрахунок проведено правильно, оскільки виконується умова $R \leq [r_в]$.

Розрахунок штучного заземлення:

Приймаємо, що опір захисного заземлення не повинен перевищувати 4 Ом:

$$R_{ЗЗ} = \frac{R_c R_п}{R_c + R_п} \leq 4 Ом$$

де $R_{ЗЗ}$ – опір захисного заземлення;

R_c – опір стержневих заземлювачів;

$R_п$ - опір поперечних заземлювачів.

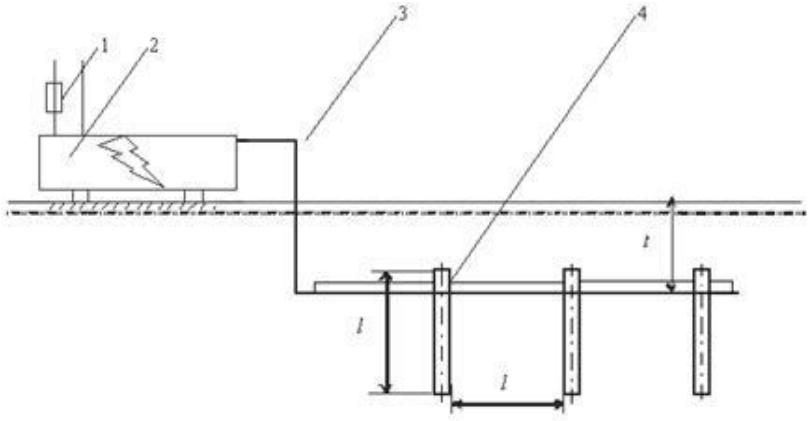


Рисунок .1 - Пристрій заземлення

4 – плавка вставка; 2 – електроустановка; 3 – з'єднувальна штаба; 4 – трубчатий заземлювач

Опір одиночного стержневого заземлювача розтіканню електричного струму:

$$R_{oc} = \frac{\rho_p}{2\pi l}(\ln\frac{2l}{d} + \ln\frac{4h'+l}{4h'-l})$$

де h – відстань від поверхні грунту до заземлювача і становить 0,8 м;

l – довжина стержневого заземлювача 3 м;

d – діаметр стержневого заземлювача 50 мм.

$$R_{oc} = \frac{750}{2\cdot 3.14\cdot 3}(\ln\frac{2\cdot 3}{0.05} + \ln\frac{4\cdot 0.8+3}{4\cdot 0.8-3}) = 39.8\cdot(0.18+3.43) = 143.8Ом$$

Опір одиночного поперечного заземлювача:

$$R_{on} = \frac{\rho_p}{2\pi l}\ln\frac{2l^2}{bh'}$$

де l – довжина поперечного заземлювача 2,5 м;

b – ширина полоси заземлювача 30 мм;

ρ_p - розрахунковий опір грунту: для поперечних електродів 1000 Ом·м, для стержневих електродів 750 Ом·м.

$$R_{on} = \frac{1000}{2\cdot 3{,}14\cdot 2{,}5}\ln\frac{2\cdot 2{,}5^2}{0{,}03\cdot 0{,}8} = 63.7\cdot 6{,}25 = 398.1Ом$$

В наслідок взаємовпливу вводимо коефіцієнт використання заземлювачів:

$$\eta = \frac{R_0}{nR_o}$$

де $R_д$ – допустимий опір заземлення, що становить 4 Ом;

R_o – опір одиночного заземлювача.

З цієї формули методом ітерацій підбирають n, при якому η =1:

n	**R_n**	**R_c**	**R_o**	**ŋ**
1	398,1	143,8	105,6	26,1
5	398,1	143,8	105,6	5,2
10	398,1	143,8	105,6	2,6
15	398,1	143,8	105,6	1,7
20	398,1	143,8	105,6	1,3
25	398,1	143,8	105,6	1,1
26	398,1	143,8	105,6	1,0
27	398,1	143,8	105,6	0,9

Отже приймаємо кількість одиночних заземлюючих електродів рівною 26.

ОСНОВНІ ВИСНОВКИ ДИПЛОМНОЇ РОБОТИ

В результаті проведеної роботи було проаналізовано основні параметри та чинники, які впливають на забезпечення комфорту отримання та збору інформації про потік товарів, а також їх ідентифікацію.

В рамках даної випускної кваліфікаційної роботи були отримані наступні результати:

- Розроблено архітектуру системи.
- Покращена оцінка потоку товарів.
- Отримання даних з bluetooth пристроїв виділено в бібліотеку.
- Реалізована система для інтеграції алгоритмів аналізу потоку товарів.
- Проведено тестування.

В роботі було розроблено та досліджено автоматизовану систему збору та аналізу управлінням потоків товарів на підприємствах.

БІБЛІОГРАФІЯ

1. David D. McManus Jinseok Lee Oscar Maitas-Nada Esa Rahul Pidikiti Alex Carlucci Josephine Harrington Eric Mick Ki H. Chon. A novel application for the detection of an irregular pulse using an iPhone 4S in patients with atrial fibrillation. - 2013.

2. Fan Xiangmin, Wang Jingtao. BayesHeart: A Probabilistic Approach for Robust, LowLatency Heart Rate Monitoring on Camera Phones. - 2015.

3. Improved heart rate detection using smart phone / Arpan Pal, Aishwarya Visvanathan, Anirban Dutta Choudhury, Aniruddha Sinha. – 2014.

4. Laure Denis, Paramonov Ilya. Improved Algorithm for Heart Rate Measurement Using Mobile Phone Camera. – 2013.

5. Lenskiy Artem A., Aitzhan Yerlan. Extracting Heart Rate Variability from a Smartphone Camera. - 2013.

6. A Novel Method to Detect Heart Beat Rate Using a Mobile Phone / Pelegris P., Banitsas K., Orbach T., Marias K. - 2010 року.

7. A Novel Method to Detect Heart Beat Rate Using a Mobile Phone / Arpan Pal, Aniruddha Sinha, Anirban Dutta Choudhury et al. – 2013.

8. Real time heart rate variability assessment from Android smartphone camera photoplethysmography: Postural and device influences / F. Guede-Fernandez, V. Ferrer-Mileo, J. Ramos-Castro et al.- 2015.

9. Rong-Chao Peng Xiao-Lin Zhou Wan-Hua Lin Yuan-Ting Zhang. Extraction of Heart Rate Variability from Smartphone Photoplethysmograms. – 2015.

10. Vikram Chandrasekaran BE Measuring Vital Sings Using Smart Phones. – 2010 року.

11. Yuriy Kurylyak Francesco Lamonaca Domenico Grimaldi. Smartphone-Based Photoplethysmogram Measurement. - 2012.

12. Филлипс Б., Стюарт К., Марсикано К. Android. Программирование для профессионалов. 3-е изд. -2017.

13. Денис Колисниченко. Программирование для Android. -2020

14. Донн Фелкер Android: разработка приложений для чайников. : Пер. с

англ. — М. : ООО “И.Д. Вильямс”, 2012. — 336 с.

15. Харди Б. , Филлипс Б. Программирование под Android. Для профессионалов. — СПб.: Питер, 2014. —592 с.

16. Микитишин А.Г., Митник М.М., Стухляк П.Д. Телекомунікаційні системи та мережі : навчальний посібник для студентів спеціальності 151 «Автоматизація та комп’ютерно-інтегровані технології» – Тернопіль: Тернопільський національний технічний університет імені Івана Пулюя, 2017 – 384 с.