Форма № 24

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

НУ „Запорізька політехніка"
(повне найменування інституту, факультету)

Інтелектуальні технології мікросистемної радіоелектроніки
(повне найменування кафедри)

Пояснювальна записка

до дипломного проекту (роботи)

магістр
(ступінь вищої освіти)

на тему Системи контролю якості пакування та обліку кількості продукції підприємства з виробництва побутової хімії

Виконав: студент(ка) 5 курсу, групи РТ-518м
Висічов Д.І.

Спеціальності 172 „Телекомунікації та радіотехніка"
(код і найменування спеціальності)

Освітня програма (спеціалізація)
Інтелектуальні технології мікросистемної радіоелектронної техніки
(прізвище та ініціали)

Керівник Мала О.Ю
(прізвище та ініціали)

Рецензент Заковоротнов В.О.
(прізвище та ініціали)

20 19

Форма № 25

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»
(повне найменування закладу вищої освіти)

[Ін]ститут, факультет ІІРЕ, ФРЕТ
[Ка]федра ІТЕЗ
[Ст]упінь вищої освіти магістр
[Сп]еціальність 172, Інстелекомунікаційні технології мікрохвильової радіо
(код і найменування)
[Ос]вітня програма (спеціалізація) 172 Телекомунікації та радіотехніка
(назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ
Завідувач кафедри ІТЕЗ
Т. М. Шило
«17» грудня 2019 року

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

Басілов Дмитро Ігорович
(прізвище, ім'я, по батькові)

[Т]ема проекту (роботи) Система контролю якості пакування та [об]ліку кількості продукції підприємства з виробництва побутової лінії

[Ке]рівник проекту (роботи) Михно Олександр Юрійович,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

[за]тверджені наказом закладу вищої освіти від «08» листопада 2018 року № 368

Строк подання студентом проекту (роботи) 16 грудня 2018

Вихідні дані до проекту (роботи) [illegible]

науково-техн. література
– [illegible]

Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно зробити) 1. огляд області розробки та постановка задачі 2) розробка структурної схеми [illegible] 3) розробка [illegible] 4) розробка керуючого програмного забезпечення системи на мікроконтролері 5) [illegible] 6) Економічне обґрунтування 7) Охорона праці та безпека у надзвичайних ситуаціях, Висновки, перелік літератури, Додаток А, додаток Б, додаток В, додаток Д

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

1) схема електрична структурна принципова
2) ПЛАТА, Складальне креслення
3) ПЛАТА [illegible]

Консультанти розділів проекту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1	Малий О.Ю. к.т.н. доц. каф. ІТЕЗ		
1.1	Малий О.Ю к.т.н. доц. каф. ІТЕЗ		
2	Малий О.Ю к.т.н. доц. ІТЕЗ		
3	Малий О.Ю к.т.н. доц. ІТЕЗ		
4	Лівшиць Т.В. к.е.н. доц.		
5	Яцишин Ю.В. к.і.н.		
Нормоконтроль	Лосєва Т.Є		

7. Дата видачі завдання «3» вересня 20 21 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту (роботи)	Строк виконання етапів проекту (роботи)	
1	Огляд області розробки та постановка задач	11.10	
2	Розробка структури системи охоронної сигналізації	13.10	
3	Розробка конструкції системи	16.10	
4	Розробка керуючого програмного забезпечення та методів роботи з системою	20.10 20.11	
5	Методики роботи з системою та техніки безпеки	24.11	
6	Економічне обґрунтування	28.11	
7	Охорона праці та безпеки у надзвичайних ситуаціях	07.12	

Студент(ка) _____ (підпис) Бичінська Д. (прізвище та ініціали)

Керівник проекту (роботи) _____ (підпис) Малий О.Ю. (прізвище та ініціали)

РЕФЕРАТ

Магістерська робота містить 122 с., 22 табл., 36 рис., 10 джерел.

Об’єкт дослідження – програмно-апаратний модуль контролю якості пакування фасованої продукції по масі.

Предмет дослідження – розробка модулю контролю якості пакування продукції на підприємстві побутової хімії за масовими характеристиками з розробкою програми обліку продукції, що випускається.

Мета роботи – розробка системи контролю якості пакування та обліку кількості продукції підприємства з виробництва побутової хімії.

Методи дослідження – системний, діалектичний, структурно-функціональний, абстрагування, аналіз, синтез, індукція, дедукція, аналогія, порівняння, статистичний аналіз.

Внутрішній контроль якості-це сукупність процедур, досліджень і методик, спрямованих на забезпечення виробництва якісної готової продукції.

Внутрішній контроль якості здійснюється на всіх етапах виробництва: від моменту отримання сировини (інгредієнтів, з яких складається продукт) і комплектуючих матеріалів (упаковки) до надходження готової хімічної продукції на склад.

Система внутрішнього контролю якості поширюється на всі ділянки виробництва.

МОДУЛЬ, СТРУКТУРНА СХЕМА, ВАГОВИЙ ТЕРМІНАЛ, МОВА ПРОГРАМУВАННЯ, КОНТРОЛЬ ЯКОСТІ, ОБЛІК ПРОДУКЦІЇ

ЗМІСТ

ВСТУП

Хоча загальні принципи контролю якості протягом багатьох років залишаються незмінними, в даний час замість поняття «контроль» часто використовується поняття «забезпечення».

У словниках терміна «забезпечувати» (to assure) зіставляється значення «гарантувати», тоді як «контролювати» (to control) означає управляти, регулювати в рамках визначених повноважень. У зв'язку з цим термін «забезпечення» видається більш гідною кандидатурою, оскільки управління в рамках даних повноважень часто не гарантує хороших результатів (що варто мати на увазі менеджерам з контролю якості).

Відповідальність за виробництво відповідно до стандарту якості повністю лежить на виробничих підрозділах, в яких в даний час працюють інженери і технологи. Абсолютно невірно вважати, що за якість продукту відповідає відділ контролю якості (або ВТК). При такому підході виробничий персонал починає піклуватися тільки про кількість, а не про якість, і підрозділи, зайняті контролем якості, змушені бракувати непридатну продукцію на виході технологічної лінії, що вельми марнотратно. Справжньою завданням відділу контролю якості є своєчасне надання інформації про помічених відхиленнях від стандарту якості.

Поставки некомплектної або неякісної продукції на ринок негативно позначаються на бізнесі. В результаті виробник може втратити довіру клієнтів, погіршити репутацію і зіткнутися з відкликанням продукції.

Метою дипломної роботи є розробка системи контролю якості пакування та обліку кількості продукції підприємства з виробництва побутової хімії

Цільове підприємство ТОВ «Бара», що займається виробництвом побутової хімії.

Згідно з поставленою метою були поставлені наступні задачі та стадії

- розробка і затвердження технічного завдання;

- розробка структури системи;

- вибір виконавчих елементів та розробка схеми електричної принципової системи;

- написання програмного забезпечення;

- тестування системи, знаходження і усунення можливих несправностей;

- доповнення та виправлення початкового технічного завдання на підставі тестування;

- внесення змін і доповнень відповідно до додаткового договору та остаточне тестування роботи системи;

- установка розробленої системи на підприємство замовника;

- розробка умов та вимог до розробленого обладнання;

- складання технічної документації.

1. ОГЛЯД ОБЛАСТІ РОЗРОБКИ ТА ПОСТАНОВКА ЗАДАЧ

1.1 Використання систем контролю якості та систем обліку продукції на виробництві

Виробництво виробів з надмірною масою, природно, позначається на прибутку, а виготовлення великої кількості продуктів з масою нижче номінальної є обманом споживачів і порушенням закону. Звідси випливає необхідність організації системи контролю маси виробів. Деякі підприємства використовують карти типу «менше-більше» з нульовою лінією, що представляє собою номінальну (правильну) масу. Цілі числа на карті, що позначають грами, залежать від виду продукту. По карті розподілу мас контролер може швидко скласти уявлення про те, як розподілені маси виробів - якщо спостерігається зрушення в бік верхнього контрольного межі, оператори виконують відповідні регулювання в сторону зменшення маси, а коли спостерігається зрушення в бік нижнього контрольного межі, то виконується регулювання в сторону збільшення.

Крива розподілу дозволяє визначити середню масу, що закладалася в собівартість. Крім того, вона показує характеристики установки і вказує на точки, де є потреба у посилений контроль. Така крива дозволяє оператору визначити момент, коли ситуація «виходить з-під контролю» і для виправлення необхідно внести корективи в налаштування. Для одержання кривої розподілу, а на її основі повірочної карти, необхідний відповідний пробовідбір. Якщо в результаті аналізу проб не отримана колоколообразная крива (нормальний закон розподілу), то значить, процес вийшов з-під контролю. При отриманні колоколообразной кривої нижня лінія, відповідна середньоквадратичного відхилення, - це нижній контрольний межа, а верхня лінія - верхня межа. Метод, який використовується для підготовки повірочних карт і карт розподілу, описаний нижче.

1.2 Постановка задач дипломного проекту

Найменування програмно-апаратного комплексу (системи): «Scale Connect Pallet Control».

«Scale Connect» в назві системи об'єднує сімейство програмно-апаратних засобів використовуваних для передачі даних з вагових терміналів, віддаленого управління ваговими терміналами і Вагодозуючі пристроями.

«Pallet Control» - характеризує точне призначення системи. У даній ситуації система призначена для контролю ваги палет (піддонів) з ящиками з виробленою продукцією.

Програмно-апаратний комплекс (система) «Scale Connect Pallet Control» призначений для роботи з електронними палетними вагами з максимальною межею зважування 1 т. Даний програмно-апаратний комплекс використовується для реєстрації, обліку та зберігання даних результатів зважування вантажів (локально в базі даних і на сервері) з можливістю вибору найменування вантажу за допомогою клавіатури комп'ютера. Можливість віддаленого доступу до інформації (на сервері підприємства), користувач може проводити не тільки облік товарообігу вантажів, що зважуються на вагових терміналах, а й складати звіти про фінансовому еквіваленті цього товарообігу.

Програмно-апаратний комплекс (система) може застосовуватися в різних галузях економіки для обліку зважувань товарів і вантажів.

Основні вимоги до системи:

1) Система повинна фіксувати вага піддона, ящиків, перестилаючи з похибкою не більше 300 м включно. І відображати вагу на індикаторі і сигналізувати про відхилення ваги ящика від заданого діапазону.

2) Час фіксації ваги ящика, що входить в заданий допустимий діапазон 1с.

3) Час фіксації ваги ящика з відхиленнями, польоти і перестилаючи не більше 5 с.

4) Сума ваги всіх ящиків, піддону і перестилаючи і фактична вага палети не повинні відрізнятися більш ніж на 500г включно.

5) Система повинні безпомилково фіксувати кількість і вагу встановлених ящиків.

6) Система повинна експортувати результати зважувань в текстовий файл і базу даних.

7) Можливість навчання оператора (з мінімальним рівнем володіння комп'ютером) для роботи з системою повинно займати не більше 1 години. Робота з системою повинна вимагати мінімальний перелік дій з боку оператора.

Для забезпечення наочності отриманих системою даних, а також спрощення обліку за допомогою сторонніх програмних комплексів таких як (MS Excel, 1С Бухгалтерія і т.д.) в системі необхідно передбачити крім зберігання даних в базі даних автоматичне і при необхідності ручне (за заданими фільтрам) створення файлів з даними в форматі xls (для роботи в MS Excel) та txt (для обміну даними з програмою розробленою для автоматизованого обліку в системі 1С Бухгалтерія).

Файл формату xls формується за запитом користувача за заданими фільтрам і зручний для проведення статистики за різними групами товарів, змінах, діапазонами маси повної і порожньої палети, статистики кількості зваженої продукції за проміжок часу.

Структура файлу формату xls для повного звіту зважувань має такий вигляд (у одному рядку):

- Дата;
- Час;
- Найменування товару;
- Вага ящика (в грамах);
- Вага повної палети (в грамах);
- Вага порожньої палети (в грамах);
- Вага Перестила (в грамах);
- ПІБ Оператора;
- Номер зміни;
- Кількість ящиків на палеті.

Крім того повинні формуватися загальні звіти в яких вказується кількість зважених палет за період, за зміну, певним оператором, за певним найменуванням продукції.

Файл формату txt формується автоматично і має назву вес.txt і містить інформацію про останні 7мі добі. Після закінчення 7мі доби файл перейменовується в файл з назвою весРРММДД.txt (де РР останні дві цифри року, ММ - номер місяця і ДД число місяця, які відповідають даті його створення тобто датою створення нового файлу вес.txt), а замість нього створюється новий файл з назвою вес.txt. Файли формату весРРММДД.txt адміністратор системи може самостійно видаляти в разі потреби або налаштувати систему на автоматичне видалення через певний період часу.

2. РОЗРОБКА СТРУКТУРИ СИСТЕМИ ТА СХЕМИ ЕЛЕКТРИЧНОЇ ПРИНЦИПОВОЇ

2.1 Розробка структури системи

Система складається з таких функціональних елементів:

1) Електронні тензометричні палетні ваги з максимальною межею зважування 1000 кг складаються з платформи з тензометричним датчиком і вагового терміналу (з можливістю автоматичного визначення стабілізації ваги для ідентифікації моменту передачі даних на ПК з метою виключення помилкового зважування) сполученого з платформою за допомогою шнура.

2) Персональний комп'ютер для установки програмного забезпечення. Для кількох екземплярів (до 3х) системи може використовуватися один персональний комп'ютер. Чим більша кількість підключень, тим більші апаратні вимоги до ПК.

3) Кабель або приймач на частоті 2,4 ГГц для передачі даних від вагового терміналу до персонального комп'ютера. При використанні COM порту для комутації між ваговим терміналом і персональним комп'ютером для забезпечення стабільного зв'язку довжина кабелю не повинна перевищувати 50м.

4) Програмне забезпечення встановлюється на ПК для перегляду та редагування даних, отриманих з вагових терміналів і формування звітів.

5) Локальна база даних (на ПК) для довготривалого зберігання інформації про проведені зважуваннях

6) Пристрій додаткової індикації для відображення даних біля робочого місця оператора, в разі якщо відстань від робочого місця оператора і персональним комп'ютером більше 2м (для зручності управління системою). Пристрій являє собою дублюючий індикатор.

Вантаж, що зважується (ящики з продукцією, перестилаючи, піддон) фіксуються ваговим терміналом і при стабілізації ваги (час стабілізації від 0,2 с до 3 с, в залежності від того, на скільки коливається платформа при установці ящика) термінал через сполучний кабель або радіоканал передає масу вантажу на персональний комп'ютер. Персональний комп'ютер фіксує отримане значення маси і

час твори зважування і записує отримані дані в локальну базу даних з паралельним відображенням користувачеві прочитаної інформації на екрані монітора. У базу даних також заносяться дані про тип вантажу (ящик з можливим зазначенням продукції за бажанням користувача, палети, пересилу - система на основі попередньо встановлених параметрів автоматично визначає тип вантажу). При наявності зв'язку з локальним сервером підприємства дані з локальної бази даних синхронізуються з базою даних на сервері.

2.2 Вибір електрорадіоелементів та розробка схеми електричної принципової

Новою унікальною конструкцією в системі, що розробляється в дипломному проекті є пристрій додаткової індикації для відображення даних біля робочого місця оператора.

Керуючим елементом системи є мікроконтроллер, а в якості керуючою схеми використовується мікроконтролер на платі Ардуіно оскільки там вже промислово виконано всі з’єднання.

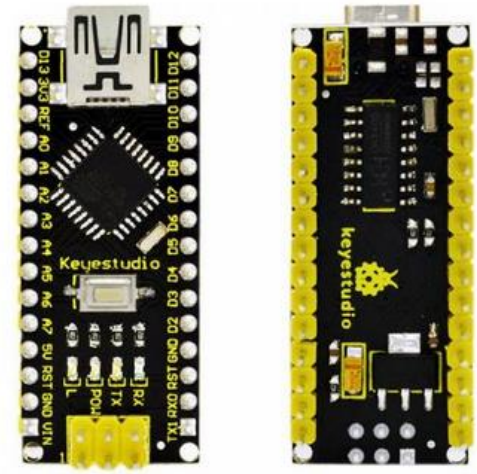


Рисунок 2.1 – Керуючий мікроконтролер у складі плати Arduino Nano

Стислі характеристики Arduino Nano:

- Мікроконтролер: ATmega328;
- Робоча напруга (логічна рівень): 5 В;
- Вхідна напруга (рекомендований): 7-12 В;

- Вхідна напруга (граничне): 6-20 В;

- Цифрові Входи / Виходи: 14 (6 з яких можуть використовуватися як виходи ШІМ);

- Аналогові входи: 8;

- Постійний струм через вхід / вихід: 40 мА;

- Флеш пам`ять: 32 Кб при цьому 2 КБ використовуються для завантажувача;

- ОЗУ: 2 КБ;

- EEPROM: 1 Кб;

- Тактова частота: 16 МГц;

- Розміри: 1.85 см х 4.2 см;

Для забезпечення можливості відображення параметрів для оператора було обрано рідинно кристалічний монохромний графічний індикатор, що наведено на рис.2.2

Рисунок 2.2 – Обраний індикатор

Графічний дисплей 128x64 точки з підсвічуванням на контролері ST7920. На відміну від текстових дисплеїв дозволяє виводити так само і картинки. Може бути підключений як за паралельним так і по послідовному інтерфейсу.

Дисплей підтримує як паралельний, так і послідовний інтерфейс передачі даних. Обидва інтерфейсу підтримує і бібліотека U8glib дозволяє працювати з дисплеями 12864B V2.0. Для передачі даних по послідовному інтерфейсу бібліотека U8glib може використовувати як апаратний так і програмний SPI. При використанні паралельного інтерфейсу або програмного SPI дисплей можна підключити до будь-яких висновків Arduino. А при використанні апаратного SPI дисплей підключається тільки до висновків апаратної шини SPI.

Для забезпечення бездротового зв’язку з персональним комп’ютером, що надасть можливості проводити зєднання без необхідності протягування на підприємстві додаткових дротових ліній було застосовано радіозв’язок.

Для його реалізації було обрано пару радіо модулів NRF24L01 (рис2.3). NRF24L01 працює на частоті 2.4ГГц, швидкість до 2Мбіт, підключення виконується по інтерфейсу SPI, живлення 3.3В. Істотно дешевше xbee модуля, добре підходить для промислових систем управління.

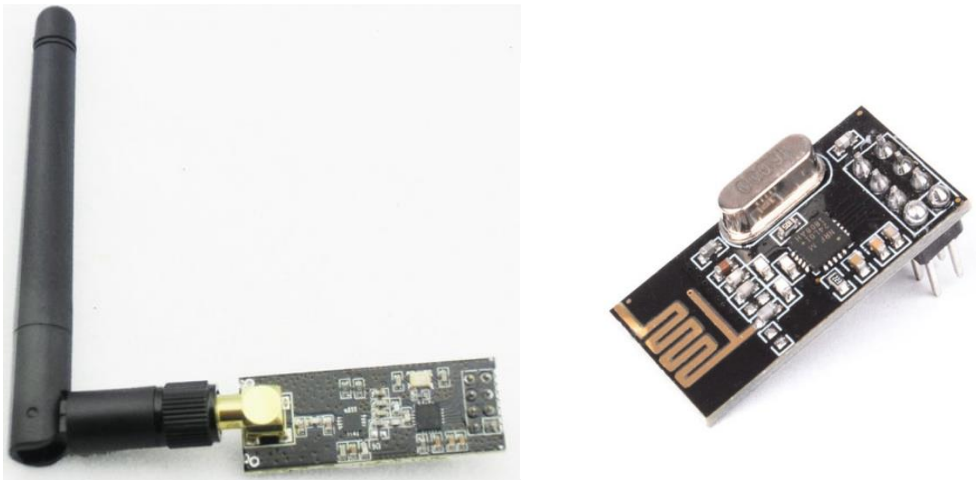


Рисунок 2.3 - Радіомодуль NRF24L01

Організація живлення радіомодулів вимагає підвищеної уваги, так як більшість початківців користувачів стикається з проблемами при їх запуску. Справа в тому, що в момент ініціалізації NRF24L01 споживають значний струм, який не

може забезпечити стандартний 3-вольта перетворювач Arduino. Як наслідок, спостерігаються збої в роботі радіозв'язку. Виключити цю неприємну ситуацію допоможе електролітичний конденсатор, ємністю близько 100 мкФ. Його необхідно підпаяти паралельно висновків GROUND і VCC модуля. Додаткова ємність допоможе згладити пульсації при старті і забезпечить достатній запас енергії.

Ще одним варіантом вирішення проблеми запуску, є використання додаткового адаптера з вбудованим стабілізатором напруги. У такому випадку для NRF24L01 можна використовувати зовнішнє живлення від 4.8В до 12В, а максимальний вихідний струм складе 800мА. Зовнішній вигляд такого адаптера показаний на рисунку 2.4.

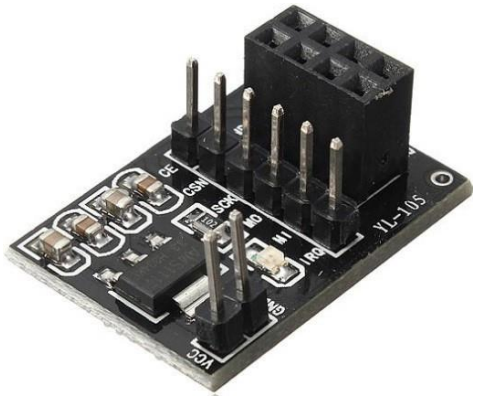


Рисунок 2.4 - Адаптер NRF24L01

У якості додаткового дротового з’єднання використовування підключення за допомогою Інтерфейсу RS232.

RS-232 (Recommended Standard 232) - стандарт описує інтерфейс для послідовної двобічної передачі даних між терміналом (DTE, Data Terminal Equipment) і кінцевим пристроєм (DCE, Data Circuit-Terminating Equipment).

Рішення, які закладені в цей стандарт, використовуються практично повсюдно.

Інтерфейс RS-232 повністю апаратно реалізований на персональних комп'ютерах у вигляді мікросхем і роз'ємів.

COM порт реалізований за стандартом RS-232- універсальний.

Робота цього інтерфейсу реалізовано за рахунок використання мікросхеми MAX232 з конденсаторною обв’язкою.

Оскільки в пристрої застосовується також звукова індикація, що сповіщає оператора про якість встановленого на палету ящика, а в виробничому приміщенні при працюючих станках дуже шумно треба також обрати підсилювач потужності аудіо частоти. В якості такого було обрано підсилювач на мікросхемі PAM8403 (рис.2.5).

Рисунок 2.5 – Обраний підсилювач частоти аудіодіапазону

Маленький але досить потужний стерео підсилювач D класу на мікросхемі PAM8403 з вихідною потужністю до 3 Вт на канал (при 10% нелінійних спотворень). Підсилювач має дуже малими шумами, невеликими розмірами і високою економічністю, що дозволяє використовувати його в портативних пристроях з автономним живленням.

З'єднувати виходи правого і лівого каналів даного підсилювача не можна - це призведе до виходу його з ладу.

Характеристики:

- мікросхема підсилювача: PAM8403;
- клас підсилювача: D;
- вихідна потужність: 2 х 3 Вт;
- опір навантаження: від 4Ом до 8Ом;
- напруга живлення: від 2.5В до 5В;
- граничне напруга живлення: 5.5В;
- розміри: 18.5 х 21.1 мм.

Після комутації обраних елементів було розроблено схему електричну принципову, яка наведена на рис.2.6.

У схемі використовуються пристрої, які працюють через периферійний інтерфейс (SPI), а саме радіомодуль NRF24L01 та дісплейний модуль.

Послідовний периферійний інтерфейс (SPI) - це синхронний протокол послідовної передачі даних, який використовується для зв'язку мікроконтролера з одним або декількома периферійними пристроями. Інтерфейс SPI відрізняється відносно високою швидкістю і призначений для зв'язку близько розташованих пристроїв. Він також може використовуватися для взаємодії двох мікроконтролерів.

Згідно з протоколом SPI, одне з взаємодіючих пристроїв (зазвичай мікроконтролер) завжди є провідним і контролює ведені периферійні пристрої. Як правило, всі взаємодіючі пристрої об'єднані трьома загальними лініями.

Також до пристрою можливо підключиться за допомогою апаратного або програмного UART.

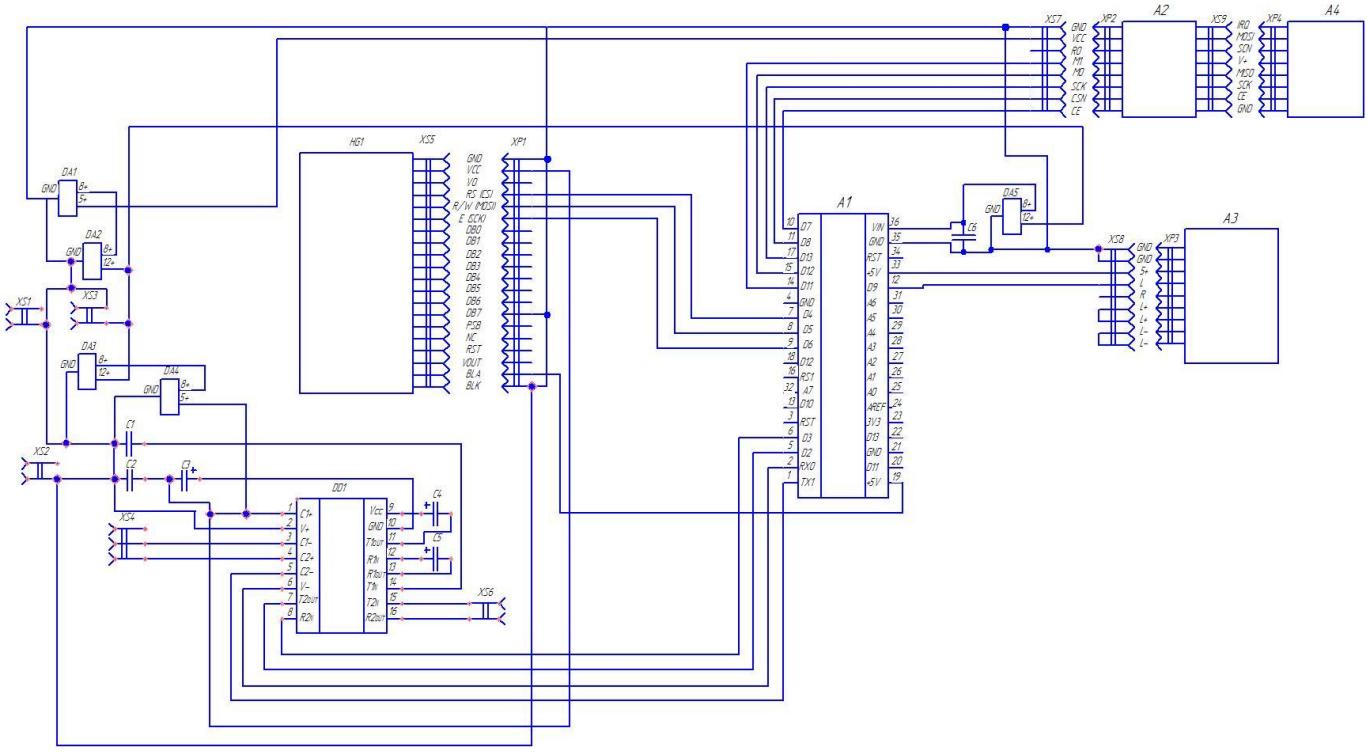


Рисунок 2.6 – Схема електрична принципова

Для розробки схеми електричної-принципової була використана програма «КОМПАС-3D»

3. РОЗРОБКА КОНСТРУКЦІЇ СИСТЕМИ

3.1 Розробка конструкції плати

Для з’єднання радіоелементів електричної схеми між собою, в якості базової несучої конструкції вибираємо двосторонню друковану плату виготовлену комбінованим позитивним методом по полуаддитивній технології. З огляду на те, що при проектуванні ПП використовуються ІС, а також високий рівень насиченості ПП навісними елементами по ГОСТ 23751-86 вибираємо четвертий клас точності.

Відповідно до того, що максимальний діаметр виводів навісних елементів, розміщених на платі, дорівнює 0,7 мм, то обираємо товщину плати рівну 1,5 мм.

Як матеріал проектованої ДПП обираємо склотекстоліт нагрівостійкості вищого сорту, товщиною 1,5 мм, облицьований з двох сторін мідною оксидною фольгою, товщиною 50 мкм

Було обрано текстоліт у зв’язку із наступними перевагами:

- чудові електростатичні показники;
- високий рівень ударної в'язкості;
- хімічна інертність;
- найвища міцність на стиск;
- простота механічної обробки.

Текстолит відмінно штампується, свердлиться, ріжеться.

Здебільшого вищевказані переваги обумовлені шаруватою структурою і перевагами бавовняної тканини, яка є сировиною для виготовлення полімеру. Вони дозволяють застосовувати текстолітові вироби в умовах високочастотного електричного поля, підвищеного тертя, сильних механічних навантажень.

Наступною перевагою цього конструкційного матеріалу виступає нетоксичність матеріалів чудова стійкість до високих температур, впливу агресивних середовищ..

В конструкції даного пристрою застосовуються стандартні ЕРЕ, що мають вологозахисне покриття та низьку інтенсивність відмов, що забезпечує надійну працездатність пристрою протягом гарантованого терміну служби при впливі на

нього несприятливих кліматичних факторів. ЕРЕ кріпляться на платі за допомогою пайки.

Номінальні діаметри отворів для кріплення складають 3,4 мм.

Друкована плата була розтрассірованна на двох сторонах. Ширина трас з'єднань 0,5 мм. Допуск між провідником і елементом друкованої плати 0,5 мм, допуск між провідником і отвором 0,25 мм, допуск між провідником і краєм друкованої плати 2 мм, допуск відстаней елементів друкованої плати 0,15 мм.

Для розробки розводки плати була обрана програма Sprint Layout v.6, тому що, програма підтримує швидкий експорт в Gerber формат, який в подальшому вирушає на виробництво (рис. 3.1 та 3.2).

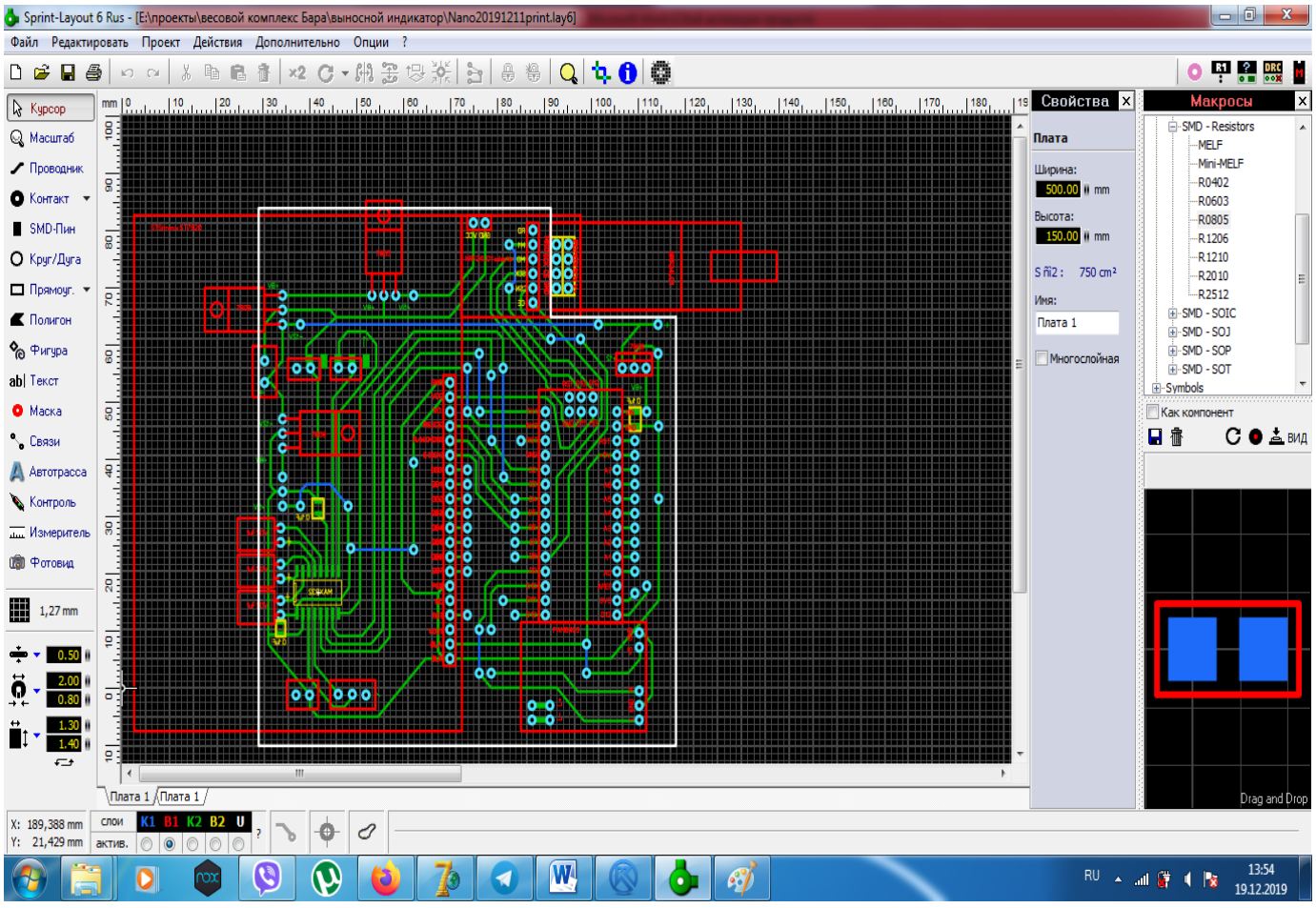


Рисунок 3.1 – Плата дублюючого індикатора розроблена в програмі Sprint Layout

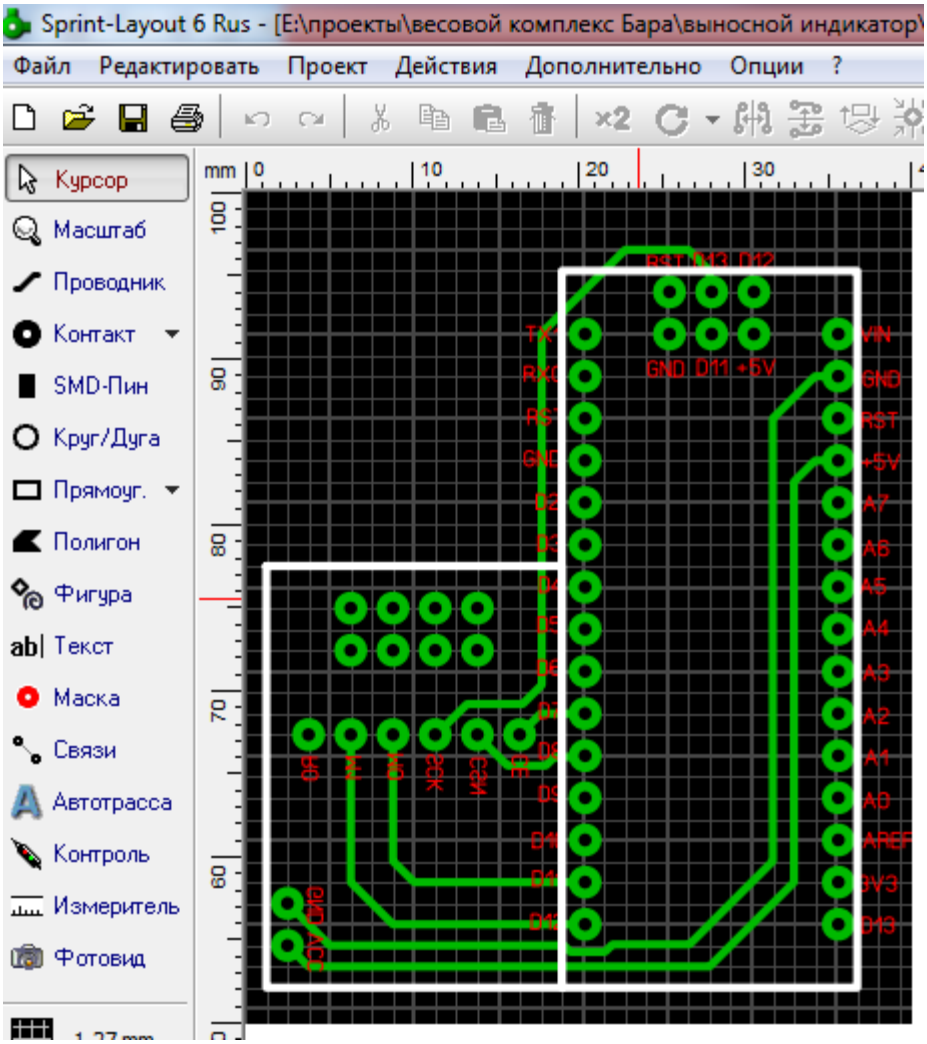


Рисунок 3.2 – Плата приймача, що приєднується до персонального комп'ютера розроблена в програмі Sprint Layout

Для розробки схеми креслень була використана программа «КОМПАС-3D».

Ключовою особливістю продукту є використання власного математичного ядра C3D і параметричних технологій, розроблених фахівцями АСКОН.

КОМПАС-3D забезпечує підтримку найбільш поширених форматів 3D-моделей (STEP, ACIS, IGES, DWG, DXF), що дозволяє організовувати ефективний обмін даними із суміжними організаціями і замовниками, що використовують будь-які CAD / CAM / CAE-системи в роботі.

Отримане креслення сторони установки елементів поверхневого монтажу наведено на рис. 3.3. Креслення сторони установки елементів об'ємного монтажу наведено на рис. 3.4.

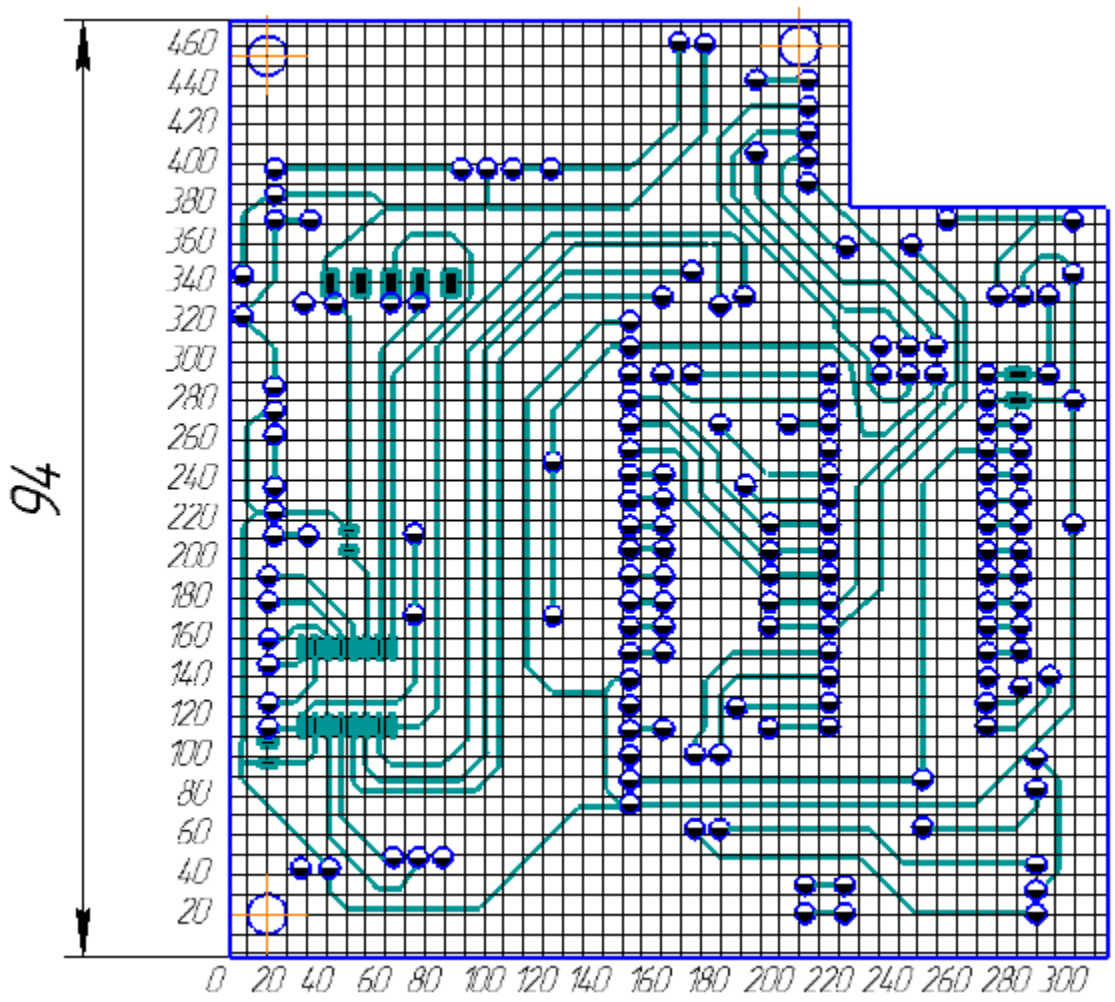


Рисунок 3.3 – Ескіз креслення розводки плати для установки елементів поверхневого монтажу

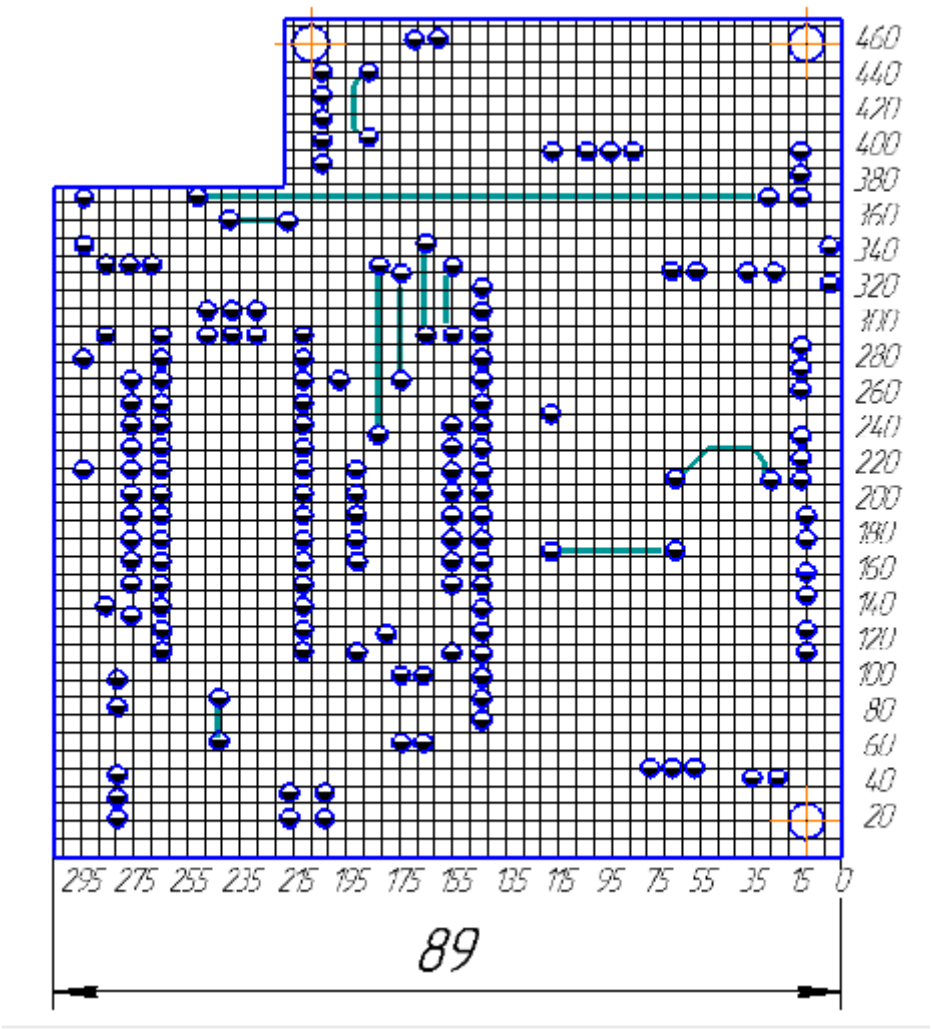


Рисунок 3.4 – Ескіз креслення розводки плати для установки елементів об’ємного монтажу

Технічні вимоги до виготовлення друкованої плати:

1) Плату виготовити комбінованим методом.

2) Плата повинна відповідати ОСТ 4Г0.077.200.

3) *Розмір для довідок. Клас точності 0,25 мм.

4) Крок координатної сітки 1,25 мм.

5) Незазначені проедельние відхилення розмірів між осями двох будь-яких отворів ±0,2 мм.

6) Конфігурацію провідників витримати за кресленням з відхиленням між осями двох будь-яких отворів ±0,2 мм.

7) Параметри елементів провідного малюнка наведені в таблицях 1,2.

8) Інші технічні вимоги по ОСТ 4Г0.070.014.

Компонування приладу зроблена таким чином, щоб забезпечити вільний доступ до елементів схеми при регулюванні і заміні в разі виходу їх з ладу.

Для виготовлення друкованої плати був обраний комбінований позитивний метод. Даний метод був обраний з урахуванням наступних чинників виготовлення друкованої плати:

- спроектований пристрій виконаний по одношарової, двосторонньої технології з SMD монтажем;

- необхідна висока точність друкованого малюнка;

- необхідна висока надійність і якість друкованого малюнка.

Переваги:

- створення елементів друкованого малюнка з високою точністю – при використанні фольги товщиною 9 мкм досяжна ступінь допуску провідників і проміжків між ними – 75 мкм;

- практично на всіх етапах техпроцесу фольга захищає діелектричну підставку від впливу технологічних розчинів; цим досягається висока якість поверхні діелектрика і, як наслідок, висока надійність ізоляції;

- добра адгезія (міцність зчеплення) елементів друкованого малюнка і діелектричного підстави плати.

Недоліки: Операцій травлення призводить до виникнення бокового підтраву провідників. Це обмежує роздільну здатність процесу.

Комбінований позитивний метод застосовується при виробництві двосторонніх друкованих плат. За своєю суттю комбіновані способи виготовлення плат відносяться до полуаддитивним. Як і при субтрактивному методі, для виготовлення плат по полуаддитивній технології використовуються фольговані діелектрики. Формування малюнка провідників відбувається, як і при адитивних методах, шляхом гальванічного осадження міді з застосуванням фотошаблонів.

Формування виводів і установка елементів є стандартною по ОСТ 4ГО.010.030, крім елементів зазначених на кресленні – плата в зборі. Кількість типорозмірів елементів зведено до мінімуму.

Завдяки тому, що майже всі елементи встановлюються на одній стороні плати, і як було сказано раніше, застосована стандартна елементна база, для установки і

пайки ЕРЕ використовуються автоматизовані системи, добре відпрацьовані на виробництві. Таким чином, можна використовувати групову пайку, зокрема пайку хвилею, установку елементів здійснювати шляхом використання спеціалізованих автоматів. Це в свою чергу зменшить витрати часу, фінансові та трудові ресурси на виробництво даного вироби при великосерійному типі виробництва.

Після розробки розводки плати було розроблено конструкцію складання та зроблене складальне креслення (рис.3.5 та рис.3.6)

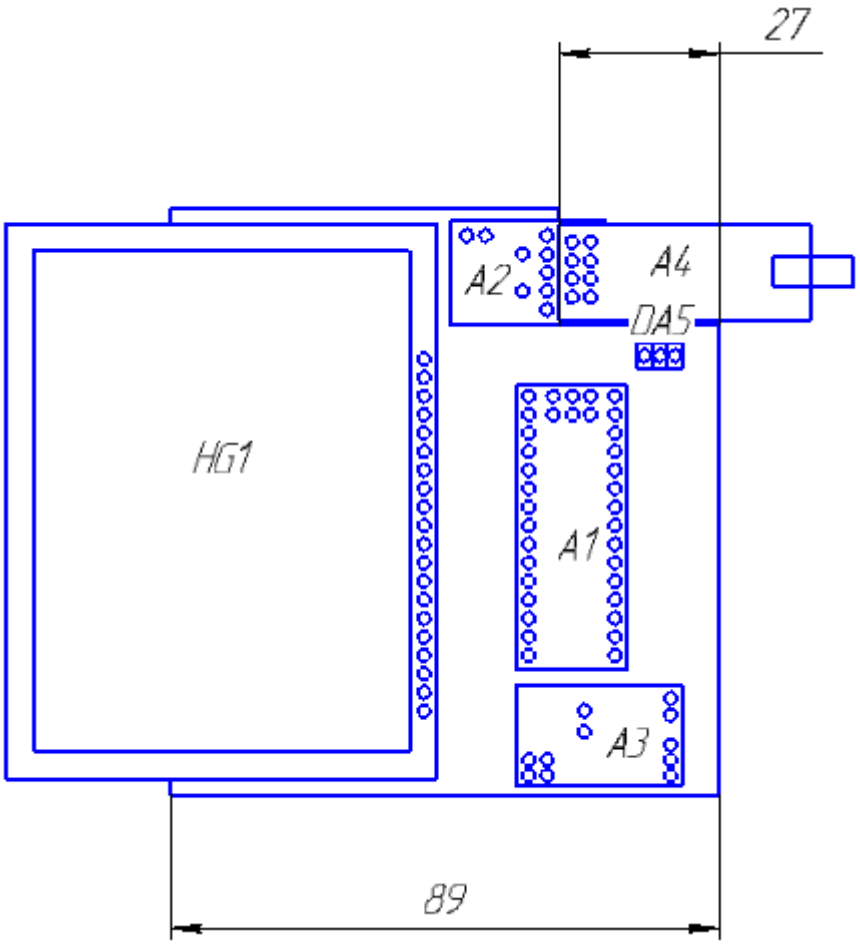


Рисунок 3.5 – Ескіз складального креслення плати зі сторони встановлення елементів об’ємного монтажу

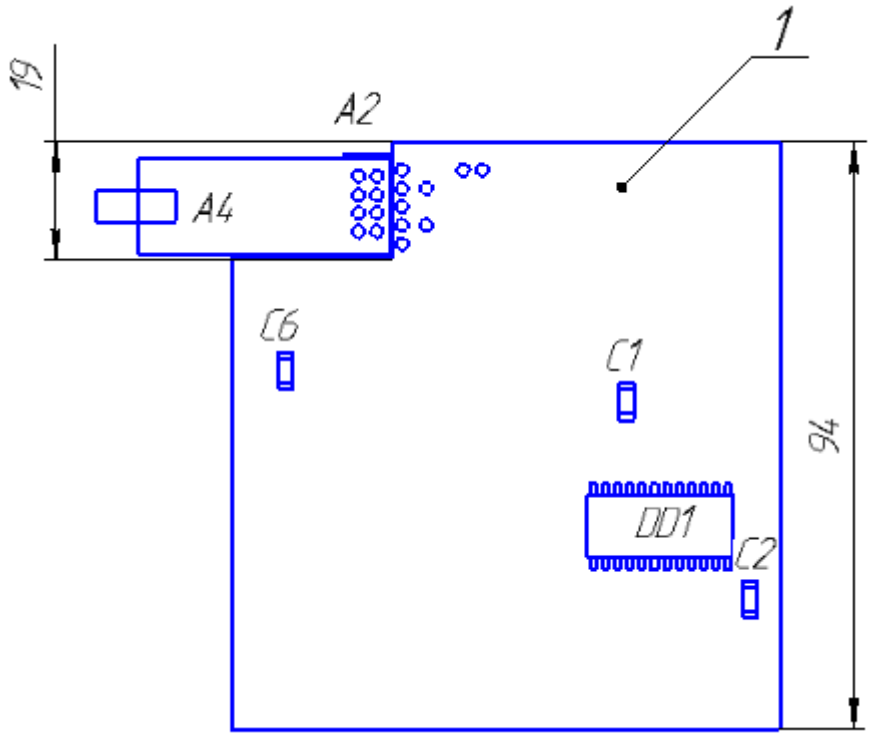


Рисунок 3.6 – Ескіз складального креслення плати зі сторони встановлення елементів поверхневого монтажу

3.2 Розробка конструкції корпусу

В якості конструкції корпусу було обрано базову несучу конструкцію електричної розподільчої коробки з подальшим прорізанням отвору для дисплею.

Перевагами використання такої конструкції є її невелика вартість та доступність, а також можливість монтажу з’єднувальних проводів через гумові виводи, що дозволяють робити отвори різних діаметрів.

В корпусі плата закріпляється болтовим з’єднанням (4 болти) у отвори дисплею, який встановлюється в роз’їм. Рисунок зібраної конструкції наведено на рис.3.7.

Рисунок 3.7 – Фото зібраної конструкції табло додаткової індикації

Конструкція корпусу приймача також виконана на базі базової несучою конструкції – електричної розподільчої коробки але меншого розміру (рис.3.8)

Рисунок 3.8 – Фото зібраної конструкції приймача сигналу

4. Розробка керуючого програмного забезпечення системи та методики роботи з системою

4.1 Розробка програмного забезпечення керуючого мікроконтролера

Для розробки програмного забезпечення було використано середу розробки Arduino IDE оскільки вона саме налаштована для написання програмного забезпечення для плат Arduino.

Проект включає в себе файли бібліотек з розширенням * .h та *cpp, Arduino IDE виконує завдання з'єднання файлів підпрограм і бібліотек в єдину «прошивку» для запису коду в пам'ять програм мікросхеми мікроконтролера.

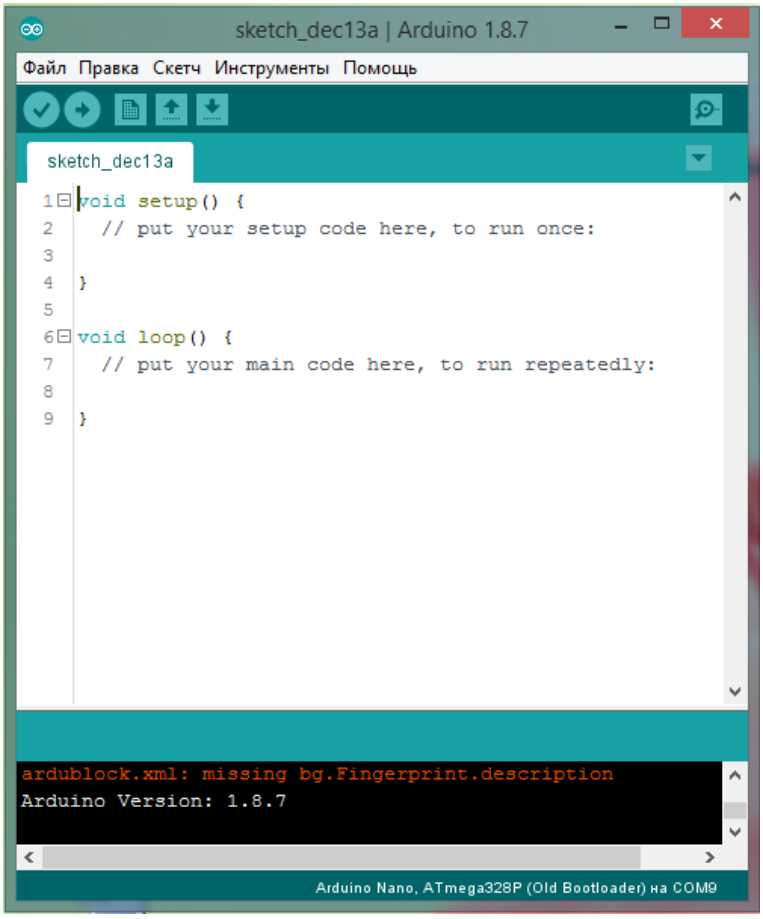


Рисунок 4.1 – Середовище розробки програм для мікроконтролерів, Arduino IDE

Розглянемо більш детально підключені бібліотеки:

1) Бібліотека SPI дозволяє контролеру Arduino взаємодіяти з пристроями що підтримують SPI протокол. Arduino в даному випадку виступає в якості ведучого устройтва.

Для організації з'єднання SPI необхідно одне провідне пристрій, зазвичай це мікроконтролер, яке управляє з'єднанням з відомими пристроями. Зазвичай підключення здійснюється трьома загальними лініями і лінією вибору переферійного (веденого) пристрої:

Master In Slave Out (MISO), перекладається як "вхід ведучого вихід веденого", використовується для передачі даних від веденого до ведучого.

Master Out Slave In (MOSI) - вихід ведучого вхід відомого, для передачі даних від провідного до периферійних пристроїв.

Serial Clock (SCK) - синхронізуюча лінія, синхросигнал генерується провідним пристроєм.

Slave Select pin - вхід на ведених пристроях за допомогою якого ведучий може ініціювати обмін даними з периферійним пристроєм. Якщо на цьому вході LOW, то ведений взаємодіє з провідним, якщо HIGH, то ведений ігнорує сигнали від ведучого.

2) Бібліотека RF24 використовується для роботи з радіо модулями nRF24L01 +

Бібліотека дозволяє налаштовує роботу радіо модуля в режим передачі / прийому даних, вказувати номер каналу, швидкість передачі, потужність передачі і ідентифікатор труби. Після чого, зчитувати і відправляти дані радіо модулю..

3) Бібліотека SoftwareSerial дозволяє реалізувати послідовний інтерфейс на будь-яких цифрових портах Ардуіно за допомогою програмних засобів, які дублюють функціональність UART (звідси і назва "SoftwareSerial"). Бібліотека дозволяє програмно створювати кілька послідовних портів, які працюють на швидкості до 115200 бод. Для пристроїв, що працюють з інвертованим сигналом, в бібліотеці передбачено відповідний параметр, що включає інвертування.

4) Бібліотека EEPROM. У мікроконтролері Ардуіно є EEPROM - пам'ять, в якій інформація зберігається навіть після вимкнення апарата (подібно маленькому жорсткому диску). Дана бібліотека дозволяє записувати і зчитувати інформацію з цієї пам'яті.

Обсяг пам'яті EEPROM різних мікроконтролерів, що входять до складу Ардуіно, може відрізнятися: 1024 байти в ATmega328, 512 байт - в ATmega168 та ATmega8, 4 КБ (4096 байт) - в ATmega1280 і ATmega2560.

Бібліотека EEPROM потрібна для збереження налаштувань у пам’яті мікроконтролеру навіть після вимкнення пристрою.

5) Бібліотека U8glib.h для роботи з графічним дисплеєм

Повний текст програми керуючого мікроконтролера виносного індикатора та приймача наведено відповідно у додатках А та Б.

4.2 Розробка бази даних

Для зберігання даних програми було розроблено базу даних, що складається з 12 таблиць (табл.4.1-4.12).

Таблиця 4.1 – Таблиця БД Зважування за міяцями Weightings_YYMM (YY рік MM місяць)

№	Назва стовбця	Що зберігається	Тип даних	На які таблиці посилається
1	Id_weight	Номер запису (ключ таблиці)	INT	
2	WDateTime	Дата, час зважування	DATETIME	
3	Massa	Маса зважування в граммах	INT	
4	Cargo_Id	Товар	INT	Cargos (Id_cargo)
5	User_Id	Користувач	INT	Users (Id_user)
6	WorkShift	Зміна (номер)	INT	
7	Pallette	Палета (номер по порядку)	INT	
8	WType	Тип того, що зважується (0 – unknown <min box, 1 – good box, 2 – septum, 3 – empty palette, 4 – angle, 5 - unknown >max box, 6 – масса кратная нескольким ящикам)	INT	

Продовження таблиці 4.1

№	Назва стовбця	Що зберігається	Тип даних	На які таблиці посилається
9	Wactivity	Активність (зважування видалено – 0, зважування ящика залишилось в обліку –1)	INT	
10	Scale_Id	Ваги (робоче місце)	INT	Scales (Id_scale)
11	TaskDate	Дата наряда	DATE	
12	Closed	Палета закрита (0 – закрита, 1 – відкрита)	INT	
13	BoxCount	Кількість покладений нормальних та ненормальних ящиків	INT	

SQL запит для створення таблиці:

CREATE TABLE Weightings (Id_weight INT, WDateTime DATETIME, Massa INT, Cargo_Id INT, User_Id INT, WorkShift INT, Pallette INT, WType INT, WActivity INT, Scale_Id INT, TaskDate DATE, Closed INT, BoxCount INT, PRIMARY KEY (Id_weight), FOREIGN KEY (Cargo_Id) REFERENCES Cargos(Id_cargo), FOREIGN KEY (User_Id) REFERENCES Users (Id_user), FOREIGN KEY (Scale_Id) REFERENCES Scales(Id_scale));

Таблиця 4.2 – Продукция Cargos

№	Назва стовбця	Що зберігається	Тип даних	На які таблиці посилається
1	Id_cargo	Номер запису (ключ таблиці)	INT	
2	Cargo_Name	Назва товару	VARCHAR(150)	
3	Category_Id	Категорія товару	INT	Categories (Id_category)

Продовження таблиці 4.2

№	Назва стовбця	Що зберігається	Тип даних	На які таблиці посилається
4	Cargo_Code	Код товару	INT	
5	MinBox	Мінімальна масса ящика в граммах	INT	
6	MaxBox	Максимальна масса ящика в граммах	INT	
7	CountInBox	Кількість штук в ящику	INT	
8	Price_Id	Ціна за одиницю товару	INT	Prices (Id_price)
9	Descr_Id	Опис товару	INT	Descriptions (Id_description)
10	Photo_Id	Фото товара	INT	Photos (Id_photo)
11	CActivity	Активність (0 – Видалений товар, 1 – Активний товар)	INT	

SQL запит для створення таблиці:

CREATE TABLE Cargos (Id_cargo INT, Cargo_Name VARCHAR(150), Category_Id INT, Cargo_Code INT, MinBox INT, MaxBox INT, CountInBox INT, Price_Id INT, Descr_Id INT, Photo_Id INT, CActivity INT, PRIMARY KEY (Id_cargo), FOREIGN KEY (Category_Id) REFERENCES Categories (Id_category), FOREIGN KEY (Price_Id) REFERENCES Prices (Id_price), FOREIGN KEY (Descr_Id) REFERENCES Descriptions (Id_description), FOREIGN KEY (Photo_Id) REFERENCES Photos (Id_photo));

Таблиця 4.3 – Категорії товарів Categories

№	Назва стовбця	Що зберігається	Тип даних	На які таблиці посилається
1	Id_category	Номер запису (ключ таблиці)	INT	
2	Category_Name	Назва категорії	VARCHAR(100)	
3	Category_Description	Опис категорії	INT	Descriptions (Id_description)
4	Cat_Activity	Активність (0 – Видалена категорія, 1 – Активна категорія)	INT	

SQL запит для створення таблиці:

CREATE TABLE Categories (Id_category INT, Category_Name VARCHAR(100), Category_Description INT, Cat_Activity INT, PRIMARY KEY (Id_category), FOREIGN KEY (Category_Description) REFERENCES Descriptions (Id_description));

Таблиця 4.4 – Ціни на товари Prices

№	Назва стовбця	Що зберігається	Тип даних	На які таблиці посилається
1	Id_price	Номер запису (ключ таблиці)	INT	
2	Price_Value	Значення ціни	FLOAT	
3	Price_Date	Дата/час встановлення ціни	DATETIME	
4	Cargo_Id	До якоготовару відоситься	INT	

SQL запит для створення таблиці:

CREATE TABLE Prices (Id_price INT, Price_Value FLOAT, Price_Date DATETIME, Cargo_Id INT, PRIMARY KEY (Id_price));

Таблиця 4.5 – Описи Descriptions

№	Назва стовбця	Що зберігається	Тип даних	На які таблиці посилається
1	Id_description	Номер запису (ключ таблиці)	INT	
2	Descr_Text	Текст опису	TEXT	

SQL запит для створення таблиці:

CREATE TABLE Descriptions (Id_description INT, Descr_Text TEXT, PRIMARY KEY (Id_description));

Таблиця 4.6 – Користувачі Users

№	Назва стовбця	Що зберігається	Тип даних	На які таблиці посилається
1	Id_user	Номер запису (ключ таблиці)	INT	
2	Username	Імя користувача	VARCHAR(50)	
3	Access_level	Рівень доступу	INT	
4	Md5_hash	Md5 Хеш паролю	CHAR(32)	

SQL запит для створення таблиці:

CREATE TABLE Users (Id_user INT, Username VARCHAR(50), Access_level INT, Md5_hash CHAR(32), PRIMARY KEY (Id_user));

Таблиця 4.7 – Фотографії Photos

№	Назва стовбця	Що зберігається	Тип даних	На які таблиці посилається
1	Id_photo	Номер запису (ключ таблиці)	INT	
2	Photo_file	Ссылка на файл с фото	VARCHAR(255)	

SQL запит для створення таблиці:

CREATE TABLE Photos (Id_photo INT, Photo_file VARCHAR(255), PRIMARY KEY (Id_photo));

Таблиця 4.8 – Реєстрація входів в програму та виходів з программи Enterance

№	Назва стовбця	Що зберігається	Тип даних	На які таблиці посилається
1	Id_enterance	Номер запису (ключ таблиці)	INT	
2	User_Id	Користувач, який здійснил дію	INT	Users (Id_user)
3	EDate	Дата та час дії	DATETIME	
4	EnType	Тип дії (1 – вхід, 2 – вихід, 3 – зміна користувача)	INT	

SQL запит для створення таблиці:

CREATE TABLE Enterance (Id_enterance INT, User_Id INT, EDate DATETIME, EnType INT, PRIMARY KEY (Id_enterance), FOREIGN KEY (User_Id) REFERENCES Users(Id_user));

Таблиця 4.9 – Вагові термінали Terminals

№	Назва стовбця	Що зберігається	Тип даних	На які таблиці посилається
1	Id_Term	Номер запису (ключ таблиці)	INT	
2	TermName	Назва терміналу	VARCHAR(50)	
3	DllName	Імя бібліотеки для роботи з терміналом	VARCHAR(50)	
4	TermDescr	Опис	INT	Descriptions (Id_description)
5	TermPhoto	Фото терміналу	INT	Photos (Id_photo)

SQL запит для створення таблиці:

CREATE TABLE Terminals (Id_Term INT, TermName VARCHAR(50), DllName VARCHAR(50), TermDescr INT, TermPhoto INT, PRIMARY KEY (Id_Term), FOREIGN KEY (TermDescr) REFERENCES Descriptions (Id_description), FOREIGN KEY (TermPhoto) REFERENCES Photos (Id_photo));

Таблиця 4.10 – Ваги (робочі місця) Scales

№	Назва стовбця	Що зберігається	Тип даних	На які таблиці посилається
1	Id_scale	Номер запису (ключ таблиці)	INT	
2	Scale_Name	Імя робочого місця	VARCHAR(80)	
3	COMNumb	Номер COM порту	INT	
4	BaudRate	Швидкість обміну даними з терміналом	INT	
5	ScaleAddr	Адреса терміналу	INT	
6	OutputFolder	Расташування файлів з вихідними даними (імя папки)	TEXT	
7	InputFolder	Розташування файлів нарядів (імя папки)	TEXT	
8	InputFName	Імя файла наряда	VARCHAR(200)	
9	Scale_Descr	Опис	INT	Descriptions (Id_description)
10	ScaleType	Тип терміналу, що використовується	INT	Terminals (Id_Term)
11	COMDisplay	Номер COM порта для виносного табло	INT	
12	ScaleID	ID для работы по радіоканалу	CHAR(5)	
13	Activity	Активність (підключені чи ні до системи)	INT	
14	ModeType	Режим роботы ваг (0 - набірний, 1 –штучний)	INT	

SQL запит для створення таблиці:

CREATE TABLE Scales (Id_scale INT, Scale_Name VARCHAR(80), COMNumb INT, BaudRate INT, ScaleAddr INT, OutputFolder TEXT, InputFolder TEXT,

InputFName VARCHAR(200), Scale_Descr INT, ScaleType INT, COMDisplay INT, ScaleID CHAR(5), Activity INT, ModeType INT, PRIMARY KEY (Id_enterance), FOREIGN KEY (Scale_Descr) REFERENCES Descriptions (Id_description), FOREIGN KEY (ScaleType) REFERENCES Terminals (Id_Term));

Таблиця 4.11 – Контроль версій (використовується, щоб можна було у клієнта оновити БД до нової версії в залежності від тої яка в нього стоїть) Version

№	Назва стовбця	Що зберігається	Тип даних	На які таблиці посилається
1	Id_Version	Номер запису (ключ таблиці)	INT	
2	Version_Numb	Номер версії	INT	
3	Version_Name	Імя версії	VARCHAR(50)	
4	Version_Date	Дата версії	DATETIME	

SQL запит для створення таблиці:

CREATE TABLE Version (Id_Version INT, Version_Numb INT, Version_Name VARCHAR(50), Version_Date DATETIME, PRIMARY KEY (Id_Version));

Таблиця 4.12 – Журнал дій (використовується для запису дій користувача) Logs

№	Назва стовбця	Що зберігається	Тип даних	На які таблиці посилається
1	Id_Log	Номер запису (ключ таблиці)	INT	
2	Elem_Name	Імя елементу над яким проводилась дія	VARCHAR(100)	
3	Elem_Event	Дія, що проводилась над елементом	VARCHAR(50)	
4	Event_Time	Час проведення дії	DATETIME	
5	User_Did	ІД користувача, що проводив дію	INT	Users(Id_user)

SQL запит для створення таблиці:

CREATE TABLE Logs (Id_Log INT, Elem_Name VARCHAR(100), Elem_Event VARCHAR(50), Event_Time DATETIME, User_Did INT, PRIMARY KEY (Id_Log), FOREIGN KEY (User_Did) REFERENCES Users(Id_user));

4.3 Написання програмного забезпечення для ПК

Розробка програми для персонального комп’ютера проводилася в середовищі розробки Delphi 7 поставляється виробником Borland (рис.).

Delphi - імперативний, структурований, об'єктно-орієнтована, високорівнева мова програмування з суворою статичної типізацією змінних. Основна область використання - написання прикладного програмного забезпечення.

Ця мова програмування є діалектом мови Object Pascal. Спочатку мова Object Pascal ставився до дещо іншої мови, який був розроблений у фірмі Apple в 1986 році групою Ларрі Теслера. Однак, починаючи з Delphi 7 в офіційних документах компанії Borland назву Delphi стало використовуватися для позначення мови, раніше відомого як Object Pascal.

На першому етапі розробки програми для персонального комп’ютера було розроблено інтерфейс.

Загалом програма складається з 18 форм (рис.4.2)

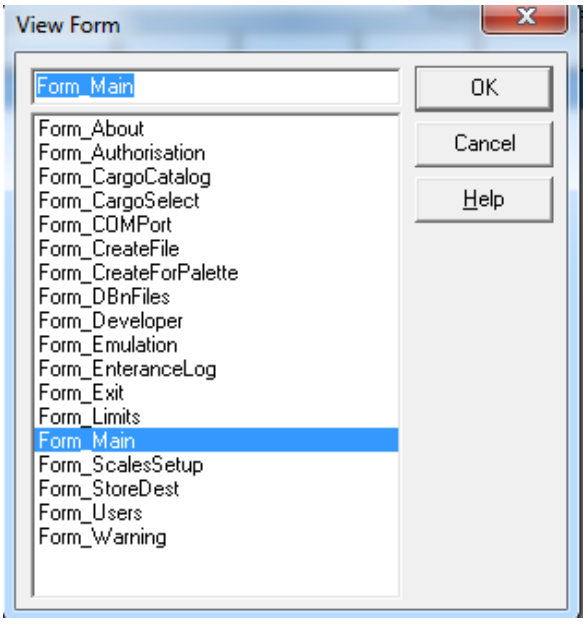


Рисунок 4.2 – Склад інтерфейсу програмного проекту

Основною є форма Form_Main, що виводить користувачеві основні параметри при роботі. Її зовнішній вигляд наведено на рис. 4.3, а структура на рис.4.4

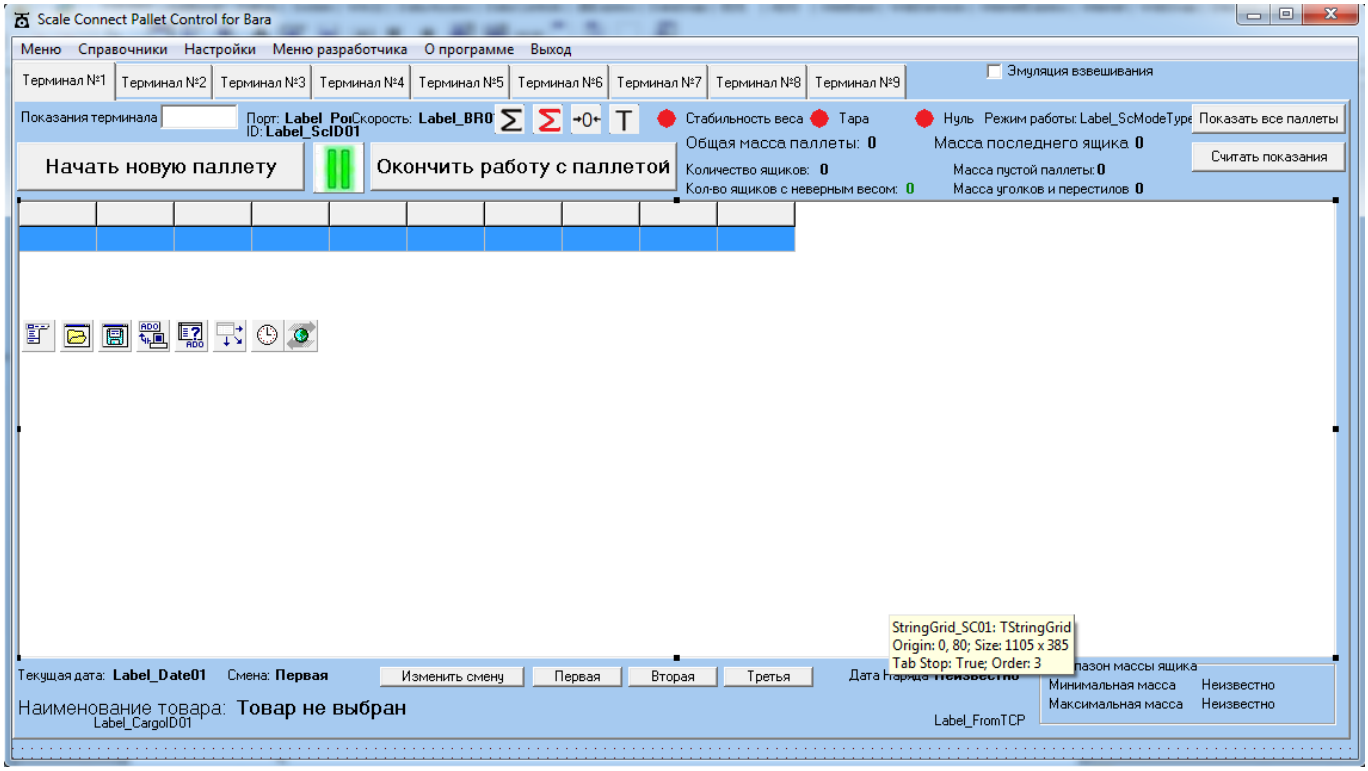


Рисунок 4.3 – Основна форма інтерфейсу програми

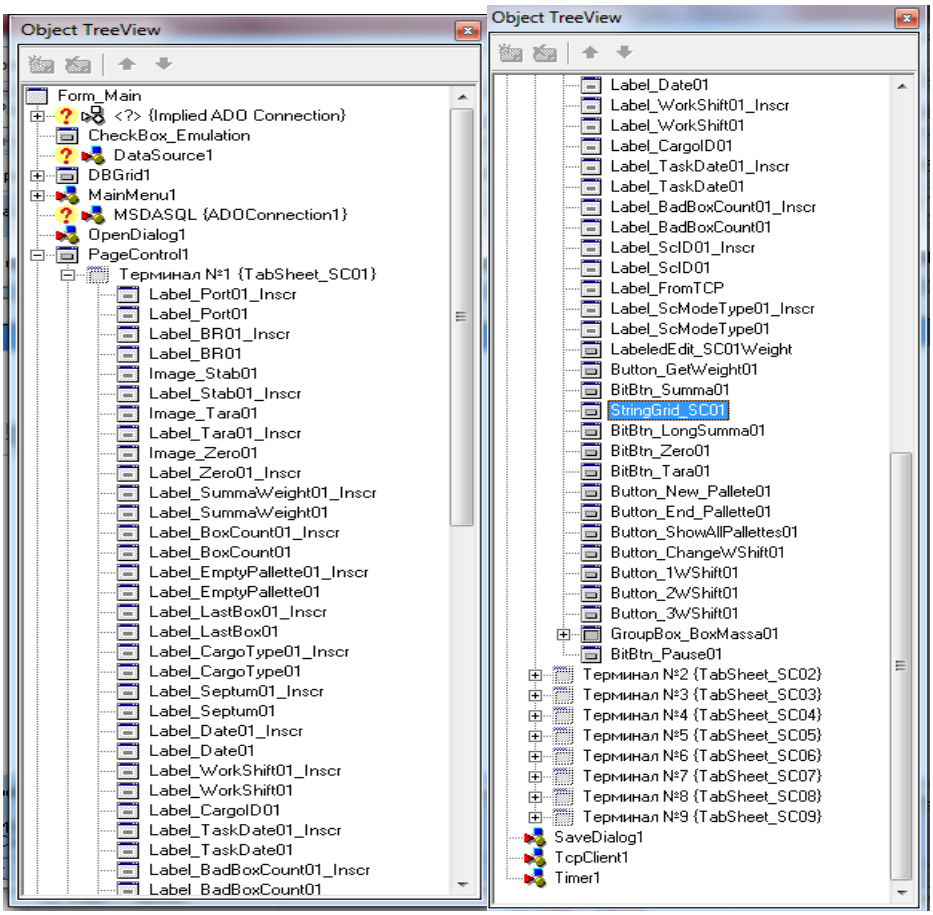


Рисунок 4.4 – Структура інтерфейсу основною форми інтерфейсу програми

Оскільки при розробці інтерфейсу на 18 формах було використано дуже багато елементів інтерфейсу – всі їх ми не наводимо у пояснювальній записці із-за браку місця.

Повний текст основного модуля написаної програми наведено у додатку Д

5 МЕТОДИКА РОБОТИ З СИСТЕМОЮ ТА ВИМОГИ ДО СИСТЕМИ

5.1 Вимоги до системи

5.1.1 Вимоги надійності

Ваговий термінал і персональний комп'ютер вимагають підключення до мережі живлення 220В. Для виключення виходу даного обладнання з ладу при перепадах напруги живлення на підприємстві необхідно використовувати джерело безперебійного живлення оптимально подвійного перетворення («Онлайн» тип). У разі відсутності даного обладнання на об'єкті замовника виконавець за вихід з ладу перерахованого обладнання через перепад напруги відповідальності не несе.

Крім того при відсутності ДБЖ (джерел безперервного живлення) на підприємстві від розетки до якої підключений комп'ютер не рекомендується подавати електроживлення на пристрої, що створюють при роботі великі імпульсні перешкоди в електричній мережі. Це може викликати збої в роботі комп'ютера і привести до втрати інформації.

Забороняється використовувати в якості заземлення водопровідні та газові труби, радіатори та інші вузли парового опалення.

5.1.2 Кліматичні умови експлуатації

Кліматичні умови експлуатації системи аналогічні кліматичних умов експлуатації самого незахищеного елемента системи тобто персонального комп'ютера.

Комп'ютер призначений для роботи в закритому опалювальному приміщенні при наступних умовах навколишнього середовища:

- температура навколишнього повітря від + 10 ° С до + 35 ° С;
- атмосферний тиск від 630 до 800 мм ртутного стовпа;
- відносна вологість повітря не більше 80%;
- запиленість повітря не більше 0,75 мг / м³;

- крім цього, в повітрі не повинно бути парів агресивних рідин і речовин, що викликають корозію.

5.1.3 Вимоги до електроживлення

Електроживлення здійснюється від однофазної мережі змінного струму напругою 220 В ± 10% і частотою 50-60 Гц.

Комп'ютер і його периферійні пристрої повинні підключатися до електричної мережі через спеціальні розетки, які мають заземлювальні контакти. Заземляющие контакти повинні забезпечувати надійне заземлення. Опір заземлюючого контуру має бути не більше 4 Ом.

5.1.4 Вимоги до кваліфікації та чисельності персоналу

Для роботи з одним екземпляром системи (одне робоче місце для організації зважування однієї палети з ящиками з продукцією) достатня наявність одного працівника володіє базовими навичками роботи з комп'ютером. Оптимально, щоб налаштування комплексу робочих місць (загальної кількості примірників системи на підприємстві) виробляв адміністратор системи, що має необхідний набір знань про номенклатуру товарів підприємства з усіма характеристиками, включаючи коди і ціни цих товарів (у разі потреби фінансового обліку в системі).

5.1.5 Вимоги до складу та параметрів технічних засобів

Перелік вагових терміналів з виявленням моменту стабілізації ваги і можливістю організації зв'язку з персональним комп'ютером (ПК) через інтерфейс RS-232 з якими повністю перевірена функціональність зв'язку. Це не означає, що система не буде працювати з іншими ваговими терміналами даного класу, а характеризує лише перелік терміналів, з якими на момент складання ТЗ вироблялося тестування підключення і виконавець може гарантувати повну функціональність процесу передачі стабілізованих даних:

- Keli XK3118T1;
- Keli A12;
- ESIT ART;

- CAS CI2001.

Для забезпечення захисту від електромагнітних полів з'єднувальний кабель виконується з екранованого багатожильного дроту.

5.1.6 Склад документації

Комплект документації системи включає:

- договір між замовником і виконавцем;
- технічне завдання;
- інструкція з експлуатації системи;
- акт прийому-передачі робіт по розробці, встановлення та налаштування системи.

5.2 Порядок роботи з системою

Для початку роботи системи необхідно виконати наступні етапи:

1). З'єднати платформу палетних ваг з електронним ваговим терміналом;

2) Провести калібрування вагового терміналу;

3) Поєднати ваговий термінал з влаштуванням додаткової індикації за допомогою з'єднувального шнура йде в комплекті з пристроєм додаткової індикації;

4) З'єднати за допомогою шнура коробку пристрою додаткової індикації (в разі дротового з'єднання) або під'єднати до комп'ютера приймаючу коробку.

При з'єднанні за допомогою шнура 9ти контактний (COM) роз'єм підключається безпосередньо до COM порту мережі (в разі його наявності на системному блоці) комп'ютера або до будь-якого вільного USB через COM-USB перехідник або виконати монтаж плати розширення COM портів з подальшим з'єднанням шнура з COM роз'ємом плати розширення.

При з'єднанні по радіоканалу необхідно приймаючу коробку під'єднати шнуром в будь-який вільний USB роз'єм.

5) Встановити і налаштувати програмне забезпечення на ПК

З'єднання платформи з ваговим теміналі

В даний момент система дозволяє працювати з 2-ма видами електронних вагових терміналів:

- ВН (IE-04 ВН-1000-4);
- Keli (XK3118T1).

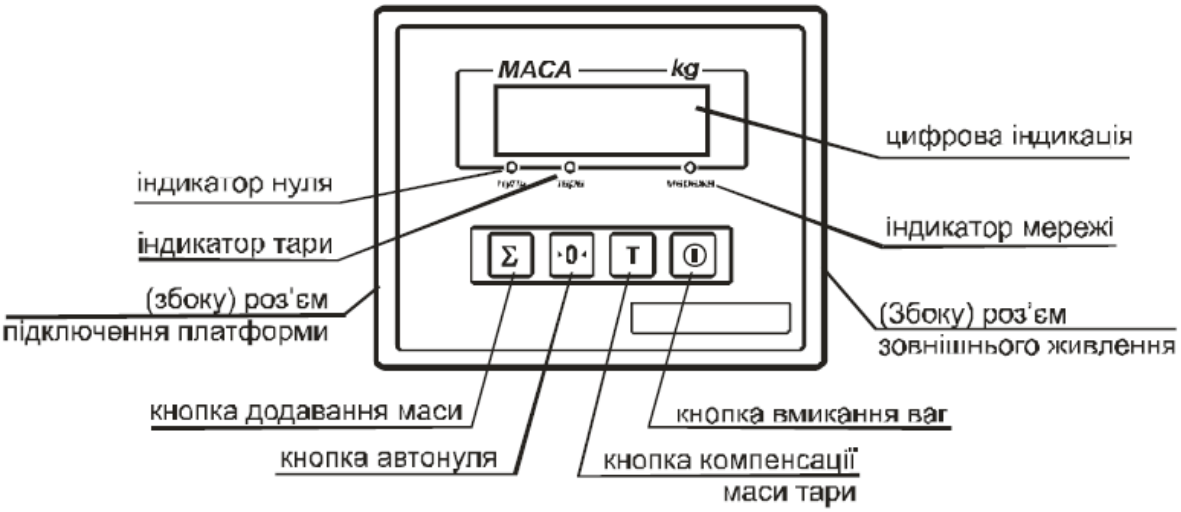


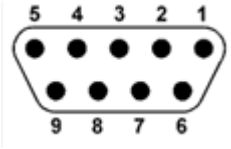
Рисунок 5.1 – Схематичний вигляд передньої панелі вагового терміналу ВН-1000-4

Рисунок 5.2 – Зовнішній вигляд передньої панелі вагового терміналу XK3118T1

Для підключення датчика (вагової платформи) до вагового терміналу ВН-1000-4 необхідно провести наступні дії:

Визначаємо дроти: живлять опір більше (приблизно 400 Ом), сигнальні приблизно 350 Ом. Якщо після підключення знак мінус - поміняти місцями сигнальні дроти - зменшується значення перевернути датчик.

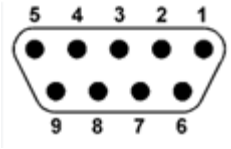
Таблиця 5.1 - З'єднання роз'єму вагового терміналу ВН-1000-4 з датчиком

Ваги	Датчик
 (шнур с разємом «Папа»)	
6– E+	Живлення +
7 – S-	Сигнал -
8 – S+	Сигнал +
9 – E-	Живлення -

Для підключення датчика (вагової платформи) до вагового терміналу ХК3118Т1 необхідно провести наступні дії:

Визначаємо дроти: живлять опір більше (приблизно 400 Ом), сигнальні приблизно 350 Ом. Якщо після підключення знак мінус - поміняти місцями сигнальні дроти - зменшується значення перевернути датчик.

Таблиця 5.2 - З'єднання роз'єму вагового терміналу ХК3118Т1 з датчиком

Ваги	Датчик
	
1 – E- (якщо дріт не довгий з’єднуємо з 2)	Живлення -
2 – F- (якщо дріт не довгий з’єднуємо з 1)	
5 – общий	Екрануюча опльотка (якщо є)
6 – E+ (якщо дріт не довгий з’єднуємо з 7)	Живлення +
7 – F+ (якщо дріт не довгий з’єднуємо з 6)	
8 – S-	Сигнал -
9 – S+	Сигнал +

Калібрування використовуваних вагових терміналів

В даний момент система дозволяє працювати з 2-ма видами електронних вагових терміналів:

- ВН (IE-04 ВН-1000-4);

- Keli (XK3118T1).

Калібрування терміналу ВН:

1) розбираємо термінал шляхом відкручування болтів по кутах на задній панелі;

2) включаємо термінал

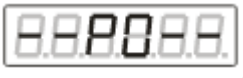
3) на друкованій платі вагового терміналу знаходимо кнопку (вона там єдина) і при включеному терміналі двічі (ДВА РАЗИ) її натискаємо - при цьому на індикаторах вагового терміналу з'являться шістнадцяткові коди вбудованого АЦП;

4) записуємо для подальших налаштувань код відповідний «0» тобто НЕ навантаженої платформі і як мінімум 1 код відповідний заздалегідь відомою масою (наприклад ставимо на платформу 300кг і записуємо значення - встановлюється маса повинна бути не менше 20% від максимальної межі зважування - при калібрування на 1000кг відповідно не менше 200кг);

5) вимикаємо термінал;

6) включаємо термінал;

7) При включеному харчуванні ОДИН раз натискаємо кнопку на платі з'явиться напис  ;

8) Затискаємо на 2-3 секунди кнопку  на передній панелі (термінал при цьому буде видавати пищали звук) до появи напису  ;

9) Кнопкою  входимо у зважування або кнопкою  виходимо з калібрування якщо всі крапки (яких може бути до 16ти P0-P15) вже виставлені. Якщо увійшли у зважування висвітиться напис  і через 2-3 секунди з'явиться значення маси яка буде покладена на ваги. Для точки P0 значення маси потрібно виставити 000000 тобто нульове. Для переходу між розрядами праворуч-

ліворуч використовується кнопка [0] для збільшення значення поточного розряду використовується кнопка [Σ], для підтвердження введеного значення і переходу до зважування використовується кнопка [T].

10) У режимі P0 ваги повинні бути розвантажені, після підтвердження кнопкою [T] термінал виводить значення на виході АЦП. Термінал спочатку покаже напис [C0d] і через 2-3 секунди виведе значення (6 знаків в 16тірічной формі). Перевіряємо щоб код збігався з тим що був записаний при розвантажених вагах і записаний нами в п.4. підтверджуємо кнопкою [T] і переходимо до наступної точки. Починаючи з точки P1 необхідно встановлювати і зважувати масу відмінну від нуля - також задаємо масу і відповідні значення записані в п.4

11) Для калібрування досить вказівки 2х точок (P0 і P1) якщо є впевненість що датчик лінійний

Калібрування терміналу XK3118T1:

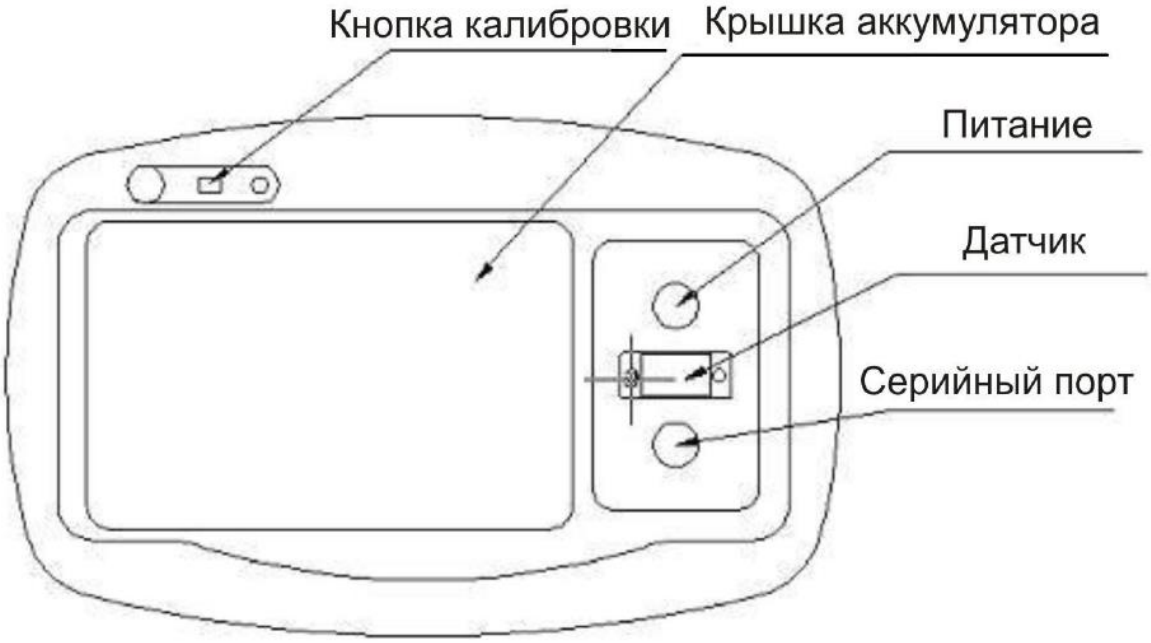


Рисунок 5.3 – Розташування елементів на задній панелі терміналу XK3118T1

1) При включеному терміналі натискаємо на задній панелі кнопку калібрування (кнопка може бути захована під чорною кришкою з пломбувальних

болтом - в цьому випадку потрібно отримати до неї доступ шляхом вигвинчування болта і зняття кришки);

2) Після натискання кнопки на екрані індикатора з'явиться напис «CAL»;

3) Натиснути кнопку «→**0**←» для наступного етапу.

4) Чи з'явиться напис з дискретністю «E 01». Це крок до якого термінал буде округляти свідчення. Дискретність вибирається з ряду 1, 2, 5, 10, 20, 50. Натискаємо кнопку «**Σ**» для вибору дискретності (для наших потреб використовується дискретність 5 - «05»).

5) Натиснути кнопку «→**0**←» для наступного етапу, на екрані індикатора з'явиться напис «dC 0.000» де кількість нулів після коми буде характеризувати точність показань (при кроці 50 г кнопкою «**Σ**» вибираємо режим «dC 0.00»);

6) Натиснути кнопку «→**0**←» для наступного етапу. З'явиться напис «F030.00» - найбільша границя зважування - шляхом натискань «Тара» і «Σ» встановлюємо 990,00;

7) Натиснути кнопку «→**0**←» для наступного етапу, на екрані індикатора з'явиться напис «noLoAd» - платформу розвантажуємо, щоб термінал запам'ятав стан порожньої платформи;

8) Натиснути кнопку «→**0**←» для наступного етапу, після показу протягом 2 секунд написи «AdLoAd», на дисплеї з'явиться маса контрольного вантажу. Шляхом натискання кнопок «Тара» і «**Σ**» встановлюємо масу вантажу яким будемо проводити калібрування (маса вантажу не повинна бути менше 20% від максимальної межі зважування тобто при 990 кг не менше 200кг), встановлюємо вантаж маса якого відповідає введеному значенню і натискаємо кнопку «→**0**←». Після цього на екрані терміналу з'явиться маса встановленого вантажу.

Налаштування режимів передачі даних для вагових терміналів

Для того щоб вагові термінали правильно видавали дані програмі їх необхідно налаштувати в правильний режим передачі даних.

Для настройки передачі даних терміналу ВН:

1) При включенні до закінчення проходження тесту необхідно натиснути будь-яку кнопку на лицьовій панелі при цьому на терміналі з'явиться напис  ;

2) За допомогою кнопок  та  виставляємо код 1723 (Для переходу між розрядами праворуч-ліворуч використовується кнопка  для збільшення значення поточного розряду використовується кнопка ). Після закінчення введення підтверджуємо кнопкою 

3) Чи з'явиться напис настройки першого параметра F0 з його поточним значенням, устанавліваем6

F0 - 1

F1 - адреса 31,32 або 33 (відповідний параметр повинен бути виставлений в програмі при її налаштування);

F2 - ШВИДКІСТЬ обміну 2

F3 - номер протоколу обміну 0

Інші параметри залишаємо без змін.

Для настройки передачі даних терміналу XK3118T1:

1). Для включення режиму передачі даних натискаємо кнопку «РЕЖ», утримуємо до звукового сигналу.

2). Вибираємо Fn Cot натискаємо «0» для підтвердження

3). Режим PS символом «Сума» (стрілка вгору) вибираємо OFF натискаємо «0» для підтвердження

4). Режим br символом «Сума» (стрілка вгору) вибираємо 9600 натискаємо «0» для підтвердження

5). Режим Co символом «Сума» (стрілка вгору) вибираємо 5 натискаємо «0» для підтвердження

6). Натискаємо «РЕЖ» для виходу з режиму налаштувань

При першій установці системи на персональний комп'ютер і підключення до комп'ютера платформи з ваговим індикатором в системі прописаний тільки один користувач з правами адміністратора і паролем за замовчуванням (логін "Admin", пароль "admin") і один користувач з правами оператора (логін "ОТК" , пароль "operator" - вводиться автоматично при запуску програми).

Для настройки системи треба зробити вхід в систему під адміністратором. Перший вхід в систему проводиться введенням даного пароля у відповідне поле (рис.5.4).

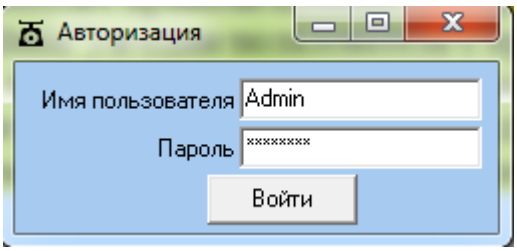


Рисунок 5.4 - Вікно входу користувача в програму

Адміністратор входить в меню налаштувань системи, натиснувши відповідну кнопку «Налаштування» (рис.5.5)

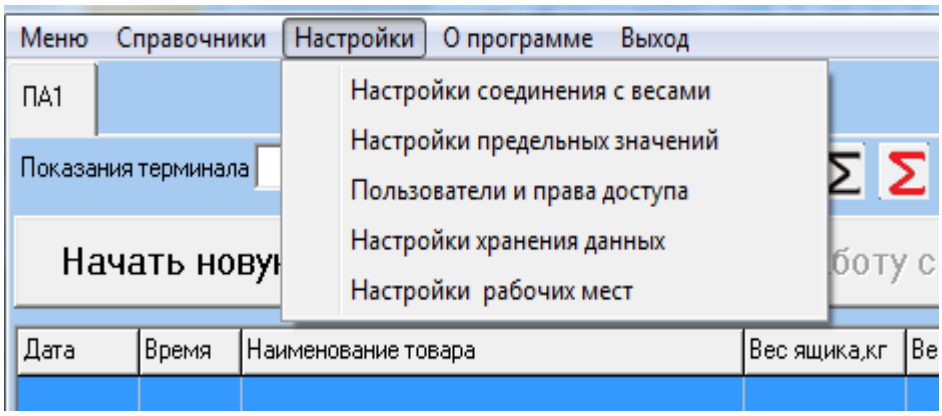


Рисунок 5.5 – Пункт меню налаштувань

У пункті «Налаштування граничних значень» використовуваному для настройки діапазону мас порожній палети, максимальної маси повної палети і її максимальної похибки, діапазону маси Переста, діапазону маси куточків необхідно ввести їх обчислення у відповідних полях (рис.5.6).

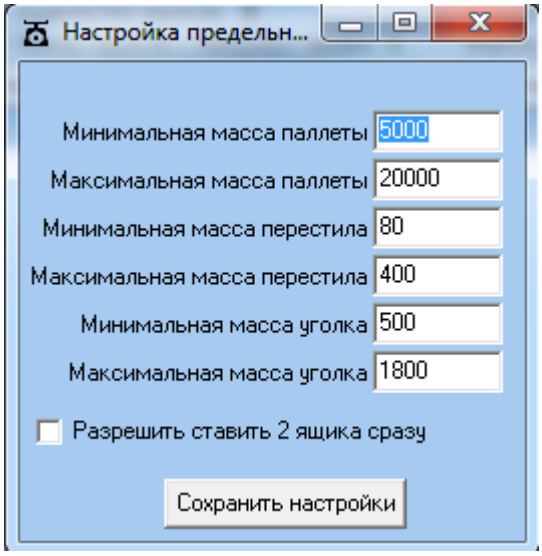


Рисунок 5.6 – Вікно для настройки граничних значень

Галочка «Дозволити ставити 2 ящика відразу» використовується для програмного поділу двох ящиків для випадків коли оператор виробничої лінії швидко ставить ящики або ставить їх по два відразу - для правильного підрахунку кількості встановлених ящиків.

Для настройки назву лінії і кількості підключаються до програми ліній використовується пункт «Налаштування» - «Налаштування робочих місць» (рис.5.7)

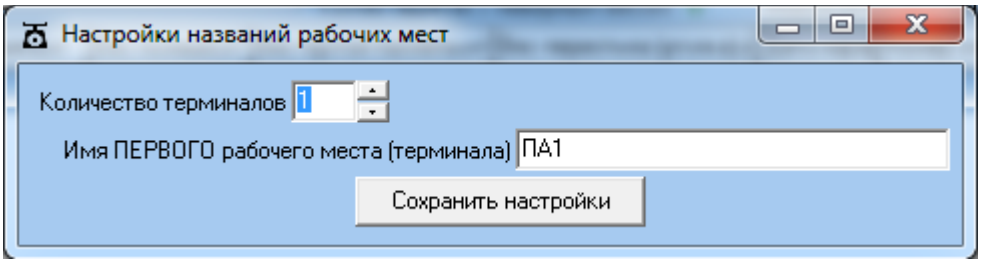


Рисунок 5.7 – Налаштування назв робочих місць

В даному вікні адміністратор вводить кількість вагових терміналів з яким буде працювати програма і назви кожного з робочих місць. При збільшенні кількості терміналів кількість полів з іменами робочого місця автоматично збільшується (рic.5.8)

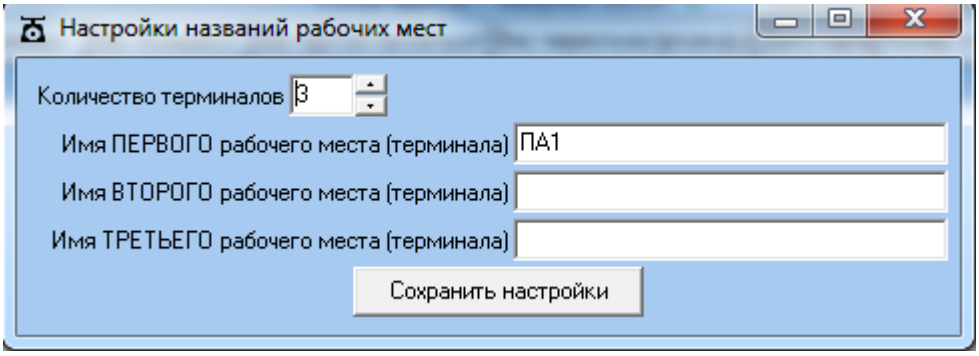


Рисунок 5.8 - Автоматичне збільшення кількості текстових полів з ім'ям робочих місць при збільшенні кількості вагових терміналів

Після збереження даних налаштувань в програмі зліва вгорі автоматично з'являться закладки з відповідною назвою робочих місць (рис.5.9).

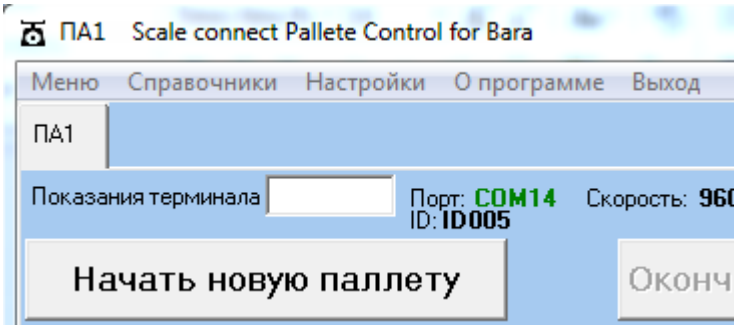


Рисунок 5.9 - Оновлення закладок після установки назв робочих місць

Для забезпечення отримання даних наряду на роботу ділянки і подальшого зберігання результатів роботи використовується меню «Налаштування» - «Налаштування зберігання даних» (рic.5.10)

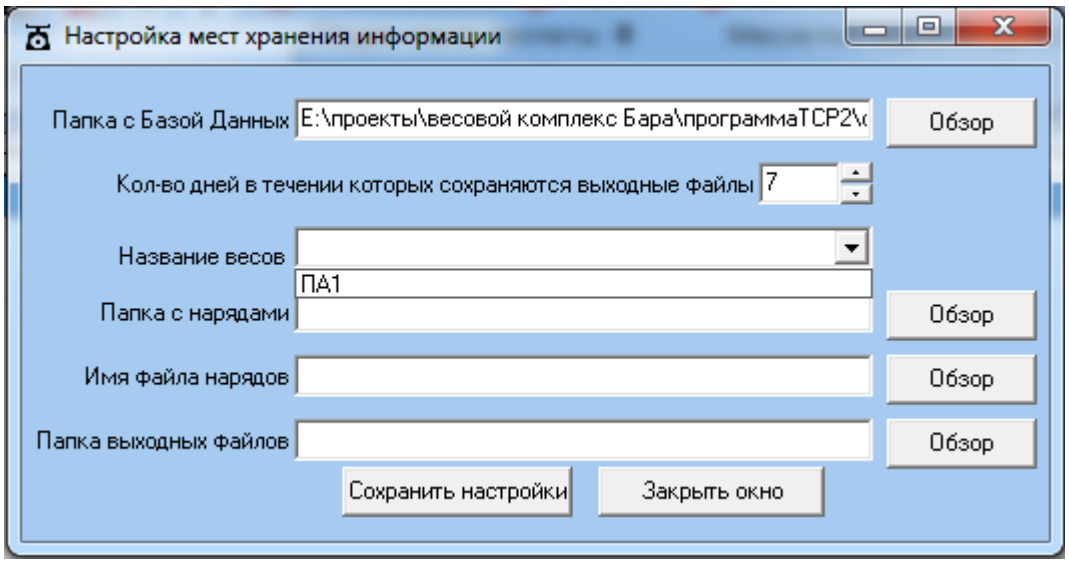


Рисунок 5.10 – Вікно з налаштуваннями для зазначення місця зберігання наряду і місць зберігання результатів роботи системи

У першому текстовому полі вказується місце зберігання бази даних. Це вікно заповнюється автоматично при запуску програми. В поле з назвою «Кількість днів протягом яких зберігаються вихідні файли» вказується кількість днів після яких програма автоматично очистить вихідний файл.

У вивалюється списку «Назва ваг» вибираємо назву робочого місця для якого проводиться настройка місця розміщення файлу з нарядами (поля «Папка з нарядами», «Файл нарядів») і папки з файлом «вес.txt» який є вихідним файлом роботи системи (поле «Папка вихідних файлів»).

Користувач вибирає в вивалюється списку назву робочого місця і через кнопку «Огляд» праворуч від відповідних текстових полів вибирає необхідні місця розміщення папок і файлів (рic.5.11).

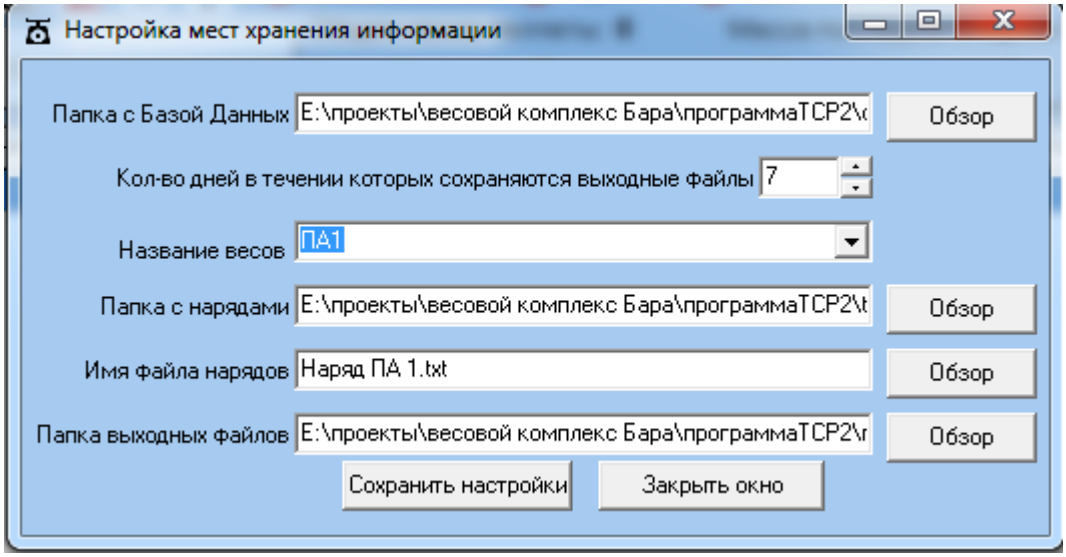


Рисунок 5.11 – Введення місць зберігання для обраного робочого місця на прикладі «ПА1»

Для настройки з'єднання з ваговим терміналом використовується пункт «Налаштування» - «Налаштування з'єднання з вагами» (рис.5.12)

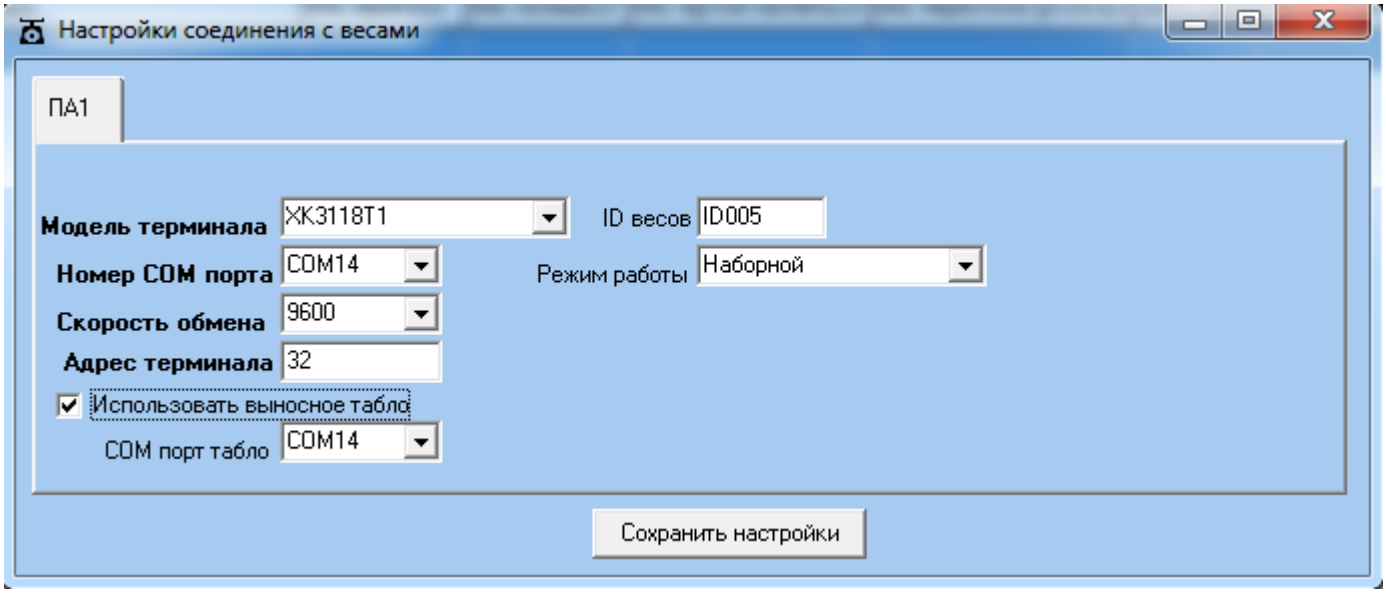


Рисунок 5.12 – Налаштування з'єднання з вагами

Програма автоматично створює кількість закладок відповідає кількості підключених і попередньо налаштованих робочих місць (в даному прикладі використовується одне робоче місце з ім'ям «ПА1»).

У випадаючому списку «Модель терміналу» адміністратор вибирає модель вагового терміналу для даного робочого місця (на прикладі використовується термінал XK3118T1).

У вивалюється поле «Номер COM порту» адміністратор вибирає номер послідовного порту до якого приєднана приймаюча коробка (в разі використання радіоканалу) або додатковий виносний індикатор (при підключенні по шнуру) - програма автоматично вкаже номера активних в системі COM портів.

У вивалюється списку «Швидкість обміну» вибирається швидкість обміну даними з пристроєм додаткової індикації (стоїть за умовчанням 9600 і з урахуванням того як налаштовані в попередніх пунктах вагові термінали цю швидкість НЕ ЗМІНЮВАТИ)

Поле «Адреса терміналу» використовується тільки при застосуванні вагового терміналу типу ВН-1000-4.

Оскільки в системі завжди використовується пристрій додаткової індикації необхідно встановити галочку «Використовувати виносне табло» і вибрати порт який буде збігатися з послідовним портом підключеної коробки.

В поле «ID ваг» адміністратор вказує ідентифікаційний номер зовнішнього виносного індикатора (даний номер прошивається в Arduino яке управляє зовнішнім виносним індикатором).

Вивалюється список «Режим роботи» служить для вибору режиму в якому ящики лягають на платформу ваг. Режим «Набірної», щоб виконувати завдання при якій на паллету ящик лягають послідовно (спочатку палета порожня, потім лягає 1й ящик, потім додається 2й, 3й, 4й і т.д.). Режим «Штучний» служить для одиночного контролю ящиків т.е.ложітся 1й ящик, потім він забирається, потім лягає 2й, потім він забирається і т.д.

Порядок роботи з програмним забезпеченням

Початок роботи.

Оператор вводить особистий пароль для ідентифікації в системі (логін і пароль оператора лінії введений в системі при запуску за замовчуванням - якщо є необхідність розрізняти операторів - необхідно кожному з них створити персональний логін і пароль).

Після вхід а в систему (якщо був правильно введений шлях до файлу наряду) внизу програми (рис.5.13) буде вказано найменування товару (у прикладі «Парус DAILY CMC Універсальний 9 кг (в ящиках)»), поточна дата (в прикладі «25.02.2019»), дата створення наряду (в прикладі« 02.07.18 »), номер зміни (в прикладі «Перша»), допустимий діапазон маси ящика (в прикладі«18000» та «18500»).

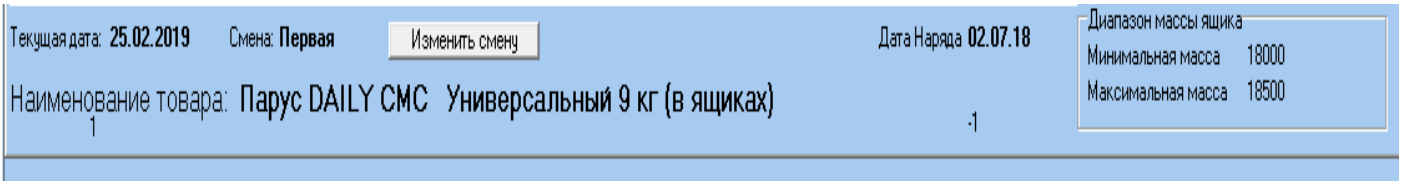


Рисунок 5.13 – Написи з найменуванням товару, поточною датою, датою створення наряду, номером зміни і допустимим діапазоном одного ящика

Для зміни номера зміни користувач натискає кнопку «Змінити зміну» і вибирає необхідну йому зміну (рic.5.14).

Рисунок 5.14 – Выбор смены

Оператор укладає на платформу порожню паллету. Початок роботи з палетою вказується шляхом натиснення відповідної кнопки на ПК «Нова паллету» в лівому верхньому кутку програми (рис.5.15).

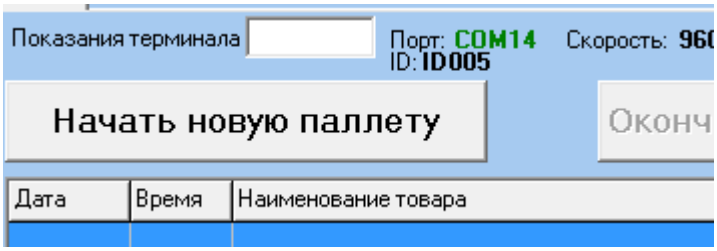


Рисунок 5.15 – Кнопка «Начать новую паллету»

Користувач послідовно вкладає на паллету, що стоїть на платформі, ящики з продукцією. Маса кожного покладеного ящика фіксується і контролюється системою на предмет відповідності налаштованого діапазону для даного виду продукції. Крім того на екрані монітора (у правому верхньому кутку програми) користувачу вказується сумарна маса ящиків з продукцією, сумарна маса знаходиться на платформі (порожня палета + маса ящиків + перестилаючи і куточки), кількість укладених ящиків (рис.5.16).

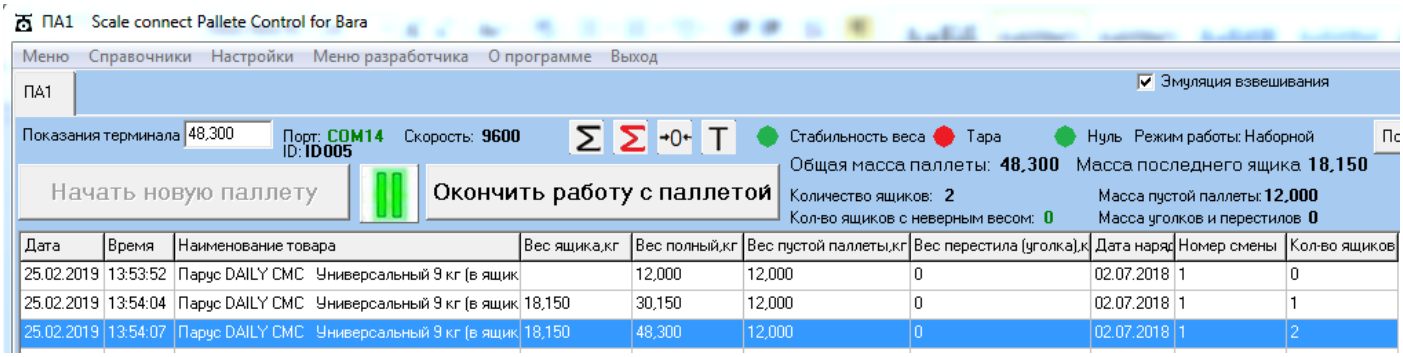


Рисунок 5.16 - Робота з палетою при укладанні ящиків

У разі, якщо черговий покладений ящик не відповідає встановленим для даного виду продукції діапазону мас - користувач отримує повідомлення про помилку у вигляді напису на моніторі (ящик з масою менше заданої буде підсвічений червоним кольором, а з масою більше заданої в наряді жовтим кольором см. Рис .5.17) і написи «ERROR» (рис.5.18) і звукового сигналу на виносному дублюючому табло. Після цього користувачеві пропонується зняти останній ящик для проведення контролю вмісту. При знятті ящика з палети система

автоматично розпізнає, що ящик був знятий для контролю і видаляє його з відображається на екрані монітора інформації.

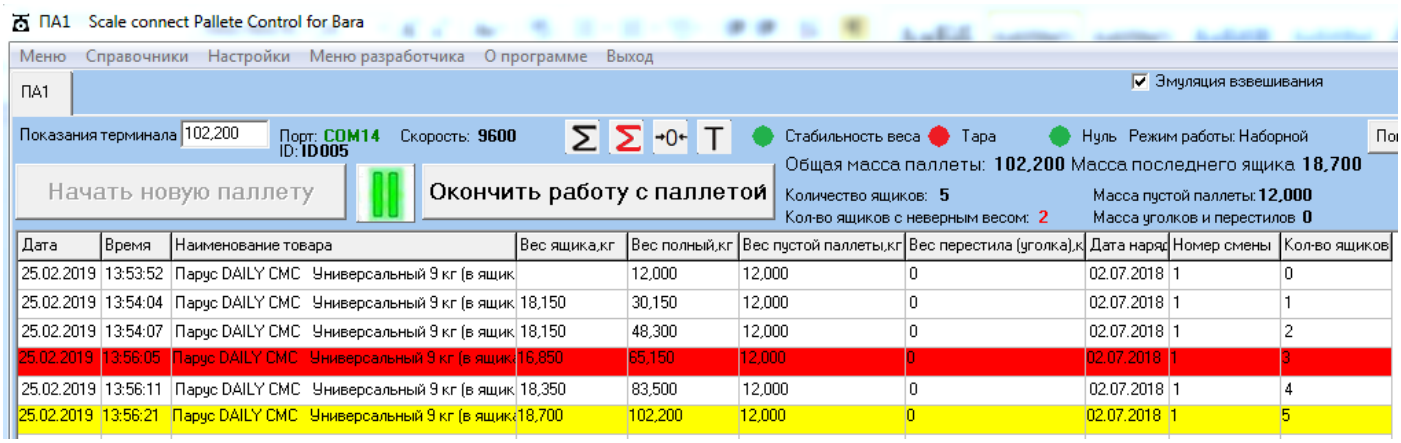


Рисунок 5.17 – Підсвічування в програмі ящиків виходять за встановлений в наряді діапазон

Рисунок 5.18 – Напис на виносному дублюючому табло в разі установки ящика з неправильним вагою

При укладанні перестилаючих і куточків система автоматично фіксує цей процес з урахуванням того, що діапазон мас перестилаючих і куточків значно відрізняється від маси ящиків і в налаштуваннях можна вказати після якого ящика для кожного виду продукції прийнято укладати перестилаючи і куточки.

6 ЕКОНОМІЧНЕ ОБҐРУНТУВАННЯ

Розроблена магістерська робота «Система контролю якості пакування та обліку кількості продукції підприємства з виробництва побутової хімії» призначена для контролю якості пакування на ТОВ «Бара» за ваговими показниками.

6.1 Визначення трудомісткості та тривалості

Весь комплекс моделювання можна розділити на етапи. Для кожного етапу вказуються трудомісткість, кількість виконавців і тривалість робіт. У розробці пристрою приймають участь радіотехнік протягом одного місяця і програміст протягом 0,5 місяця. Дослідження починається першого вересня і повинна бути виконана до п’ятнадцятого жовтня 2019 року. Тривалість робіт визначають за формулою 5.1:

$$T_{\text{ц}} = \frac{Q}{R} = \frac{45}{2} = 22{,}5 \tag{6.1}$$

де $T_ц$ - тривалість циклу, днів;

Q - трудомісткість, людино-днів;

R - кількість виконавців, чол.

Отримана інформація зведена в табл. 6.1

Таблиця 6.1 - Завдання та обов'язки по розробці пристрою

Найменування роботи	Трудомісткість		Виконавці	Тривалість, днів
	люд.-дні	%к підсумку		
1. Отримання технічного завдання	6	13,3	Програміст Радіотехнік	3 3
2. Огляд літературних джерел	5	11,1	Радіотехнік	5
3. Аналіз матеріалів та технологій	5	11,1	Радіотехнік	5
4. Розробка конструкції	10	22,2	Радіотехнік	10
5. Розробка програмного забезпечення	5	11,1	Програміст	5
6. Тестування пристрою та програмного забезпечення	10	22,2	Програміст Радіотехнік	5 5
7. Аналіз отриманих результатів	4	8,8	Програміст Радіотехнік	2 2
Разом	45	100		45

За даними табл. 6.1 складається зведений стрічковий графік планування розробки моделі, який представляє собою таблицю, в першому стовпці якої розміщені в порядку збільшення термінів початку виконання всі види роботи, а навпаки - календарний період їх виконання. Даний графік наведено в табл. 6.2.

Таблиця 6.2 - Зведений стрічковий графік планування розробки

Найменування робіт	Календарний період, дні						
	01.09-07.09.19	07.09-12.09.19	12.09-17.09.19	17.09-27.09.19	27.09-02.10.19	02.10-12.10.19	12.10-16.10.19
1. Отримання технічного завдання							
2. Огляд літературних джерел							
3. Аналіз матеріалів та технологій							
4. Розробка конструкції							
5. Розробка програмного забезпечення							
6. Тестування пристрою та програмного забезпечення							
7. Аналіз отриманих результатів							

- програміст

- радіотехнік

6.2 Визначення витрат на розробку пристрою

Для визначення витрат на розробку пристрою складається калькуляція вартісної вартості робіт, яка включає наступні статті:

- основна заробітна плата;
- додаткова заробітна плата;
- єдиний соціальний внесок (ЄСВ);
- витрати на спеціальне обладнання;
- матеріали і комплектуючі вироби;
- накладні витрати;
- податки.

6.2.1 Розрахунок основної заробітної плати

Витрати за цією статтею складаються з планового фонду зарплати всіх категорій працівників, зайнятих в розробці пристрою. Розрахунок зарплати ведеться на підставі даних про трудомісткості, представлених в табл. 6.3.

Таблиця 6.3 - Розрахунок основної заробітної плати

Посада виконавця	Чисельність, чол.	Місячний оклад, грн.	Кількість місяців роботи	Сума ЗП, грн.
Радіотехнік	1	6000	1	6000
Програміст	1	8000	0,5	4000
Разом	2			10000

6.2.2 Розрахунок додаткової заробітної плати

Додаткову заробітну плату приймають рівною 10% від основної заробітної плати працівників і розраховують за формулою 5.2:

$$ЗП_{доп} = ЗП_{осн} \cdot 0{,}1 \quad (6.2)$$

Підставивши величину основної заробітної плати в формулу 5.2, отримуємо:

$$ЗП_{доп} = 10000 \cdot 0{,}1 = 1000 \text{ грн}$$

6.2.3 Відрахування на єдиний соціальний внесок

Вони становлять 22% і беруться від основної та додаткової заробітної плати.

$$ОТ = (ЗП_{осн} + ЗП_{доп}) \cdot 0{,}22 \quad (6.3)$$

$$ОТ = (10000 + 1000) \cdot 0{,}22 = 2420 \text{ грн.}$$

6.2.4 Визначення затрат на матеріали

У цю статтю включають вартість основних і допоміжних матеріалів, напівфабрикатів, що купуються, і комплектуючих виробів. Транспортно-заготовчі витрати приймають рівними 3-10% від вартості матеріалів.

Таблиця 6.4 - Витрати на матеріали

Матеріали	Одиниця виміру	Кількість	Ціна за одиницю, грн.	Загальна вартість, грн.
Arduino Nano	шт.	1	95	95
Ethernet модуль ENC28J60	шт.	1	95	95
Модуль NRF24L01	шт.	1	30	30
Адаптер для NRF24L01	шт.	1	23	23
UART TTL - RS-485	шт.	1	35	35
Модуль RS232	шт.	1	130	130

Продовження таблиці 6.4

Матеріали	Одиниця виміру	Кіль кість	Ціна за одиницю, грн.	Загальна вартість, грн.
Интегральная схема MAX232	шт.	1	45	45
Модуль HX711	шт.	4	29	116
Стабилизатор AMS1117	шт.	1	7	7
Плата двошарова	шт.	1	20	20
Блок живлення	шт.	1	57	57
Разом				653
Транспортно-підготовчі роботи 5%				32,6
Разом матеріальних затрат				685,6

Витрати на комплектуючі розраховують за формулою 6.4:

$$M = \sum_{i=1}^{n} (Ц_i \cdot N_i \cdot (1 + K_{т.з.}) - Ц_{io} \cdot N_{io}), \quad (6.4)$$

де M - витрати на покупні напівфабрикати і комплектуючі вироби, грн.;

$K_{т.з.}$ - коефіцієнт, що враховує транспортно-заготівельні витрати;

$Ц_i$ - ціна і-го найменування напівфабрикату і комплектуючого, грн.;

N_i - потреба в і-му напівфабрикаті і комплектуючому;

$Ц_{io}$ - вартість зворотних відходів і-го найменування комплектуючого, грн.;

N_{io} - кількість зворотних відходів і-го найменування;

n - кількість найменувань напівфабрикатів і комплектуючих.

$Ц_{io} = 0$; $N_{io} = 0$; $K_{т.з.} = 0,05$;

М = (1 + 0,05)·(653) = 685,6 грн.

Разом, витрати на матеріали становлять 685,6 грн.

6.2.5 Витрати на спеціальне обладнання

У цю статтю входять витрати на придбання, транспортування, монтаж і налагодження нестандартного обладнання.

Практично, в даному випадку, в цій статті враховуються витрати на оплату машинного часу ЕОМ для розробки програмної частини. Для чого необхідно скласти кошторис «витрат на утримання і експлуатацію устаткування» виходячи з якої визначиться вартість одного машино-години роботи ПК, після множення якої на машинний час пішло на розробку програмної частини отримаємо витрати на оплату машинного часу. У табл. 6.5 наведене обладнання яке використовується при роботі.

Таблиця 6.5 – Спеціальне обладнання

Матеріали	Одиниця виміру	Кількість	Ціна за одиницю, грн.	Загальна вартість, грн.
Ноутбук Lenovo IdeaPad 330-15IKB	шт.	1	11000	11000
Artline Business B27	шт.	1	7800	7800
Стіл комп’ютерний	шт.	1	950	950
Крісло комп’ютерне	шт.	1	1100	1100
Разом				20850
Транспортно-підготовчі роботи 5%				1043
Разом				21893

Амортизаційні відрахування визначають за формулою 5.5:

$$A = Ф_б \cdot \frac{H_а}{100}, \quad (6.5)$$

де $Ф_б$ - балансова вартість обчислювальної техніки, грн .;

$H_а$ - норма амортизаційних відрахувань на повне відновлення обчислювальної техніки, для ПК 25%.

Балансова вартість обчислювальної техніки становить 21893 грн.

Отримуємо:

$$А = 21893 \cdot 0,25 = 5473,25 \text{ грн.}$$

Статтю «Експлуатація обладнання» розраховують підсумовуванням витрат на електроенергію і допоміжні комплектуючі.

$$C_е = N_н \cdot Ф_{еф} \cdot K_{зв} \cdot K_{зм} \cdot Ц_е, \quad (6.6)$$

де $N_н$ - номінальна потужність ЕОМ, кВт;

$Ф_{еф}$ - річний ефективний фонд часу роботи ЕОМ, машино-год;

$K_{зв}$ - середній коефіцієнт завантаження за часом;

$K_ш$ - коефіцієнт завантаження по потужності;

$Ц_е$ - ціна одного кВт-год електроенергії, грн./(кВт-ч).

Номінальна потужність ЕОМ - 0,5 кВт. Річний ефективний фонд часу роботи ЕОМ становить 1800 годин. Середні коефіцієнти завантаження за часом і за потужністю рівні відповідно 0,9 і 0,6. Ціна однієї кіловат-години електроенергії становить 2,68 грн.

Отримуємо:

$$C_е = 0,5 \cdot 1800 \cdot 0,9 \cdot 0,6 \cdot 2,68 = 1302,5 \text{ грн.}$$

Зарплата обслуговуючого персоналу розраховується за формулою 5.7:

$$ЗП_{обсл} = ФЗП_{г} \cdot (1 + К_{отч}) \cdot \frac{t_{обсл}}{Ф_{еф.обсл}} \quad (6.7)$$

де $ФЗП_г$ - річний фонд заробітної плати (основної і додаткової) обслуговуючих робітників, грн.;

$К_{отч}$ - коефіцієнт, що враховує відрахування на соціальне страхування і в інші фонди;

$t_{обсл}$ - час протягом року, необхідне на технічне обслуговування ЕОМ, ч/рік;

$Ф_{еф.обсл}$ - річний ефективний фонд часу обслуговуючого персоналу, ч/рік.

Приймаємо умовно, що місячна заробітна плата обслуговуючого персоналу становить 10000 грн., а річний фонд заробітної плати відповідно дорівнює 120000 грн. Річний ефективний фонд робочого часу обслуговуючого ПК працівника дорівнює 1750 год / рік.

$$ЗП_{обсл} = 120000 \cdot (1+0,22) \cdot 12/1750 = 1003,85 \text{ грн.}$$

Стаття «Поточний ремонт обладнання» приймається рівною 3% від балансової вартості обладнання і складає 600 грн.

Стаття «Інші витрати» приймається рівною п'яти відсоткам від суми всіх попередніх статей витрат на утримання і експлуатацію обладнання. Сума всіх попередніх статей дорівнює 7870 грн., 5% від суми складають 393,5 грн.

Розраховані статті витрат на утримання і експлуатацію устаткування внесені в табл. 6.6.

Витрати на оплату машинного часу ЕОМ для розробки програмної частини і налагодження програмних засобів визначаються за формулою 5.8:

$$C_{мо} = P_{екс} \cdot t_{мо}, \quad (5.8)$$

де $C_{мо}$ - витрати на оплату машинного часу, грн.;

$P_{екс}$ - експлуатаційні витрати на одну годину машинного часу цієї цифрової ЕОМ, грн. / машино-год.;

$t_{мо}$ - машинний час цифрової ЕОМ для написання і налагодження даного програмного продукту, машино-год.

Таблиця 6.6 - Кошторис витрат на утримання і експлуатацію устаткування

Найменування статей витрат	Сума, грн.
Амортизація обладнання	5473,25
Експлуатація обладнання (крім витрат на поточний ремонт)	1302,5
Заробітна плата основна і додаткова обслуговуючих робітників з ЄСВ	1003,85
Поточний ремонт обладнання	600
Інші витрати	393,5
Разом	8773

Експлуатаційні витрати на одну годину машинного часу використовуваної ЕОМ розраховують діленням суми витрат за кошторисом «Витрати на утримання та експлуатацію обладнання (ЕОМ)» (табл. 6.4) на річний ефективний фонд часу роботи ЕОМ. Річний ефективний фонд часу роботи ЕОМ дорівнює 1800 годин. В результаті експлуатаційні витрати на одну годину машинного часу рівні:

$$P_{екс} = 8773/1800 = 4{,}87 \text{ грн./машино-год}$$

ЕОМ експлуатується 45 днів в одну зміну, що становить в сумі 360 годин. Таким чином, витрати на оплату машинного часу складуть:

$$С_{мо} = 4{,}87 \cdot 360 = 1754{,}6 \text{ грн.}$$

6.2.6 Інші прямі витрати

В інші прямі витрати включаються витрати на яке використовується при розробці системи комерційне програмне забезпечення:

- дольове ПЗ, що використовується постійно при роботі ПК (Windows 10) - 5000 грн. без НДС;

- цільове ПЗ, що купується для даного конкретного завдання (Компас-Электрик) - 5000 грн. без НДС.

$$S_{дол.ПЗ} = \frac{Ц_{ПЗWindows} \cdot T_{КТС}}{Ф_{еф.КТС} \cdot T_{с.ПЗ}} \quad (6.9)$$

$$S_{ціл.ПЗ} = Ц_{ПЗ\,А}$$

де $S_{дол.ПЗ}$ - витрати на дольове ПЗ при розробці програмної частини розробляється в розрахунку ПЗ, грн .;

$S_{ціл.ПЗ}$ - витрати на цільове ПЗ, що купується виключно для розробки програмної частини в розрахунку ПЗ, грн .;

$Ц_{ПЗWindows}$ - ціна ПЗ Windows (без ПДВ), грн;

Ц$_{ПЗ\ А}$ - ціна ПЗ Компас-Электрик (без ПДВ), грн;

$T_{КТС}$ - машинний час КТС, необхідне користувачеві для розробки програмної частини, машино-год / рік;

$Ф_{еф.ктс}$ - річний ефективний фонд часу роботи КТС, машино-год / рік;

$T_{с.ПЗ}$ - термін служби дольової ПЗ, років.

$$S_{\text{доп.ПЗ}} = \frac{10000 \cdot 360}{1800 \cdot 5} = 400\text{грн.}$$

$$S_{ціл.ПЗ} = 10000 \text{ грн.}$$

$$S_{\Sigma} = 400 + 10000 = 10400 \text{ грн.}$$

6.2.7 Розрахунок загальновиробничих витрат

До загальновиробничих витрат відносяться витрати на загальне управління і загальногосподарські потреби (заробітна плата апарату управління, канцелярські витрати і т.д.), утримання та експлуатацію будівель. Загальновиробничі витрати включаються до вартості розробки пристрою непрямим шляхом - у відсотках до основної заробітної плати розробників. В даному випадку накладні витрати становлять 40% до основної заробітної плати розробників, що складає (10000 грн. х 0,4 = 4000 грн).

Результати визначення витрат на розробку пристрою у вигляді калькуляції кошторисної вартості робіт наведені в табл. 6.7.

Таблиця 6.7 - Калькуляція кошторисної вартості робіт з розробки пристрою

№	Найменування статей витрат	Сума, грн.	Питома вага до підсумку,%
1	Основна заробітна плата	10000	19,84
2	Додаткова заробітна плата	1000	1,98
3	ЄСВ	2420	4,8
4	Матеріали	685,6	1,36
5	Витрати на спец. обладнання	21893	43,44
6	Інші прямі витрати	10400	20,64
7	Накладні витрати	4000	7,94
8	Разом ($S_{пр}$)	50398,6	100

5.3 Розрахунок техніко-економічної ефективності моделі

Для теоретичних досліджень у більшості випадків важко чи навіть неможливо розрахувати економічний ефект, тому доцільно визначити їхню техніко-економічну ефективність з урахуванням наступних показників:

- важливості дослідження для народного господарства;
- складності розробки;
- результативності й можливості використання.

Важливість теоретичного дослідження оцінюємо як пошук принципово нових конструктивних і технологічних рішень і ін.

Результативність НДР визначається по повноті рішень поставленого завдання: отриманий результат відповідає планованому, задовільний (часткове рішення) чи негативний.

Аналіз залежності між цими показниками й витратами на їхнє досягнення дає можливість кількісної оцінки техніко-економічної ефективності теоретичних НДР і визначається за формулою (6.10):

$$K_{НДР} = \frac{J^n \cdot R \cdot T}{B_{НДР} \cdot t_{НДР}}, \qquad (6.10)$$

де $K_{НДР}$ - рівень ефективності дослідження (коефіцієнт техніко-економічної ефективності НДР):

J^n - важливість роботи;

R - результативність роботи;

T - технічна складність виконання НДР;

$B_{НДР}$ - витрати на проведення НДР, років:

n - показник використання результатів НДР:

n = 0 - результати НДР не використовуються;

n = 1 - результати НДР використовуються частково;

n = 2 - результати НДР використовуються в дослідно-конструкторських роботах (ДКР);

n = 3 - результати НДР можуть бути використані без проведення ДКР.

Для НДР, у яких $B_{НДР}$ >30 тис. грн. і $t_{НДР} \leq 2$ років, можна застосовувати такі значення оцінних факторів наведених в табл. 6.8

Таблиця 6.8 – Значення оцінних факторів

Оцінні фактори	J	R	T	C	$t_ф$	n
Припустимі значення	2…5	1…4	1…3	-	-	1…8
Прийняті значення	4	3	2	-	-	3

Згідно значень з таблиці оцінних факторів, отримуємо такий вираз:

$$К_{ндр} = \frac{4^3 \cdot 3 \cdot 2}{50398{,}6 \cdot 0{,}03} = 1{,}12$$

Таким чином, так як коефіцієнт техніко-економічної ефективності НДР $K_{НДР} \geq 1$, в нашому випадку рівний $K_{НДР}$ = 1,12, то дослідницька робота вважається ефективною.

7 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА У НАДЗВИЧАЙНИХ СИТУАЦІЯХ

7.1 Аналіз потенційних небезпек

Тема дипломного проекту «Система контролю якості пакування та обліку кількості продукції підприємства з виробництва побутової хімії», тому розглянемо робоче місце інженера радіотехніка.

Основними потенційними небезпеками при проведенні робіт у лабораторії є такі:

- небезпека ураження електричним струмом, внаслідок недотримання правил електробезпеки або виходу з ладу електроприладів;
- порушення роботи кістково-м’язового апарату внаслідок тривалих статичних навантажень при роботі з ПК.
- нервово-психічні перевантаження внаслідок постійного контакту з клієнтами, колегами по роботі, керівництвом при вирішенні робочих питань, які можуть носити конфліктний характер і призвести до емоційного дискомфорту, внутрішнього роздратування, емоційної нестабільності та захворювань нервової системи;
- незадовільні ергономічні характеристики робочого місця внаслідок нераціонального планування робочого місця, що може призвести до механічних травм, уражень електричним струмом та порушень кістковом’язового апарату;
- негативний вплив недостатнього освітлення робочої зони на зір та продуктивність роботи працюючого, внаслідок несправності освітлювальних приладів або неправильного проектування освітлювальної системи;
- негативний вплив незадовільних параметрів повітряного середовища робочої зони на здоров’я працюючого, внаслідок неправильного проектування системи вентиляції або несправності її несправності;
- негативний вплив підвищеного рівня шуму на психоемоційний стан працюючого, який пов’язаний з використанням застарілої периферійної техніки, кондиціонерів, копіювальної техніки, освітлювальних приладів;

- небезпека загоряння у зв'язку із несправністю електричного обладнання, недотримання, або порушення правил протипожежної безпеки обслуговуючим персоналом, що може призвести до пожежі.

Приведення цих чинників у відповідність до сучасних норм, нормативів і стандартів є передумовою нормальної працездатності людини.

Небезпечні і шкідливі наслідки для людини від дії електричного струму, електричної дуги, електричного і магнітного полів, електростатичного поля і ЕМВ проявляються у вигляді електротравм, механічних пошкоджень і професійних захворювань. Ступінь впливу залежить від експозиції фактора, в тому числі: роду і величини напруги і струму, частоти електричного струму, шляху струму через тіло людини, тривалості впливу електричного струму або електричного і магнітного полів на організм людини, умов зовнішнього середовища.

7.2 Заходи по забезпеченню техніки безпеки

Приміщення лабораторії, в яких перебувають співробітники галузі радіотехніка, відносяться до приміщень без підвищеної небезпеки ураження електричним струмом.

Організаційні та технічні заходи щодо забезпечення електробезпеки (ГОСТ 12.1.019-79) полягають у відповідному навчанні, інструктажі і допуску до роботи осіб, дотриманні особливих вимог при роботах на струмопровідних частинах, що знаходяться під напругою, або поблизу них.

Важливим заходом, що забезпечує електробезпеку обслуговуючого персоналу, є захисне заземлення або занулення металевих неструмоведучих частин електрообладнання. Відповідно до «Правил улаштування електроустановок» застосовується захисне заземлення, виконане для забезпечення електробезпеки, їм називається навмисне металеве з'єднання із заземлюючим пристроєм елементів електроустановок, які нормально не знаходяться під напругою. Занулення в електроустановках і мережах напругою до 1000 В - це навмисне електричне з'єднання металевих елементів установки, нормально ізольованих від частин, що знаходяться під напругою з глухозаземленою нейтраллю генератора або

трансформатора в мережах змінного струму, а також з глухозаземленою середньою точкою в трьохпровідних мережах постійного струму з нульовим проводом.

Основними причинами поразки персоналу електричним струмом є:

- дотик до струмоведучих частин, що знаходяться під напругою (оголені кабелі живлення, розетки - 220В);

- дотик до металевих неструмоведучих частин електроустановок або пов'язаного з електроустановками виробничого обладнання (корпусу, кожуха, огорожі і т. Д.) Після переходу на них напруги з струмоведучих частин.

Захисні заходи від ураження електричним струмом:

1) організаційні заходи:

- спеціальне навчання персоналу;

- інструктажі персоналу;

- призначені особи відповідальної за безпеку;

- проводяться періодичні огляди, вимірювання та випробування електрообладнання.

2) технічні заходи:

- застосовані пристрої (запобіжники, що вимикають реле і т. д.) Захист електрообладнання і мереж від перевантажень, а також струмів коротких замикань;

- захист людей від дотику до струмоведучих частин обладнання забезпечено за допомогою ізоляції струмоведучих частин електрообладнання;

- захист від ураження електричним струмом при переході напруги на металеві корпуси електроустановок здійснюється пристроєм захисного заземлення, занулення електрообладнання в мережах з глухо-заземленою нейтраллю, застосуванням захисного відключення.

7.3 Заходи по забезпеченню виробничої санітарії та гігієни праці

Площа приміщень, в яких розташовують персональні комп'ютери, визначають згідно з чинними нормативними документами. Згідно ДСанПіН 3.3.2.007-98 з розрахунку на одне робоче місце, обладнане ПК, встановлені такі норми:

- площа - не менше ніж 6,0 м^2;

- обсяг - не менше ніж 20,0 м^3.

Для внутрішнього оздоблення інтер'єру приміщень з комп'ютерами використовуються дифузно-відбивні матеріали з коефіцієнтами відбиття для стелі - 0,7-0,8; для стін - 0,5-0,6; для підлоги - 0,3-0,5.

Поверхня підлоги в приміщеннях експлуатації комп'ютерів рівна, без вибоїн, неслизька, зручна для очищення та вологого прибирання, має антистатичні властивості.

Приміщення для роботи з персональними комп'ютерами обладнані системами опалення, кондиціонування повітря і припливно-витяжною вентиляцією. У приміщеннях на робочих місцях забезпечуються оптимальні значення параметрів мікроклімату: температури, відносної вологості і рухливості повітря відповідно до норм і правил, а також ДБН В.2.5-67 діє до: 2013 «Опалення, вентиляція и кондіцююванння», затверджених наказом Мінрегіону від 25.01.2013 р № 24.

У лабораторії температура становлять 22-25 ° С, відносна вологість повітря - 40-60%, швидкість руху повітря - не більше 0,1 м / с. Що відповідно до санітарних норм мікроклімату виробничих приміщень ДСН 3.3.6.042-99, входить в межі норми.

Для підтримки допустимих значень мікроклімату та концентрації позитивних і негативних іонів передбачені установки або прилади зволоження та / або штучної іонізації, кондиціювання повітря.

Освітлення має важливе санітарно-гігієнічне значення. Зі збільшенням ступеня освітленості підвищується продуктивність праці (іноді на 15% і більше) і якість робіт, знижується виробничий травматизм і аварійність.

Робочі місця, згідно п. 4.3 ДСанПіН 3.3.2.007-98, розташовуються щодо світлових прорізів так, щоб природне світло падало переважно з лівого боку.

Робоче приміщення відповідає вимогам роботи високої точності з розрядом зорової роботи III, подразряд "В" за умовами ДБН В.2.5 - 28 - 2006 "Природне и штучне освітлення" освітлення повинно бути не менше 300 люкс.

Виходячи з отриманих даних, необхідно розрахувати оптимальні умови освітлення в приміщенні.

Вхідні дані: А = 12 м; В = 7 м; Н = 4 м; $E_н$ = 200 лк; $h_р$ = 0,8 м; z = 1,1; $k_з$ = 1,5.

Для світлих приміщень

Розрахунок кількості рядів світильників у приміщенні

$$N_p = \frac{B}{(H - h_p) \cdot \left[\frac{L}{h}\right]} = \frac{7}{(4 - 0{,}9) \cdot [1{,}4]} = 2 \text{ шт.} \quad (7.1)$$

Коефіцієнт світильника [L / h] = 1,4.

Розрахунок числового значення індексу приміщення:

$$i = \frac{A \cdot B}{h \cdot (A + B)} = \frac{12 \cdot 7}{3{,}2 \cdot (12 + 7)} = 1{,}38 \quad (7.2)$$

де h – висота підвісу світильника над робочою поверхнею, м.

$$h = \frac{B}{N_p \cdot \left[\frac{L}{h}\right]} = \frac{3{,}5}{1{,}4} = 2{,}5 \text{ м.} \quad (7.3)$$

Розрахунок максимальної відстані між рядами і сусідніми світильниками в ряду:

$$L_{max} = \frac{B}{N_p} = \frac{7}{2} = 3{,}5 \text{ м.} \quad (7.4)$$

При установці по 2 лампи в кожен світильник необхідно 8 світильників.

Рівні шуму на робочих місцях користувачів персональних комп'ютерів не перевищують значень, встановлених СанПіН 2.2.4 / 2.1.8.562-96 і становлять не більше 50 дБА.

Зниження рівня шуму в приміщеннях можливе використанням звукопоглинальних матеріалів з максимальними коефіцієнтами звукопоглинання в області частот 63-8000 Гц для обробки стін і стелі приміщень. Додатковий звуковбирний ефект створюють однотонні фіранки з щільною тканини, повішені в складку на відстані 15-20 см від огорожі. Ширина фіранки в 2 рази більше ширини вікна.

Робочий стіл відповідає сучасним вимогам ергономіки і дозволяє зручно розмістити на робочій поверхні обладнання з урахуванням його кількості, розмірів і характеру виконуваної роботи. Також застосовуються столи, які мають окрему від основної стільниці спеціальну робочу поверхню для розміщення клавіатури. Висота столів знаходиться в межах від 680 до 800 мм.

Глибина робочої поверхні столу становить 600-800 мм, ширина - відповідно 1200-1600 мм. Робоча поверхня стола не має гострих кутів і країв, а також має матову або напівматову фактору.

Робочий стіл має простір для ніг висотою не менше 600 мм, шириною - не менше 500 мм, глибиною на рівні колін - не менше 450 мм і на рівні витягнутих ніг - не менше 650 мм.

7.4 Заходи з пожежної безпеки

Пожежна безпека забезпечується системою запобігання пожежі і системою пожежного захисту. У всіх службових приміщеннях обов'язково повинен бути «План евакуації людей при пожежі», що регламентує дії персоналу в разі виникнення вогнища і вказує місця розташування пожежної техніки.

Продумуючи евакуацію співробітників при можливій пожежі, враховують особливості лабораторного приміщення. На плані позначені шляхи евакуації до сходових кліток або виходу на вулицю. Також зображений запасний шлях, по якому можна швидко покинути будівлю. На плані також вказують, де розташовані ручні пожежні сповіщувачі через часті переноси і перепланування, вони домальовуються вручну.

Пожежі у лабораторіях становлять особливу небезпеку, тому що пов'язані з великими матеріальними втратами. Характерна особливість таких лабораторії - невеликі площі приміщень. Як відомо, пожежа може виникнути при взаємодії горючих речовин, окислювача і джерел запалювання. У приміщеннях лабораторії присутні всі три основні чинника, необхідні для виникнення пожежі.

Приміщення з ЕОМ оснащені системою автоматичної пожежної сигналізації відповідно до вимог Переліку однотипних за призначенням об'єктів, які підлягають

обладнанню автоматичними установками пожежогасіння та пожежної сигналізації, затвердженого наказом МВС України від 20.11.97 №779 (z0567-97) і зареєстрованого в Міністерстві юстиції України 28.11.97 за №567 / 2371, і СНиП 2.04.09-84 «Пожежна автоматика будинків і споруд» з димовими пожежними сповіщувачами та переносними вуглекислотними вогнегасниками ВВК – 1.4 з розрахунку 2 шт. на кожні 20 $м^2$ площі приміщення з урахуванням граничнодопустимих концентрацій вогнегасної рідини відповідно до вимог Правил пожежної безпеки в Україні і вимог нормативно-технічної та експлуатаційної документації заводу-виробника.

Вогнегасники розташовуються на об'єкті, відповідно до вимог ГОСТ 12.4.009-83 таким чином, щоб вони були захищені від впливу прямих сонячних променів, будь-яких механічних впливів і інших несприятливих чинників, таких як вібрація, підвищена вологість та інших. Вогнегасники розміщуються в легкодоступних і помітних місцях. Не допускається зберігання і експлуатація вогнегасників в місцях, де температура може перевищувати 50 °С і під прямими променями сонця. Не рідше одного разу на рік перевіряється маса заряду вогнегасників, згідно з рекомендаціями. Експлуатація вогнегасників без чеки і пломби заводу-виготовлювача або організації, що проводила перезарядку, не дозволяється.

Під час монтажу та експлуатації ліній електромережі повністю виключені можливості виникнення електричного джерела загоряння внаслідок короткого замикання та перевантаження проводів, обмежено застосування проводів з легкозаймистою ізоляцією, по можливості застосовується негорюча ізоляція. У приміщенні, де одночасно експлуатуються понад п'яти комп'ютерів, на видному і доступному місці встановлений аварійний резервний вимикач, який може повністю відключити електричне живлення приміщення, крім освітлення.

7.5 Заходи по забезпеченню безпеки у надзвичайних ситуація

Рятувальні і невідкладні роботи при ліквідації наслідків стихійного лиха та аварії проводять рятувальні формування, а також залучено працездатне населення.

Роботи, що потребують спеціальної підготовки і залучення потужної техніки, виконують інженерні загони, створені на базі будівельних, ремонтно-будівельних, будівельно-монтажних, дорожньо-будівельних організацій і відділів комунальних господарств.

При стихійних лихах, які не є катастрофічними, всі заходи організовують керівники районів, об'єктів, населених пунктів і начальники загонів служби порятунку.

Начальник цивільного захисту (району, населеного пункту або об'єкта) приймає рішення про проведення рятувальних і невідкладних робіт, в якому вказує: де зосередити основні зусилля, завдання рятувальних загонів і порядок введення їх на об'єкти робіт, початок і тривалість робіт, управління силами, які ведуть роботи.

Рятування людей і матеріальних цінностей є основним завданням при стихійних лихах. Послідовність виконання цього завдання залежить від характеру лиха, його наслідків, наявності та підготовки сил цивільного захисту, періоду року, стану погоди та інших чинників.

Ліквідація наслідків будь-яких стихійних лих полягає в таких діях: оповіщенні формувань і населення про небезпеку стихійного лиха; організації управління рятувальними силами в районі лиха; встановлення ступеня і масштабів руйнувань, затоплення, зараження, пожеж та інших даних виявленні об'єктів і населених пунктів, яким загрожують наслідки стихійного лиха; визначенні складу, чисельності сил і засобів, що залучаються для рятувальних та інших робіт; організації медичної допомоги постраждалим і евакуації їх в лікувальні установи, виведення населення в безпечні місця; забезпеченні громадського порядку в районі лиха; організації матеріального, технічного та транспортного забезпечення; проведенні інших заходів, спрямованих на підготовку і забезпечення рятували них робіт; ліквідацію наслідків стихійного лиха.

Успіх дій формувань при стихійних лихах у значній мірі залежить від своєчасної організації та проведення розвідки Завдання для розвідки ставить

начальник цивільного захисту об'єкта. Він визначає мету розвідки, відомості і на який час необхідно отримати дані, на виконанні яких завдань слід зосередити основні зусилля, які для цього використовувати сили і засоби.

У районі стихійного лиха розвідка визначає:

- межі району лиха і напрямок його поширення;

- об'єкти та населені пункти, яким загрожує небезпека;

- місця знаходження людей і ступінь загрози їм;

- шляхи підходу формувань і техніки до місць роботи, стан пошкоджених будівель і споруд, наявність у них постраждалих, яким необхідна допомога в першу чергу;

- місця аварій на технологічних лініях, комунально-енергетичних мережах і розміри руйнувань на них;

- обсяги робіт, умови їх ведення;

- можливість використання техніки;

- найбільш зручні місця для розбирання завалів і звільнення постраждалих, а також для прокладки шляхів і їх евакуації, стан вододжерел і можливість використання їх для господарських, питних потреб та пожежогасіння, умови і доцільну черговість ведення робіт.

Зібрані дані про характер і обсяг майбутніх робіт розвідники доповідають начальнику у зазначені терміни.

Населення потрібно інформувати про ситуацію, що склалася та як себе поводити. Інформація повинна бути чіткою, ясною, а в завданні на проведення робіт слід конкретно вказати: хто, коли і що повинен зробити. Від цього буде залежати і успіх проведення рятувальних і невідкладних робіт в районі небезпечного стихійного лиха.

Для порятунку населення необхідно терміново покинути зону можливого поширення зсуву або селевого потоку. З оповіщенням про наближення зсуву або селевого потоку або при появі перших ознак потрібно загасити печі, вимкнути освітлення і електроприлади, перекрити газові крани, попередити сусідів і попрямувати в безпечне місце.

Якщо до селевого потоку потрапили люди, їм надають допомогу із застосуванням засобів, які є поблизу, канатів, дощок, мотузок, жердин.

Виводити постраждалих з селевого потоку треба у напрямку руху і з поступовим наближенням до його краю.

Рятувальні формування ЦЗ і населення повинні бути готові до проведення рятувальних та невідкладних робіт з метою ліквідації наслідків зсуву або селевого потоку.

Формування цивільного захисту населення повинні бути готовими в районі урагану, бурі, смерчу до проведення таких робіт:

- евакуації населення, сільськогосподарських тварин і матеріальних цінностей з небезпечних районів;

- розшуку та звільненні постраждалих з-під зруйнованих будівель і споруд, надання першої медичної допомоги та доставки їх в лікувальні установи; гасіння пожеж;

- порятунку людей і сільськогосподарських тварин, що знаходяться в палаючих або напівзруйнованих будівлях;

- ліквідації аварій на виробничих об'єктах, комунальних та енергетичних мережах.

При спільних діях особового складу формувань цивільного захисту та населення, при високій дисципліні і організованості населення можна значно зменшити наслідки стихійного лиха.

ВИСНОВКИ

В дипломному проекті розроблена програмно-апаратна система контролю якості пакування готової продукції підприємства з виробництва побутової хімії на прикладі ТОВ «Бара», що дозволяє контролювати межі маси ящиків з готовою продукцією після їх пакування.

При цьому алгоритм керуючої програми побудовано таким чином, що система враховує, що на палету з продукцією може потрапити сторонній предмет, або оператор може на палету з ящиками опиратися, але при цьому коли цей предмет буде прибрано система не втратить перелік порахованих ящиків.

Система має графічну (на додатковому індикаторі біля оператору та на дисплеї монітору комп'ютера) індикацію та звукову індикацію, що оповіщає оператора та адміністратора системи о стані якості (за параметром маси) запакованих ящиків, що встановлені на палету для відправки замовникам.

Крім того система проводить облік продукції за часом, змінами та сортаментом продукції. Наряди на продукцію автоматично завантажуються системою згідно сформованого адміністратором плану робіт на добу або тиждень в залежності від налаштувань.

Система має можливість взаємодіяти з іншими програмними продуктами по обліку та веденню статистики та складати звіти за різними показниками.

Одним з основних переваг системи є те, що розроблена система може бути зв'язана з персональним комп'ютером як за допомогою кабелю через UART протокол так і через радіоканал без необхідності проводити додаткові дроти на підприємстві. Тобто при бездротовому зв'язку достатньо з'єднати ваговий термінал з датчиком палети та з додатковим індикатором, а в персональний комп'ютер піж єднати приймач та виставити відповідні налаштування.

Науковою новизною магістерскої роботи є розробка алгоритмів виявлення сторонніх предметів на палеті з готовою продукцією без порушення роботи системи контролю та обліку та врахування сторонніх впливів на роботу системи.

ПЕРЕЛІК ЛІТЕРАТУРИ

1. Белейчева А.С., Гаффорова Е.Б. Экспертная оценка продукции - инструмент определения удовлетворенности потребителей//Методы менеджмента качества .-2006

2. Версан В.Г. Интеграция управления качеством продукции. Новые возможности М.: Изд-во стандартов,2003.

3. Войтоловский Н.В. Управление качеством продукции в условиях перехода к рынку. - СПб: СПб УЭФ,2000.

4. Воробьев В.П. оптимизация здравого смысла // Методы менеджмента качества.-2001-№11.

5. Гличев А. В. «Нововведения, маркетинг и управление качеством» ж. «Стандарты и качество»// №10, 2001г.

6. Гличев А.В. « Современные методы управления качеством»/ ж. «Стандарты и качество»//№4, 9,2006г.

7. Международные стандарты. «Управление качеством продукции». ИСО 9000 – ИСО 9004, ИСО 8402. – М.: Изд-во стандартов,2002.

8. Антонов П.А. Весы: типы и применение. - М.: Точмашприбор, 1998. - 254 с.

9. Берестов. П.С. Весоизмерительное оборудование в складской логистике. - М.: Дело, 2004. - 134 с.

10. Кемени Т. Новейшие достижения в весостроении // Измерение, контроль, автоматизация. - 2001. - №5. - С. 28-35.

ДОДАТОК А

ПРОГРАМА КЕРУЮЧОГО МІКРОКОНТРОЛЕРА ВИНОСНОГО ІНДИКАТОРУ

```
#include <SPI.h> //библиотеки для работы с приемо-передатчиком
#include <Mirf.h>
#include <nRF24L01.h>
#include <MirfHardwareSpiDriver.h>

#include <SoftwareSerial.h> //программная реализация последовательного интерфейса для связи с весами
#include "U8glib.h" //библиотека работы с графическим дисплеем
#define BEEPER 9 //вывод к которому подключаем динамик

#define MAX_BUFF 32 //размер буфера приемо-передатчика (стоит возможный максимум 32 байта)
#define NRF_CHNL 10 //номер канала работы приемо-передатчика

#define Scales_ID "ID003"

//настройка выводов подключения графического дисплея
U8GLIB_ST7920_128X64_1X u8g(6, 5, 4);  // SPI Com: SCK = en = 6, MOSI = rw = 5, CS = di = 4
//настройка выводов подключения програмной реализации последовательного порта
SoftwareSerial Scales(3, 2); // 3 - RX Arduino (TX ПК), 2 - TX Arduino (RX ПК) //Вход с весового терминала

//переменные для работы с приемо-передатчиком
uint8_t buff[32];//переменные приема
char buffCast[32];
char range[32];//переменные передачи
String recvSerial="";
//переменные для работы с весом
char S_WGHT[]="  0.00";//переменная для хранения полученного веса
char S_PREBOX[]=" 0.00";//переменная для хранения веса предпоследнего ящика
char S_LSTBOX[]=" 0.00";//переменная для хранения веса последнего ящика
char S_COUNT[]="  0";//переменная указывающая кол-во ящиков
char WGHT_INSCR[]=" ВЕС:";//надпись "вес"
char COUNT_INSCR[]=" КОР:";//надпись "количество"
bool B_ERR=false;
bool BOX_STAB=false;
bool BBOX=false;
char c;
String s_temp="";
String Wght="";
String Query="";

void setup(){
  u8g.setFont(u8g_font_10x20);//выбираем шрифт
  delay(200);                 // 200мс пауза при запуске
  Scales.begin(9600);         // инициализация связи с весами
  Serial.begin(9600);         //инициализация связи с табло

  //настройка динамика
  pinMode(BEEPER, OUTPUT);
  digitalWrite(BEEPER, LOW);
  Out_to_LCD();//выводим на дисплей

  //настройка приемо-передатчика
  Mirf.cePin = 7; //вывод соединенный с CE приемника
  Mirf.csnPin = 8; //вывод соединенный с CSN приемника
  Mirf.spi = &MirfHardwareSpi;
  Mirf.init();
  Mirf.setRADDR((byte *)"app2");
  Mirf.payload = MAX_BUFF;
  Mirf.channel = NRF_CHNL;
  Mirf.config();
```

```
  //Serial.println("Beginning ... "); //временно чтоб видеть что плата запустилась
}

void box_sound()
{
  tone(BEEPER, 3000, 500);
}

void warning_sound()
{
  tone(BEEPER, 3500, 2500);
  B_ERR=false;
  }

void bigbox_sound()
{
  tone(BEEPER, 2000, 1000);
}

//w123.45L12.34P12.34c999e0
void recognize_command(String s) {
 // Scales.println(s);

  if (s[0]=='w') {
    S_WGHT[0]=s[1];
    S_WGHT[1]=s[2];
    S_WGHT[2]=s[3];
    S_WGHT[3]=s[4];
    S_WGHT[4]=s[5];
    S_WGHT[5]=s[6];
    S_WGHT[6]='\0';
    }

  if (s[7]=='L') {
    S_LSTBOX[0]=s[8];
    S_LSTBOX[1]=s[9];
    S_LSTBOX[2]=s[10];
    S_LSTBOX[3]=s[11];
    S_LSTBOX[4]=s[12];
    S_LSTBOX[5]='\0';
    }

  if (s[13]=='P') {
    S_PREBOX[0]=s[14];
    S_PREBOX[1]=s[15];
    S_PREBOX[2]=s[16];
    S_PREBOX[3]=s[17];
    S_PREBOX[4]=s[18];
    S_PREBOX[5]='\0';
    }

  if (s[19]=='c') {
    S_COUNT[0]=s[20];
    S_COUNT[1]=s[21];
    S_COUNT[2]=s[22];
    S_COUNT[3]='\0';
    }

  if ((s[23]=='e') and (s[24]=='E')) {B_ERR=true;} else {B_ERR=false;}
  if ((s[23]=='e') and (s[24]=='0')) {BOX_STAB=true;} else {BOX_STAB=false;}
  if ((s[23]=='e') and (s[24]=='B')) {BBOX=true;} else {BBOX=false;}
}
```

```
// U8GLIB draw procedure: output the screen
void draw(void) {
  u8g.drawStr90( 112, 0, (char *)(WGHT_INSCR));  //надпись "ВЕС"
  u8g.drawStr90( 95, 10, (char *)(S_PREBOX)); //Масса предпоследнего ящика
  u8g.drawStr90( 77, 10, (char *)(S_LSTBOX)); //Масса последнего ящика
  u8g.drawStr90( 59, 0, (char *)(S_WGHT)); //Значение полной массы паллеты
  u8g.drawStr90( 41, 0, (char *)(COUNT_INSCR));//Надпись "количество"
  u8g.drawStr90( 23, 20, (char *)(S_COUNT)); //количество уложенных ящиков
  if (B_ERR) {
      u8g.drawStr90( 5, 0, "ERROR");  //сообщение об ошибке
      warning_sound();
      }
      else {u8g.drawStr90( 5, 0, " ");}
  if (BOX_STAB) {box_sound();}
  if (BBOX) {bigbox_sound();}
}

void Out_to_LCD()//выводим на дисплей
{
  u8g.firstPage();
  do {
    draw();
  } while( u8g.nextPage() );
  }

//убираем знак плюс если он стоит
String TestSign(String s){
  String s1="";
  if (s[0]=='+') {s1=s.substring(1);} else {s1=s;}
  return s1;
}

//убираем незначащие нули
String NotUsesZero(String s)
{
  String s1="";
  if (s.length()==7){
    for (int i=1; i <= 6; i++){
      if (s[i]!='0') {
        if (s[i]=='.'){s1=s1+'0';}
        s1=s1+s.substring(i);
        break;
      }
    }
    if (s1=="") {s1="0";}
    s1=s[0]+s1;
    return s1;
  }
  else {return "";}
}

String recognize_weightVN(String s) {
  String wtemp;
  bool minus=false;
  if (s.length()==21) {
    wtemp=s.substring(9,17);
    if (wtemp[0]=='-') {
      minus=true;
      wtemp=wtemp.substring(1);
      }
    wtemp.trim();
    if (minus==true){
      wtemp="-"+wtemp;
      }
    return wtemp;
```

```
    } else {return ""; }
}

//преобразовываем вес в формат для отправки на ПК
String ConvertWght(String s){
  String s1="";
  bool punkt=false;
  bool minus=false;
  String s2=s;
  if (s[0]=='-') {
    s2=s.substring(1);
    minus=true;}
  int k=s2.length()-1;
  for (int i=k; i>=0; i--){ //зеркалим без минуса если таковой есть
    s1=s1+s2[i];
    if (s2[i]=='.') {punkt=true;}
  }
  //добавляем нули до длины 6 если нет точки и до 7 если точка есть
  if (punkt==true) {k=7;} else {k=6;}
 // if (minus==true) {k--;}
  if (s1.length()<k) {
    s2=s1;
    for (int i=s1.length(); i<k; i++){
      s2=s2+"0";
      }
    s1=s2;
    }
  if (minus==true) {
    s1=s1+'-';
  }
  s1=s1+"=";//добавляем в конец символ "=" согласно пакету табло
  return s1;
}

//чтение данных с терминала BH1000 (на весах должен быть установлен адрес весов scaddr)
String ReadScalesVN(int scaddr)
{
  s_temp="";
  String s="";
  s=s+char(0x3C)+char(scaddr)+char(0x21)+char(0x3E);
  //отправляем запрос на весы
  for (int i=0; i<4; i++){
    Scales.print(s[i]);
  }
  //10 раз пытаемся дождаться ответа от терминала - если не дождемся вернем пустое значение
  for (int i=1; i<=10; i++){
waitscales:
  //ждем полный ответ
    if ( Scales.available() )
    {
      c = Scales.read();
      if (( c == '>')) {
        s_temp=s_temp+c;
        goto exitread;
        }
        else {
        s_temp=s_temp+c;
        goto waitscales;
        }
    }
  }
exitread:
  return s_temp;
}
```

```
//чтение данных с весов XK3118T1 (на терминале должен быть установлен режим Co=5)
String ReadScalesT1()
{
 s_temp="";
 char rc_ch;
 static int ScaleDeadTime=5000;//максимальное время через которое могут ответить весы
 String s="";
 s=s+char(0x41)+char(0x02)+char(0x03);
 //отправляем запрос на весы несколько раз пока не дождемся ответа
 for (int i=0; i<3; i++){//сам процес отправки запроса
  Scales.print(s[i]);
  }

 unsigned long last=millis();

 while(millis()-last<=ScaleDeadTime) //пока есть время делаем попытки считывая
 {
  while(Scales.available())
  {
   rc_ch=Scales.read();
   s_temp+=rc_ch;
   if(rc_ch==0x03){
    CleanRXBuffer();
    return s_temp;
   }
  }
 }

 return "";
}

//очищаем буффер приема
void CleanRXBuffer()
{
 delay(1);
 while(Scales.available())
 {
  Scales.read();
 }
}

//проверяем пакет вывода на дисплей - true если пакет целый
bool TestLCDData(String s)
{
 bool res=false;
 if (s.length()<30) {goto getres;}
 if (s[12]!='L') {goto getres;}
 if (s[18]!='P') {goto getres;}
 if (s[24]!='c') {goto getres;}
 if (s[28]!='e') {goto getres;}
 res=true;
getres:
 return res;
}

//распознаем полученную команду
//если команда не правильная или битая =0
//если команда получения веса = 1 IDXXXGET
//если команда вывода на дисплей =2 IDXXXw123.45L12.34P12.34c999e0
//если команда тестирования связи =3 IDXXXTEST
//если команда удаленного сброса =4 IDXXXRESET
int recognize_PCcommand(String s)
{
 int k=0;
 if (s!="") {
```

```
    if (s.length()>=8) {
      if (s.substring(0,5)==Scales_ID) {
        if (s.substring(5,8)=="GET") {k=1;}
        if ((s.substring(5,6)=="w")&&(TestLCDData(s))) {k=2;}
        if (s.length()>=9) {if (s.substring(5,9)=="TEST") {k=3;}}
        if (s.length()>=10) {if (s.substring(5,10)=="RESET") {k=4;}}
      }
    }
  }
  return k;
}

//получение строки из приемника и сохранение ее в buffCast
void GetNRFData()
{
  Mirf.getData(buff);
  sprintf(buffCast, "%s", buff);
  //int k=strlen(buffCast);
  }

void SendNRFData(String data)
{
  Mirf.setTADDR((byte *)"app1");//app1 - имя модуля соединенного с ПК
  sprintf(range, "%s", data.c_str());
  Mirf.send((uint8_t *)&range);
  while (Mirf.isSending()) {
  }
  //Serial.println(range);
}

void(* resetFunc) (void)=0;

//программный сброс ардуино
void BoxReset(){
  resetFunc();
  }

void loop(){
  int i;
  //ожидаем данные с приемника
  while(!Mirf.dataReady()) {
  }
  //если данные появились - записываем их в буфер
  GetNRFData();
  Query = String (buffCast);
  //Serial.println(Query);//временно выводим в порт полученные данные (для проверки)
  //распознаем полученную команду
  int n=recognize_PCcommand(Query);
  //Serial.println(n);//временно выводим тип распознанной команды
  //если надо читать данные с весов считываем их и отправляем обратно результат
  if (n==1) {
     //Wght=ReadScalesVN(32);
      Wght=ReadScalesT1();
      //Serial.println(Wght);//выводим в порт вес временно
      SendNRFData(Wght);
      }
  //если команда вывода на дисплей
  if (n==2) {
     //Serial.println(Query.substring(5));
     recognize_command(Query.substring(5));
     //Serial.println("LCD OK");
     Out_to_LCD();//выводим на дисплей
              SendNRFData("LCD OK");
     }
  //если команда тестирования связи
```

```
  if (n==3) {
    SendNRFData("TEST OK");
    }
  //если команда удаленного сброса
  if (n==4) {
    BoxReset();
    }
  Query="";
}
```

ДОДАТОК Б

ПРОГРАМА КЕРУЮЧОГО МІКРОКОНТРОЛЕРА ПРИЙМАЛЬНОГО МОДУЛЯ

```
#include <SPI.h>
#include <Mirf.h>
#include <nRF24L01.h>
#include <MirfHardwareSpiDriver.h>

#define MAX_BUFF 32 //размер буфера приемо-передатчика (стоит возможный максимум 32 байта)
#define NRF_CHNL 10 //номер канала работы приемо-передатчика

//переменные для работы с приемо-передатчиком
uint8_t buff[32];//переменные приема
char buffCast[32];
String Query="";
char range[32];//переменные передачи
String recvSerial="";
char c;//переменная для получения данных с порта

void setup() {
  Serial.begin(9600);

  Mirf.cePin = 7;
  Mirf.csnPin = 8;
  Mirf.spi = &MirfHardwareSpi;
  Mirf.init();
  Mirf.setRADDR((byte *)"app1");
  Mirf.payload = MAX_BUFF;
  Mirf.channel = NRF_CHNL;
  Mirf.config();

  Serial.println("Beginning ... ");
}

//String Type1 = "ID001w123.45L12.34P12.34c999e0";
//String Type2 = "ID001GET";

//получение строки из приемника и сохранение ее в buffCast
void GetNRFData()
{
  Mirf.getData(buff);
  sprintf(buffCast, "%s", buff);
  //int k=strlen(buffCast);
  }

void SendNRFData(String data)
{
  Mirf.setTADDR((byte *)"app2");
  sprintf(range, "%s", data.c_str());
  Mirf.send((uint8_t *)&range);
  while (Mirf.isSending()) {
  }
  Serial.println(range);
}

void loop() {
  Query="";
  //відправка даних
  while (Serial.available())
  {
    c = Serial.read();
    if (( c == '\n' ) or ( c == '\r' )) {
```

```
      SendNRFData(recvSerial);
      recvSerial = "";
    } else {recvSerial+=c;}
  }
  //ожидаем данные с приемника
  while(Mirf.dataReady()) {
    //если данные появились - записываем их в буфер
    GetNRFData();
    Query = String (buffCast);
    Serial.println(Query);//выводим в порт полученные данные (для проверки)
  }
}
```

ДОДАТОК В
ОСНОВНИЙ БЛОК ПРОГРАМИ ДЛЯ ПК

```
Unit Main;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, comobj, StdCtrls, DB, ADODB, Grids, DBGrids, DBCtrls, Menus,
  ExtCtrls, Buttons, jpeg, Registry, ComCtrls, IdMultipartFormData, wininet,
  IdBaseComponent, IdComponent, IdTCPConnection, IdTCPClient, IdHTTP, DateUtils,
  FileCtrl, Printers, OleServer, WordXP, Sockets;

type Weighting = Record
  WDBID:integer; //индекс записи в БД
  WDate:string;  //дата взвешивания
  WTime:string;  //время взвешивания
  WCargoName:string; //название продукции
  WCargoID:string;   //ИД продукции в БД
  WThisMassa:string; //значение текущей массы в символьном виде
  WThisMassaInt:integer;  //значение текущей массы в числовом виде
  WFullMassa:string;  //полное значение массы загруженной паллеты для данного взвешивания в символьном
  WFullMassaInt:integer; //полное значение массы загруженной паллеты для данного взвешивания в числовом
  WUserID:string;    //ИД пользователя который в данный момент авторизован
  WWorkShift:string; //номер смены
  WPallete:string;  //номер паллеты - соответствует ИД в БД
  WType:string;     //тип взвешиваемой массы (ящик, перестил, уголок, пустая паллеты, плохой ящик и т.д
  WScaleID:string;  //ИД весов в БД
  WBoxCount:string; //количество ящиков
  WBadBoxCount:string; //количество плохих ящиков
  WTaskDate:string; //дата наряда
  end;

var
  Form_Main: TForm_Main;
  MyDir:string; //Директория с программой
  ScalesCount:integer; //количество подключенных весов к программе
  ScCOM:array of byte;//массив хранящий номера портов для подключения весов
  DispCOM:array of byte;//массив хранящий номера портов выносных табло
  ScBR:array of longint;//массив хранящий скорость обмена для подключения весов
  ScAddr:array of byte; //массив хранящий адреса весов
  ScType:array of byte; //массив хранящий тип терминала 1 - ВН, 2 - Т1
  ScName:array of string; //массив хранящий имена рабочих мест (весовых терминалов)
  ScInpFolder:array of string; //массив хранящий имена папок с входными файлами (нарядами)
  ScOutFolder:array of string; //массив хранящий имена папок с выодными файлами
  ScInFName:array of string; //массив имен файлов с нарядами
  ScID:array of string;//массив ИД весовых терминалов
  ScModeType:array of integer;//тип режима 0 - наборной, 1 штучный
  LastWeight:array[1..9] of real;//предыдущий вес
  ThisWeight:array[1..9] of real;//текущий вес
  BoxMin:array[1..9] of integer;//минимальный вес ящика
  BoxMax:array[1..9] of integer;//максимальный вес ящика
  PrevBox,LastBox:array[1..9] of string;//массы последнего и предпоследнего ящика
  PalletteStarted:array[1..9] of boolean;//признак того что паллета запущена
  CargoType:array[1..9] of integer;//тип текущего груза (ящик, пустая паллета, перестил, уголки)
  PltWghts1:array of Weighting;//массив данных текущей паллеты участка №1
  PltWghts2:array of Weighting;//массив данных текущей паллеты участка №2
  PltWghts3:array of Weighting;//массив данных текущей паллеты участка №3
  PltWghts4:array of Weighting;//массив данных текущей паллеты участка №4
  PltWghts5:array of Weighting;//массив данных текущей паллеты участка №5
  PltWghts6:array of Weighting;//массив данных текущей паллеты участка №6
```

```
  PltWghts7:array of Weighting;//массив данных текущей паллеты участка №7
  PltWghts8:array of Weighting;//массив данных текущей паллеты участка №8
  PltWghts9:array of Weighting;//массив данных текущей паллеты участка №9
  WasZero:array[1..9] of boolean;//последний стабильный вес был 0
  Plt_ID:integer;//ИД паллеты
  Last5W:array[1..5] of integer;//последние 4 взвешивания для выведения правильного четкого веса
  FileWriteStarted:boolean;//признак того что старт паллеты был уже записан
  RedRows:array of integer; //массив строк красного цвета для ящиков <min
  YellowRows:array of integer; //массив строк желтого цвета >max
  Paused:boolean;//признак паузы
  procedure LoadTask(taskfile:string);
  function Create1stFileRec:string;
  function CreateNot1FileRec(n:integer):string;
  procedure ClearOutFile(n:integer);
  procedure TabsVisible(n:integer);//устанавливаем видимость n вкладок терминалов
  procedure ComTabsVisible(n:integer);//устанавливаем видимость n вкладок терминалов в настройках соединений
  procedure ReadCOMSettingsDB;
  procedure SetArraysLen(n:integer);
  procedure SendTCPSetings;

implementation

uses workDB, Scales, ScDisplay, CargoCatalog, COMPort, StoreDest, Users, Limits,
  Authorisation, ExitF, ScalesSetup, Emulation, About, Developer, Warning,
  CreateFile, CreateForPlt, DBnFiles, EntLog;

{$R *.dfm}

procedure ExitProgram;
begin
  Form_Exit.Show;
end;

procedure TForm_Main.N_ExitClick(Sender: TObject);
begin
  ExitProgram;
end;

//Настройка внешнего вида основной таблицы
procedure MainTableSetup;
var
  MyStringGrid:TStringGrid;
  i:integer;
  s_temp:string;
begin
  for i:=1 to 9 do begin
    s_temp:='StringGrid_SC0'+IntToStr(i);
    MyStringGrid:=TStringGrid(Form_Main.FindComponent(s_temp));
    MyStringGrid.Cells[0,0]:='Дата';
    MyStringGrid.Cells[1,0]:='Время';
    MyStringGrid.Cells[2,0]:='Наименование товара';
    MyStringGrid.Cells[3,0]:='Вес ящика,кг';
    MyStringGrid.Cells[4,0]:='Вес полный,кг';
    MyStringGrid.Cells[5,0]:='Вес пустой паллеты,кг';
    MyStringGrid.Cells[6,0]:='Вес перестила (уголка),кг';
    MyStringGrid.Cells[7,0]:='Дата наряда';
    MyStringGrid.Cells[8,0]:='Номер смены';
    MyStringGrid.Cells[9,0]:='Кол-во ящиков';
    MyStringGrid.ColWidths[0]:=60;
    MyStringGrid.ColWidths[1]:=50;
    MyStringGrid.ColWidths[2]:=250;
    MyStringGrid.ColWidths[3]:=80;
    MyStringGrid.ColWidths[4]:=80;
    MyStringGrid.ColWidths[5]:=120;
    MyStringGrid.ColWidths[6]:=130;
```

```
    MyStringGrid.ColWidths[7]:=60;
    MyStringGrid.ColWidths[8]:=80;
    MyStringGrid.ColWidths[9]:=80;
    end;
end;

//устанавливаем видимость n вкладок терминалов
procedure TabsVisible(n:integer);
begin
  Form_Main.TabSheet_SC02.TabVisible:=false;
  Form_Main.TabSheet_SC03.TabVisible:=false;
  Form_Main.TabSheet_SC04.TabVisible:=false;
  Form_Main.TabSheet_SC05.TabVisible:=false;
  Form_Main.TabSheet_SC06.TabVisible:=false;
  Form_Main.TabSheet_SC07.TabVisible:=false;
  Form_Main.TabSheet_SC08.TabVisible:=false;
  Form_Main.TabSheet_SC09.TabVisible:=false;
  if n>=2 then Form_Main.TabSheet_SC02.TabVisible:=true;
  if n>=3 then Form_Main.TabSheet_SC03.TabVisible:=true;
  if n>=4 then Form_Main.TabSheet_SC04.TabVisible:=true;
  if n>=5 then Form_Main.TabSheet_SC05.TabVisible:=true;
  if n>=6 then Form_Main.TabSheet_SC06.TabVisible:=true;
  if n>=7 then Form_Main.TabSheet_SC07.TabVisible:=true;
  if n>=8 then Form_Main.TabSheet_SC08.TabVisible:=true;
  if n=9 then Form_Main.TabSheet_SC09.TabVisible:=true;
end;

//устанавливаем видимость n вкладок терминалов в настройках сейдинений
procedure ComTabsVisible(n:integer);
begin
  Form_COMPort.TabSheet02.TabVisible:=false;
  Form_COMPort.TabSheet03.TabVisible:=false;
  Form_COMPort.TabSheet04.TabVisible:=false;
  Form_COMPort.TabSheet05.TabVisible:=false;
  Form_COMPort.TabSheet06.TabVisible:=false;
  Form_COMPort.TabSheet07.TabVisible:=false;
  Form_COMPort.TabSheet08.TabVisible:=false;
  Form_COMPort.TabSheet09.TabVisible:=false;
  if n>=2 then Form_COMPort.TabSheet02.TabVisible:=true;
  if n>=3 then Form_COMPort.TabSheet03.TabVisible:=true;
  if n>=4 then Form_COMPort.TabSheet04.TabVisible:=true;
  if n>=5 then Form_COMPort.TabSheet05.TabVisible:=true;
  if n>=6 then Form_COMPort.TabSheet06.TabVisible:=true;
  if n>=7 then Form_COMPort.TabSheet07.TabVisible:=true;
  if n>=8 then Form_COMPort.TabSheet08.TabVisible:=true;
  if n=9 then Form_COMPort.TabSheet09.TabVisible:=true;
end;

//чтение настроек расположение БД и подключение к ней
procedure ConnectToDB;
var
  s_temp:string;
  fname:Textfile;
  d_temp:string;
begin
  //Читаем настройки расположения БД, файлов нарядов и выходных файлов
  assignfile(fname,MyDir+'\settings\Folders.dat');
  reset(fname);
  readln(fname,s_temp);//Местоположение БД
  if copy(s_temp,1,9)<>'DBFolder=' then workDB.dbName:=MyDir+'\dll\bara.mdb';
  if copy(s_temp,10,5)='MyDir' then
    begin d_temp:=copy(s_temp,15,length(s_temp)-14);
    workDB.dbName:=MyDir+d_temp+'\bara.mdb';
    end else workDB.dbName:=copy(s_temp,10,length(s_temp)-9)+'\bara.mdb';
  closefile(fname);
```

```
  //проверяем наличие БД в указанной в настройках папке - если нет создаем новую БД
  if not fileexists(DBName) then begin
    workDB.CreateNewDB;
    workDB.link_to_db;
    workDB.CreateDBTables;
    end else
    //если БД есть просто соединяемся
    workDB.link_to_db;
end;

//очищаем выходной файл для весов с номером n
//и в настройках ставим текущую дату как дату начала записи файла
procedure ClearOutFile(n:integer);
var
  fout:Textfile;
  s_temp,s_temp2:string;
begin
  assignfile(fout,ScOutFolder[n-1]+'\вес.txt');
  rewrite(fout);
  closefile(fout);

  assignfile(fout,MyDir+'\settings\Flife.dat');
  reset(fout);
  readln(fout,s_temp2);
  closefile(fout);

  rewrite(fout);
  writeln(fout,s_temp2);
  s_temp:=DateToStr(Now);
  s_temp:='StartDate='+s_temp;
  writeln(fout,s_temp);
  closefile(fout);
end;

//проверяем не прошло ли указанное кол-во дней хранения и обновляем файл если прошли
//!!!добавить n - номер весов для которых проверяется время жизни файла
procedure TestFileLife;
var
  fout:textfile;
  s_temp:string;
  i,c:integer;
  dstart,dfinish:string;
  dcount:string;
  d_temp1,d_temp2:TDateTime;
begin
  assignfile(fout,MyDir+'\settings\Flife.dat');
  reset(fout);
  readln(fout,s_temp);//кол-во дней после которых очищать
  dcount:=copy(s_temp,6,length(s_temp)-5);
  val(dcount,i,c);
  readln(fout,s_temp);//дата начала записи
  dstart:=copy(s_temp,11,length(s_temp)-10);
  closefile(fout);

  d_temp1:=StrToDate(dstart);
  d_temp1:=IncDay(d_temp1,i); //увеличиваем стартовую дату на кол-во дней хранения
  d_temp2:=Now;   //и сравниваем с текущей датой
  if CompareDate(d_temp1,d_temp2)<0 then ClearOutFile(1); //если текущая дата больше граничной
end;

//читаем места хранения для терминалов
procedure ReadFolders;
var
  s_temp3:string;
  i,n:integer;
```

```
  TabSheet_SC:TTabSheet;
  TShName:string;
begin
  s_temp3:='SELECT * FROM Scales WHERE Activity=1 ORDER BY Id_scale;';
  Form_Main.ADOQuery1.Active:=false;
  Form_Main.ADOQuery1.SQL.Text:=s_temp3;
  //Form_Main.ADOQuery1.ExecSQL;
  Form_Main.ADOQuery1.Active:=true;
  n:=Form_Main.ADOQuery1.RecordCount;
  if n=0 then begin showmessage('В БД нет весовых терминалов'); Exit; end;
  setlength(ScName,n);
  setlength(ScInpFolder,n);
  setlength(ScOutFolder,n);
  setlength(ScInFName,n);
  for i:=1 to n do begin
    s_temp3:=Form_Main.ADOQuery1.FieldByName('Scale_Name').AsString;
    Main.ScName[i-1]:=s_temp3;
    TShName:='TabSheet_SC0'+IntToStr(i);
    TabSheet_SC:=TTabSheet(Form_Main.FindComponent(TShName));
    TabSheet_SC.Caption:=s_temp3;
    s_temp3:=Form_Main.ADOQuery1.FieldByName('InputFolder').AsString;
    Main.ScInpFolder[i-1]:=s_temp3;
    s_temp3:=Form_Main.ADOQuery1.FieldByName('OutputFolder').AsString;
    Main.ScOutFolder[i-1]:=s_temp3;
    s_temp3:=Form_Main.ADOQuery1.FieldByName('InputFName').AsString;
    Main.ScInFName[i-1]:=s_temp3;
    Form_Main.ADOQuery1.Next;
    end;
end;

procedure TCPSetup;//Настроиваем TCP клиент
begin
  Form_Main.TcpClient1.RemoteHost:='127.0.0.1';//'91.218.96.34'; //'127.0.0.1';
  Form_Main.TcpClient1.RemotePort:='5533';
end;

//читаем кол-во подключенных к программе весов
procedure ReadScalesCount;
var
  fname:string;
  fsets:textfile;
  s_temp:string;
  n,c:integer;
begin
  //ScalesCount; //количество подключенных весов к программе
  fname:=MyDir+'/settings/ScCnt.dat';
  assignfile(fsets,fname);
  reset(fsets);
  readln(fsets,s_temp);
  s_temp:=copy(s_temp,16,1); //ScalesQuantity=n
  val(s_temp,n,c);
  if c<>0 then Main.ScalesCount:=1 else Main.ScalesCount:=n;
  closefile(fsets);
end;

//устанавливаем размеры массивов для параметров весов
procedure SetArraysLen(n:integer);
begin
  setlength(Main.ScCOM,n);//массив хранящий номера портов для подключения весов
  setlength(Main.ScBR,n);//массив хранящий скорость обмена для подключения весов
  setlength(Main.ScAddr,n); //массив хранящий адреса весов
  setlength(Main.DispCOM,n); //массив хранящий номера портов выносных табло
  setlength(Main.ScType,n); //массив хранящий тип терминала 1 - ВН, 2 - Т1
  setlength(Main.ScID,n);  //массив ИД весовых терминалов
  setlength(Main.ScModeType,n); //тип режима 0 - наборной, 1 штучный
```

```
  setlength(Main.ScName,n); //массив хранящий имена рабочих мест (весовых терминалов)
  setlength(Main.ScInpFolder,n); //массив хранящий имена папок с входными файлами (нарядами)
  setlength(Main.ScOutFolder,n); //массив хранящий имена папок с выодными файлами
  setlength(Main.ScInFName,n); //массив имен файлов с нарядами
end;

//=============================================================================
//читаем настройки в массив и ставим активным порт из БД
//если его нет в списке меняем цвет формы
procedure ReadCOMSettingsDB;
var
  i:integer;
  n:byte;
  s_temp,s_temp2,s_temp3:string;
begin
  s_temp3:='SELECT * FROM Scales WHERE Activity=1 ORDER BY Id_scale;';
  Form_Main.ADOQuery1.Active:=false;
  Form_Main.ADOQuery1.SQL.Text:=s_temp3;
  //Form_Main.ADOQuery1.ExecSQL;
  Form_Main.ADOQuery1.Active:=true;
  n:=Form_Main.ADOQuery1.RecordCount;
  if n=0 then begin showmessage('В БД нет весовых терминалов'); Exit; end;
  Main.ScalesCount:=n;
  SetArraysLen(n); //устанавливаем размеры массивов для параметров весов
  for i:=1 to n do begin
    Main.ScCOM[i-1]:=Form_Main.ADOQuery1.FieldByName('COMNumb').AsInteger;
    Main.ScBR[i-1]:=Form_Main.ADOQuery1.FieldByName('BaudRate').AsInteger;
    Main.ScAddr[i-1]:=Form_Main.ADOQuery1.FieldByName('ScaleAddr').AsInteger;
    Main.DispCOM[i-1]:=Form_Main.ADOQuery1.FieldByName('COMDisplay').AsInteger;
    Main.ScType[i-1]:=Form_Main.ADOQuery1.FieldByName('ScaleType').AsInteger;
    Main.ScID[i-1]:=Form_Main.ADOQuery1.FieldByName('ScaleID').AsString;
    Main.ScModeType[i-1]:=Form_Main.ADOQuery1.FieldByName('ModeType').AsInteger;
    Main.ScName[i-1]:=Form_Main.ADOQuery1.FieldByName('Scale_Name').AsString;
    Main.ScOutFolder[i-1]:=Form_Main.ADOQuery1.FieldByName('OutputFolder').AsString;
    Main.ScInFName[i-1]:=Form_Main.ADOQuery1.FieldByName('InputFolder').AsString;
    Main.ScInFName[i-1]:=Form_Main.ADOQuery1.FieldByName('InputFName').AsString;
    Form_Main.ADOQuery1.Next;
    end;
end;

//отправляем TCP серверу настройки
procedure SendTCPSetings;
var
  i:integer;
  s_temp:string;
begin
  //сбрасываем настройки
  Form_Main.TcpClient1.Connect();
  Form_Main.TcpClient1.Sendln('ResetSettings');
  Form_Main.TcpClient1.Close();
  //устанавливаем кол-во весов
  Form_Main.TcpClient1.Connect();
  //!!!Form_Main.TcpClient1.Sendln('SETScCount=1');
  Form_Main.TcpClient1.Sendln('SETScCount='+IntToStr(Main.ScalesCount));
  Form_Main.TcpClient1.Close();
  //прописываем ИД весов
  s_temp:='';
  for i:=1 to Main.ScalesCount do
    s_temp:=s_temp+';'+Main.ScID[i-1];
  s_temp:=copy(s_temp,2,length(s_temp)-1);
  Form_Main.TcpClient1.Connect();
//!!! Form_Main.TcpClient1.Sendln('SETID=ID005');
//!!!  Form_Main.TcpClient1.Sendln('SETID=ID002;ID003');
  Form_Main.TcpClient1.Sendln('SETID='+s_temp);
  Form_Main.TcpClient1.Close();
```

```
  //устанавливаем порт к которому подключен передатчик
  Form_Main.TcpClient1.Connect();
  Form_Main.TcpClient1.Sendln('SETCOM=COM'+IntToStr(Main.ScCOM[0]));
  Form_Main.TcpClient1.Close();
end;

//запускаем TCP сервер
function StartTCPServer:boolean;
var
  srvName:string;
begin
  Form_Main.TcpClient1.Connect();
  if not Form_Main.TcpClient1.Connected then
  begin
    srvName:=MyDir+'\proxyBuild\TCPServer.exe';
    WinExec(Pchar(srvName), SW_SHOWNORMAL);
    Result:=false;
    end else Result:=true;
end;

procedure TForm_Main.FormCreate(Sender: TObject);
var
  s_temp:string;
  i:integer;
  TCPWasStarted:boolean;
begin
  MyDir:=ExtractFileDir(Application.ExeName); //Name of current folder
  TCPWasStarted:=StartTCPServer;//запускаем TCP сервер
  ConnectToDB; //чтение настроек расположение БД и подключение к ней
  MainTableSetup; //Настройка внешнего вида основных таблиц
  ReadScalesCount; //кол-во подключенных весов Main.ScalesCount из файла
  TabsVisible(1);//!!!!пока не сделаем несколько сразу TabsVisible(Main.ScalesCount);
  for i:=1 to Main.ScalesCount do begin
    LastWeight[i]:=0;
    PalletteStarted[i]:=false;//Паллета не запущена
    WasZero[i]:=true;
    end;
  setlength(PltWghts1,0);//изначально очищаем массы взвешиваний для текущей паллеты
  setlength(PltWghts2,0);//изначально очищаем массы взвешиваний для текущей паллеты
  setlength(PltWghts3,0);//изначально очищаем массы взвешиваний для текущей паллеты
  setlength(PltWghts4,0);//изначально очищаем массы взвешиваний для текущей паллеты
  setlength(PltWghts5,0);//изначально очищаем массы взвешиваний для текущей паллеты
  setlength(PltWghts6,0);//изначально очищаем массы взвешиваний для текущей паллеты
  setlength(PltWghts7,0);//изначально очищаем массы взвешиваний для текущей паллеты
  setlength(PltWghts8,0);//изначально очищаем массы взвешиваний для текущей паллеты
  setlength(PltWghts9,0);//изначально очищаем массы взвешиваний для текущей паллеты
  ReadFolders; //читаем места хранения для терминалов
  TestFileLife;//проверяем не прошло ли указанное кол-во дней хранения и обновляем файл если прошли
  TCPSetup;//Настроиваем TCP клиент
  ReadCOMSettingsDB;
  if not TCPWasStarted then SendTCPSetings;//отправляем TCP серверу настройки
  Form_Main.Timer1.Enabled:=true;
end;

procedure TForm_Main.N_Menu_ExitClick(Sender: TObject);
begin
  ExitProgram;
end;

procedure TForm_Main.N_CargoCatalogClick(Sender: TObject);
begin
  Form_CargoCatalog.Show;
end;

procedure TForm_Main.N_COMPortClick(Sender: TObject);
```

```
begin
  Form_COMPort.Show;
end;

//save record about weighting to DataBase
procedure SaveWToDB;
var
  s_temp,s_temp3:string;
  n,i:integer;
  twid, twdate, twtime, twdatetime,  twmassa, twcargoid, twuserid, twworkshift:string;
  twplt, twctype, twscale, twtaskdate:string;
begin
  s_temp3:='SELECT Id_weight FROM Weightings;';
  Form_Main.ADOQuery1.Active:=false;
  Form_Main.ADOQuery1.SQL.Text:=s_temp3;
  //Form_Main.ADOQuery1.ExecSQL;
  Form_Main.ADOQuery1.Active:=true;
  n:=Form_Main.ADOQuery1.RecordCount;
  twid:=IntToStr(n+1);
  twdate:=DateToStr(Now);
  twdate:=copy(twdate,7,4)+'-'+copy(twdate,4,2)+'-'+copy(twdate,1,2);
  twtime:=TimeToStr(Now); //формат записи "2018-07-10 20:55:00"
  twdatetime:='"'+twdate+' '+twtime+'"';
  twctype:=IntToStr(CargoType[1]);
  twmassa:=IntToStr(round(ThisWeight[1]*1000));
  twcargoid:=Form_Main.Label_CargoID01.Caption;
  twuserid:=IntToStr(Authorisation.user_id);
  if Form_Main.Label_WorkShift01.Caption='Первая' then twworkshift:='1';
  if Form_Main.Label_WorkShift01.Caption='Вторая' then twworkshift:='2';
  if Form_Main.Label_WorkShift01.Caption='Третья' then twworkshift:='3';
  twplt:=IntToStr(Plt_ID);
  twscale:='1';//!!! когда будет несколько надо определять какие
  twtaskdate:=Form_Main.Label_TaskDate01.Caption;
  twtaskdate:='"20'+copy(twtaskdate,7,2)+'-'+copy(twtaskdate,4,2)+'-'+copy(twtaskdate,1,2)+'"';

  s_temp3:='INSERT INTO Weightings VALUES ('+twid+', '+twdatetime+', '+twmassa+', ';
  s_temp3:=s_temp3+twcargoid+', '+twuserid+', '+twworkshift+', '+twplt+', ';
  s_temp3:=s_temp3+twctype+', 1, '+twscale+', '+twtaskdate+', 1, NULL);';
  Form_Main.ADOQuery1.Active:=false;
  Form_Main.ADOQuery1.SQL.Text:=s_temp3;
  Form_Main.ADOQuery1.ExecSQL;
  //Form_Main.ADOQuery1.Active:=true;
end;

//если не найден выходной файл создаем его в папке \reports и заполняем из БД
//входные данные n - номер весов
procedure CreateNFillNewOutFile(n:integer);
var
  s_temp,s_temp3:string;
  i:integer;
  fout:Textfile;
begin
  assignfile(fout,MyDir+'\reports\вес.txt');
  rewrite(fout);
  //читаем данные начиная с даты начала записи в файл для данных весов

  //записываем их в файл

  closefile(fout);
end;

//формируем первую запись для паллеты в файле
function Create1stFileRec:string;
var
  s_temp:string;
```

```
  i,c:integer;
begin
    //дата наряда
    s_temp:=copy(PltWghts1[0].WTaskDate,1,6)+copy(PltWghts1[0].WTaskDate,9,2)+';';
    //смена
    s_temp:=s_temp+Form_Main.Label_WorkShift01.Caption+';';
    //код товара
    val(PltWghts1[0].WCargoID,i,c);
    s_temp:=s_temp+IntToStr(workDB.FindCargoCode(i))+';';
    //название товара
    s_temp:=s_temp+PltWghts1[0].WCargoName+';';
    //масса пустой паллеты т.е. текущая масса в г
    s_temp:=s_temp+PltWghts1[0].WThisMassa+';';
    //текущая дата
    s_temp:=s_temp+copy(PltWghts1[0].WDate,1,6)+copy(PltWghts1[0].WDate,9,2)+';';
    //текущее время
    s_temp:=s_temp+PltWghts1[0].WTime+';';
    Result:=s_temp;
end;

//формируем отличную от первой записи запись в файле
//n - номер записи n>1
function CreateNot1FileRec(n:integer):string;
var
  s_temp:string;
begin
  if n<1 then begin
    Result:='';
    Exit;
    end;
  //текущая дата
  s_temp:=copy(PltWghts1[n-1].WDate,1,6)+copy(PltWghts1[n-1].WDate,9,2)+';';
  //время
  s_temp:=s_temp+PltWghts1[n-1].WTime+';';
  //последняя масса
  s_temp:=s_temp+PltWghts1[n-1].WThisMassa+';';
  //полная масса
  s_temp:=s_temp+PltWghts1[n-1].WFullMassa+';';
  Result:=s_temp;
end;

//сохраняем массив PltWghts в выходной файл удаляем предыдущие записи
//используем когда происходили изменения в минус в файле
//n - номер весов
procedure SavePltToFile(n:integer);
label m1;
var
  fout:textfile;
  s_temp:string;
  i,j,k,c,ind:integer;
  F:TStringList;
  FName:string;
begin
  FName:=ScOutFolder[n-1]+'\вес.txt';
  if not fileexists(FName) then Exit;
  F:=TStringList.Create;
  F.LoadFromFile(FName); //загружаем файл в память
  k:=F.Count;
  if k=0 then goto m1;
  //находим строку в которой начиналась последняя паллета
  for i:=k-1 downto 0 do begin
    s_temp:=copy(F[i],10,1);
    val(s_temp,j,c);
    if c<>0 then begin
      ind:=i; Break;
```

```
      end;
    end;
  //удаляем строки для последней паллеты
  for i:=k-1 downto ind do
    F.Delete(i);
  F.SaveToFile(FName);
  k:=F.Count;//Новое количество строк
m1:
  F.Free;
  //создаем новые записи для последней паллеты
  j:=length(PltWghts1);
  if j<1 then Exit;
  assignfile(fout,FName);
  append(fout);
  //формируем первую запись
  s_temp:=Create1stFileRec;//формируем первую запись для паллеты в файле
  writeln(fout,s_temp);
  //формируем не первые записи в цикле
  if j>1 then for i:=2 to j do begin
    s_temp:=CreateNot1FileRec(i); //формируем отличную от первой записи запись в файле
    writeln(fout,s_temp);
    end;
  closefile(fout);
end;

//Save record about weighting to output file
procedure SaveWToFile;
var
  s_temp:string;
  fname:textfile;
  i,n,c:integer;
begin
  n:=length(PltWghts1);
  Form_Developer.Label_OutputFile.Caption:=ScOutFolder[0]+'\вес.txt';
  //if not fileexists(ScOutFolder[0]+'\вес.txt') then begin
    //CreateNFillNewOutFile(1);
    //Exit;
    //end;
  assignfile(fname,ScOutFolder[0]+'\вес.txt');
  if not fileexists(ScOutFolder[0]+'\вес.txt') then rewrite(fname) else append(fname);
  if n=0 then Exit;
  if FileWriteStarted=false then begin //Если это первая запись паллеты
    s_temp:=Create1stFileRec;//формируем первую запись для паллеты в файле
    FileWriteStarted:=true;
    end
    else begin  //если это не первая запись
      if (SomeBoxes) and (PltWghts1[n-1].WType='6') and (n>=2) then begin
        s_temp:=CreateNot1FileRec(n-1); //формируем отличную от первой записи запись в файле
        writeln(fname,s_temp);
        end;
      s_temp:=CreateNot1FileRec(n); //формируем отличную от первой записи запись в файле
    end;
  writeln(fname,s_temp);
  closefile(fname);
end;

//закрашиваем в красный цвет строку с номером r
procedure ColorRowRed(r:integer);
var
   i,n:integer;
begin
  n:=length(RedRows);
  for i:=0 to n-1 do
    if RedRows[i]=r then Exit;
  setlength(RedRows,n+1);
```

```
  RedRows[n]:=r;
end;

//закрашиваем в желтый цвет строку с номером r
procedure ColorRowYellow(r:integer);
var
   i,n:integer;
begin
  n:=length(YellowRows);
  for i:=0 to n-1 do
    if YellowRows[i]=r then Exit;
  setlength(YellowRows,n+1);
  YellowRows[n]:=r;
end;

//процедура оповещения пользователя о направильном весе
procedure WarningShow;
begin
  Form_Warning.Timer1.Enabled:=true;
  Form_Warning.Show;
end;

//получаем тип звукового сигнала wnumb - номер весов от 0
function TakeBType(wnumb:byte;n:integer):byte;
var
  tmp_btype:byte;
begin
  tmp_btype:=0;
  if wnumb>8 then begin Result:=0; Exit; end;
  case wnumb of
    0: begin
        if (PltWghts1[n-1].WType='0') or (PltWghts1[n-1].WType='7') then tmp_btype:=255;
        if PltWghts1[n-1].WType='1' then tmp_btype:=1;
        if PltWghts1[n-1].WType='5' then tmp_btype:=2;
        end;
    1: begin
        if (PltWghts2[n-1].WType='0') or (PltWghts2[n-1].WType='7') then tmp_btype:=255;
        if PltWghts2[n-1].WType='1' then tmp_btype:=1;
        if PltWghts2[n-1].WType='5' then tmp_btype:=2;
        end;
    2: begin
        if (PltWghts3[n-1].WType='0') or (PltWghts3[n-1].WType='7') then tmp_btype:=255;
        if PltWghts3[n-1].WType='1' then tmp_btype:=1;
        if PltWghts3[n-1].WType='5' then tmp_btype:=2;
        end;
    3: begin
        if (PltWghts4[n-1].WType='0') or (PltWghts4[n-1].WType='7') then tmp_btype:=255;
        if PltWghts4[n-1].WType='1' then tmp_btype:=1;
        if PltWghts4[n-1].WType='5' then tmp_btype:=2;
        end;
    4: begin
        if (PltWghts5[n-1].WType='0') or (PltWghts5[n-1].WType='7') then tmp_btype:=255;
        if PltWghts5[n-1].WType='1' then tmp_btype:=1;
        if PltWghts5[n-1].WType='5' then tmp_btype:=2;
        end;
    5: begin
        if (PltWghts6[n-1].WType='0') or (PltWghts6[n-1].WType='7') then tmp_btype:=255;
        if PltWghts6[n-1].WType='1' then tmp_btype:=1;
        if PltWghts6[n-1].WType='5' then tmp_btype:=2;
        end;
    6: begin
        if (PltWghts7[n-1].WType='0') or (PltWghts7[n-1].WType='7') then tmp_btype:=255;
        if PltWghts7[n-1].WType='1' then tmp_btype:=1;
        if PltWghts7[n-1].WType='5' then tmp_btype:=2;
        end;
```

```
    7: begin
        if (PltWghts8[n-1].WType='0') or (PltWghts8[n-1].WType='7') then tmp_btype:=255;
        if PltWghts8[n-1].WType='1' then tmp_btype:=1;
        if PltWghts8[n-1].WType='5' then tmp_btype:=2;
        end;
    8: begin
        if (PltWghts9[n-1].WType='0') or (PltWghts9[n-1].WType='7') then tmp_btype:=255;
        if PltWghts9[n-1].WType='1' then tmp_btype:=1;
        if PltWghts9[n-1].WType='5' then tmp_btype:=2;
        end;
    end;
  Result:=tmp_btype;
end;

//отправка данных на выносное табло wnumb - номер весов от 0, wt - масса которую выводим
procedure SendWghtToDisplay(wnumb:byte;wt:real);
label m1,m2;
var
  s_temp,s_temp2:string;
  s_temp3,s_temp4:string;
  er,bx:boolean;
  btype:byte;
  box:real;
  i,c,n:integer;
  LBLBoxCnt:String;
  Label_BoxCount:TLabel;
begin
  n:=0;
  case wnumb of
    0: n:=length(PltWghts1);
    1: n:=length(PltWghts2);
    2: n:=length(PltWghts3);
    3: n:=length(PltWghts4);
    4: n:=length(PltWghts5);
    5: n:=length(PltWghts6);
    6: n:=length(PltWghts7);
    7: n:=length(PltWghts8);
    8: n:=length(PltWghts9);
    end;
  if n=0 then Exit;
  btype:=TakeBType(wnumb,n);  //тип звукового сигнала
  s_temp:=FormatFloat('#0.00',wt); //текущий вес
  LBLBoxCnt:='Label_BoxCount0'+IntToStr(wnumb+1);
  Label_BoxCount:=TLabel(Form_Main.FindComponent(LBLBoxCnt));
  s_temp2:=Label_BoxCount.Caption; //кол-во ящиков
  if n<2 then begin
    s_temp3:='0.00';
    s_temp4:='0.00';
    goto m1;
    end;
  if n=2 then begin s_temp3:='0.00'; goto m2; end;
  //предыдущий вес
  case wnumb of
    0: s_temp3:=PltWghts1[n-2].WThisMassa;
    1: s_temp3:=PltWghts2[n-2].WThisMassa;
    2: s_temp3:=PltWghts3[n-2].WThisMassa;
    3: s_temp3:=PltWghts4[n-2].WThisMassa;
    4: s_temp3:=PltWghts5[n-2].WThisMassa;
    5: s_temp3:=PltWghts6[n-2].WThisMassa;
    6: s_temp3:=PltWghts7[n-2].WThisMassa;
    7: s_temp3:=PltWghts8[n-2].WThisMassa;
    8: s_temp3:=PltWghts9[n-2].WThisMassa;
    end;
  if length(s_temp3)>3 then s_temp3:=copy(s_temp3,1,length(s_temp3)-3)+'.'+copy(s_temp3,length(s_temp3)-2,2);
  //текущий вес
```

```
m2:
  case wnumb of
    0: s_temp4:=PltWghts1[n-1].WThisMassa;
    1: s_temp4:=PltWghts2[n-1].WThisMassa;
    2: s_temp4:=PltWghts3[n-1].WThisMassa;
    3: s_temp4:=PltWghts4[n-1].WThisMassa;
    4: s_temp4:=PltWghts5[n-1].WThisMassa;
    5: s_temp4:=PltWghts6[n-1].WThisMassa;
    6: s_temp4:=PltWghts7[n-1].WThisMassa;
    7: s_temp4:=PltWghts8[n-1].WThisMassa;
    8: s_temp4:=PltWghts9[n-1].WThisMassa;
    end;
  if length(s_temp4)>3 then s_temp4:=copy(s_temp4,1,length(s_temp4)-3)+'.'+copy(s_temp4,length(s_temp4)-2,2);

m1:
  //формируем пакет для отправки на дисплей
  s_temp:=ScDisplay.CreateDataForDisplay(wnumb,s_temp,s_temp4,s_temp3,s_temp2,btype);
  ScDisplay.DataForDisplay:=s_temp;

  //отправляем на дисплей
  //ScDisplay.SendToDisplay;
  ScDisplay.SendToDisplayTCP;
  s_temp:=IntToStr(ScDisplay.SendedBytesCount)+' '+ScDisplay.DataForDisplay;
  Form_Developer.Memo_Sended.Lines.Add(s_temp);
  {if ScDisplay.SendedBytesCount<31 then for i:=1 to 5 do begin
    ScDisplay.SendToDisplay;
    if ScDisplay.SendedBytesCount>=31 then Break;
    s_temp:=IntToStr(ScDisplay.SendedBytesCount)+' '+ScDisplay.DataForDisplay;
    Form_Developer.Memo_Sended.Lines.Add(s_temp);
    end;}
end;

function addkoma(s:string):string;
var
  s_temp2:string;
  ii,p:integer;
begin
  s_temp2:=s;
  ii:=length(s_temp2);
  if ii>3 then s_temp2:=copy(s_temp2,1,ii-3)+','+copy(s_temp2,ii-2,3);
  if ii<3 then for p:=1 to 3-ii do s_temp2:='0'+s_temp2;
  if length(s_temp2)=3 then s_temp2:='0,'+s_temp2;
  Result:=s_temp2;
end;

//Show weighting in table
procedure ShowWInTable;
var
  s_temp,s_temp2:string;
  i,j,k,c,m,p,ii:integer;
begin
  j:=length(PltWghts1)+1;
  Form_Main.StringGrid_SC01.RowCount:=j+1;
  Form_Main.StringGrid_SC01.Rows[j].Clear;
  if j=1 then Exit;
  for i:=1 to j-1 do begin
    Form_Main.StringGrid_SC01.Cells[0,i]:=PltWghts1[i-1].WDate;//Дата
    Form_Main.StringGrid_SC01.Cells[1,i]:=PltWghts1[i-1].WTime;//Время
    Form_Main.StringGrid_SC01.Cells[2,i]:=PltWghts1[i-1].WCargoName;//Наименование товара
    //Вес нормального и плохого ящика,г
    if (PltWghts1[i-1].WType='1') or (PltWghts1[i-1].WType='0')
      or (PltWghts1[i-1].WType='5') or (PltWghts1[i-1].WType='6')
      then begin
      s_temp2:=PltWghts1[i-1].WThisMassa;
      s_temp2:=addkoma(s_temp2);
```

```
      Form_Main.StringGrid_SC01.Cells[3,i]:=s_temp2;
      PrevBox[1]:=LastBox[1];
      LastBox[1]:=PltWghts1[i-1].WThisMassa;
      end else Form_Main.StringGrid_SC01.Cells[3,i]:='';
    if (PltWghts1[i-1].WType='0') or (PltWghts1[i-1].WType='7') then begin//Если ящик ненормального веса <min
      ColorRowRed(i);//строка красным цветом
      end;
    if (PltWghts1[i-1].WType='0') and (i=j-1) then begin//Если ящик ненормального веса <min
      WarningShow;//предупреждение о неправильном весе когда первый раз
      end;
    if (PltWghts1[i-1].WType='5') or (PltWghts1[i-1].WType='6') then begin//Если ящик ненормального веса >max или
склеенные ящики
      ColorRowYellow(i);//строка желтым цветом
      end;
    s_temp2:=PltWghts1[i-1].WFullMassa;//Вес полный,г
    s_temp2:=addkoma(s_temp2);
    Form_Main.StringGrid_SC01.Cells[4,i]:=s_temp2;
    if j=2 then //Вес пустой паллеты,г
      Form_Main.StringGrid_SC01.Cells[5,i]:=addkoma(PltWghts1[i-1].WThisMassa)
      else Form_Main.StringGrid_SC01.Cells[5,i]:=Form_Main.StringGrid_SC01.Cells[5,1];
    if (PltWghts1[i-1].WType='2') or (PltWghts1[i-1].WType='4') or (PltWghts1[i-1].WType='7') then//Вес перестила,г
      Form_Main.StringGrid_SC01.Cells[6,i]:=addkoma(PltWghts1[i-1].WThisMassa)
      else Form_Main.StringGrid_SC01.Cells[6,i]:='0';
    Form_Main.StringGrid_SC01.Cells[7,i]:=PltWghts1[i-1].WTaskDate;//Дата наряда
    Form_Main.StringGrid_SC01.Cells[8,i]:=PltWghts1[i-1].WWorkShift;//Номер смены
    //Кол-во ящиков
    Form_Main.StringGrid_SC01.Cells[9,i]:=PltWghts1[i-1].WBoxCount;
  end;
  //выводим в метки данные последней строки
  Form_Main.Label_SummaWeight01.Caption:=Form_Main.StringGrid_SC01.Cells[4,j-1];
  for i:=j-1 downto 1 do
     if Form_Main.StringGrid_SC01.Cells[3,i]<>'0' then begin
      Form_Main.Label_LastBox01.Caption:=Form_Main.StringGrid_SC01.Cells[3,i];
      Break;
      end;
  Form_Main.Label_BoxCount01.Caption:=Form_Main.StringGrid_SC01.Cells[9,j-1];
  s_temp:=PltWghts1[i-1].WBadBoxCount;
  Form_main.Label_BadBoxCount01.Caption:=s_temp;
  if s_temp='0' then Form_main.Label_BadBoxCount01.Font.Color:=clGreen
    else Form_main.Label_BadBoxCount01.Font.Color:=clRed;
  Form_Main.Label_EmptyPallette01.Caption:=Form_Main.StringGrid_SC01.Cells[5,1];
  k:=0;
  for i:=1 to j-1 do
    if Form_Main.StringGrid_SC01.Cells[6,i]<>'0' then begin
      s_temp:=Form_Main.StringGrid_SC01.Cells[6,i];
      val(s_temp,m,c);
      k:=k+m;
      end;
  Form_Main.Label_Septum01.Caption:=IntToStr(k);
  Form_Main.StringGrid_SC01.Row:=Form_Main.StringGrid_SC01.RowCount-2;
end;

//удаление n последних строк из файла с именем FName
procedure DeleteLastFromFile(FName:string; n:integer);
var
  i:integer;
  F:TStringList;
begin
  F:=TStringList.Create;
  F.LoadFromFile(FName);
  for i := 1 to n do
    F.Delete(F.Count-i);
  F.SaveToFile(FName);
  F.Free;
end;
```

```
//удаление элемента с индексом k с массива PltWghts
//и вносим изменения в массивы красных и желтых строк если они совпадают с удаляемой
procedure DelPltWghtsElem(k:integer);
var
  i,j,n:integer;
  bcount,bbcount:integer;
  lastsum:integer;
  s_temp3:string;
begin
  if (k >= 0) and (k <= High(PltWghts1)) then begin
    //делаем отметку в БД от том что запись удалена
    s_temp3:='UPDATE Weightings SET Wactivity=0 WHERE Id_weight='+IntToStr(PltWghts1[k].WDBID)+';';
    Form_Main.ADOQuery1.Active:=false;
    Form_Main.ADOQuery1.SQL.Text:=s_temp3;
    Form_Main.ADOQuery1.ExecSQL;
    //Form_Main.ADOQuery1.Active:=true;
    //перемещаем записи
    for i := k to High(PltWghts1) - 1 do
      PltWghts1[i] := PltWghts1[i + 1];
    //Устанавливаем новый размер массива.
    n:=Length(PltWghts1) - 2;
    SetLength(PltWghts1, n);
  end else Exit;
  //корректируем суммарный вес в элементах массива начиная с удаляемого на текущую массу
  if k>0 then lastsum:=PltWghts1[k-1].WFullMassaInt else lastsum:=0;
  for i:=k to n-1 do begin
    lastsum:=lastsum+PltWghts1[i].WThisMassaInt;
    PltWghts1[i].WFullMassaInt:=lastsum;
    PltWghts1[i].WFullMassa:=IntToStr(lastsum);
    end;

  //обновляем массивы красных и желтых строк
  setlength(RedRows,0);
  setlength(YellowRows,0);

  //считаем количество ящиков перебирая записи массива и расставляя соответственно элементы
  bcount:=0;
  if n>0 then for i:=0 to n do begin
     if (PltWghts1[i].WType='1') or (PltWghts1[i].WType='0') or (PltWghts1[i].WType='5')
        then bcount:=bcount+1;
     PltWghts1[i].WBoxCount:=IntToStr(bcount);
     end;
  bbcount:=0;
  if n>0 then for i:=0 to n do begin
     if (PltWghts1[i].WType='0') or (PltWghts1[i].WType='5') then
        bbcount:=bbcount+1;
     PltWghts1[i].WBadBoxCount:=IntToStr(bbcount);
     end;

end;

//проверяем массу на уменьшение
//если она стала меньше на значение больше чем минимальная масса шага
//смотрим на сколько шагов назад произошло уменьшение т.е. что сняли
//и удалем то количество элементов массива которое было снято
//также делаем неактивными те записи в БД которые относились к этим грузам
procedure TestMinusCargo(w,n:integer); //w - разница массы, n - размер массива взвешиваний
var
  i,j:integer;
  s_temp3:string;
  k,k1,m,df:integer;
  maxmassa,minmassa:integer;
begin
  if (Main.ScModeType[0]=1) then begin SaveWToFile; Exit; end;
```

```
  if (w<0) and (abs(w)>Limits.Septum_Min) and (n>1) then begin
    maxmassa:=abs(w)+Limits.Septum_Min;
    minmassa:=abs(w)-Limits.Septum_Min;
    //сначала проверяем на снятие одного груза - проверяем начиная с последнего
    for i:=n-1 downto 0 do
      if (PltWghts1[i].WThisMassaInt>=minmassa) and (PltWghts1[i].WThisMassaInt<=maxmassa) then
      begin
      DelPltWghtsElem(i);
      k:=length(PltWghts1);
      if k>0 then begin
        m:=round(ThisWeight[1]*1000);
        PltWghts1[k-1].WFullMassaInt:=m;
        PltWghts1[k-1].WFullMassa:=IntTostr(m);
        end;
      Break;
      end;

    //сохраняем массив PltWghts в выходной файл удаляем предыдущие записи
    //используем когда происходили изменения в минус в файле
    //n - номер весов
    SavePltToFile(1);
    end else SaveWToFile; //save weighting to output file
end;

//Create Weighting type Record for write to all stores
procedure CreateWRecord;
var
  i,j,n,k,k1,m,df:integer;
  s_temp,s_temp3:string;
  w,bbcount,bcount:integer;
  d_temp:TDateTime;
  porog:integer;
begin
  if Main.ScModeType[0]=0 then porog:=0 else porog:=-1;
  bcount:=0;//кол-во ящиков
  bbcount:=0;//кол-во ящиков с неправильным весом
  w:=round(ThisWeight[1]*1000)-round(LastWeight[1]*1000);
  n:=length(PltWghts1);
  setlength(PltWghts1,n+1);
//создаем новую запись для этой паллеты используя последнюю запись в БД
  //читаем номер последней записи в БД
  s_temp3:='SELECT MAX(Id_weight) AS MAX_ID FROM Weightings;';
  Form_Main.ADOQuery1.Active:=false;
  Form_Main.ADOQuery1.SQL.Text:=s_temp3;
  //Form_Main.ADOQuery1.ExecSQL;
  Form_Main.ADOQuery1.Active:=true;
  k:=Form_Main.ADOQuery1.FieldByName('MAX_ID').AsInteger;
  //читаем содержимое последней записи
  s_temp3:='SELECT * FROM Weightings WHERE Id_weight='+IntToStr(k)+';';
  Form_Main.ADOQuery1.Active:=false;
  Form_Main.ADOQuery1.SQL.Text:=s_temp3;
  //Form_Main.ADOQuery1.ExecSQL;
  Form_Main.ADOQuery1.Active:=true;
  //заполняем массив
  d_temp:=Form_Main.ADOQuery1.FieldByName('WDateTime').AsDateTime;
  PltWghts1[n].WDBID:=k;
  PltWghts1[n].WDate:=DateToStr(d_temp);
  PltWghts1[n].WTime:=TimeToStr(d_temp);
  PltWghts1[n].WThisMassa:=IntToStr(w);
  PltWghts1[n].WThisMassaInt:=w;
  PltWghts1[n].WFullMassa:=Form_Main.ADOQuery1.FieldByName('Massa').AsString;
  PltWghts1[n].WFullMassaInt:=Form_Main.ADOQuery1.FieldByName('Massa').AsInteger;
  PltWghts1[n].WCargoID:=Form_Main.ADOQuery1.FieldByName('Cargo_Id').AsString;
  PltWghts1[n].WCargoName:=Form_Main.Label_CargoType01.Caption;
```

```
  PltWghts1[n].WUserID:=Form_Main.ADOQuery1.FieldByName('User_Id').AsString;
  PltWghts1[n].WWorkShift:=Form_Main.ADOQuery1.FieldByName('WorkShift').AsString;
  PltWghts1[n].WPallete:=Form_Main.ADOQuery1.FieldByName('Pallette').AsString;
  PltWghts1[n].WType:=Form_Main.ADOQuery1.FieldByName('WType').AsString;
  PltWghts1[n].WScaleID:=Form_Main.ADOQuery1.FieldByName('Scale_Id').AsString;
  d_temp:=Form_Main.ADOQuery1.FieldByName('TaskDate').AsDateTime;
  PltWghts1[n].WTaskDate:=DateToStr(d_temp);
  //считаем количество ящиков перебирая записи массива
  if n>porog then for i:=0 to n do
     if (PltWghts1[i].WType='1') or (PltWghts1[i].WType='0') or (PltWghts1[i].WType='5')
        then bcount:=bcount+1;
  if n>porog then for i:=0 to n do
     if (PltWghts1[i].WType='6')
        then bcount:=bcount+1;
  PltWghts1[n].WBoxCount:=IntToStr(bcount);
  if n>0 then for i:=0 to n do
     if (PltWghts1[i].WType='0') or (PltWghts1[i].WType='5') then bbcount:=bbcount+1;
  PltWghts1[n].WBadBoxCount:=IntToStr(bbcount);
  //проверяем на склеенные ящики и корректируем массив если несколько ящиков одновременно
  if (SomeBoxes) and (PltWghts1[n].WType='6') then begin
    n:=n+1;
    setlength(PltWghts1,n+1);
    PltWghts1[n-1].WThisMassaInt:=Round(w/2);
    PltWghts1[n-1].WThisMassa:=IntToStr(Round(w/2));
    PltWghts1[n]:=PltWghts1[n-1];
    PltWghts1[n-1].WFullMassaInt:=PltWghts1[n-1].WFullMassaInt-PltWghts1[n-1].WThisMassaInt;
    PltWghts1[n-1].WFullMassa:=IntToStr(PltWghts1[n-1].WFullMassaInt);
    PltWghts1[n-1].WBoxCount:=IntToStr(bcount);
    PltWghts1[n].WBoxCount:=IntToStr(bcount+1);
    end;
//проверяем ее на уменьшение и сохраняем в файл
  TestMinusCargo(w,n);
end;

//Find what is this (box, septum, empty pallette, angle)
// 0 - unknown, 1 - good box, 2 - septum, 3 - empty pallette,
//4 - angle, 5 - bigbox, 6 - two boxes, 7 - minus weight
function FindCargoType:integer;
var
  TestW:integer;
begin
  TestW:=(round(ThisWeight[1]*1000)-round(LastWeight[1]*1000));
  Result:=0;
  if TestW<0 then begin
    Result:=7;
    Exit;
    end;
  //if it first weighting in new pallete it is Pallette weight
  if (Main.ScModeType[0]=0) and (length(PltWghts1)=0) then begin
    Result:=3;
    Exit;
    end;
  //the smallest weight is septum
  if (TestW>=Limits.Septum_Min) and (TestW<=Limits.Septum_Max) then begin
    Result:=2;
    Exit;
    end;
  //next weight is angle
  if (TestW>Limits.Septum_Max) and (TestW>=Limits.Angle_Min) and
    (TestW<=Limits.Angle_Max) then begin
    Result:=4;
    Exit;
    end;
  //biggest weight is box
  if (TestW>Limits.Angle_Max) and (TestW>=BoxMin[1]) and (TestW<=BoxMax[1]) then begin
```

```
    Result:=1;
    Exit;
    end;
  if (TestW>=BoxMax[1]) then begin
    if Limits.SomeBoxes then if (TestW>=(2*BoxMin[1])) and (TestW<=(2*BoxMax[1])) then begin
      Result:=6;
      Exit;
      end;
    Result:=5;
    Exit;
    end;
end;

//Test and save weight if it need
procedure SaveWeight;
var
  i:integer;
  TestW:integer;
  s_temp:string;
begin
  i:=round(ThisWeight[1]*1000);
  TestW:=round(abs(ThisWeight[1]-LastWeight[1])*1000);
  if Main.ScModeType[0]=1 then begin //в штучном режиме не записываем если
  //предыдущий вес не был 0+минимальный вес перестила
    if round(LastWeight[1]*1000)>Limits.Septum_Min then Exit;
    end;
  if (ThisWeight[1]<>LastWeight[1]) and (Scales.stab_ind) and
  (i>=Limits.Septum_Min) and (TestW>Limits.Septum_Min) then begin
    CargoType[1]:=FindCargoType; // 0 - unknown, 1 - good box, 2 - septum, 3 - empty pallette,
    //4 - angle, 6 - some boxes
    SaveWToDB; //save weighting to database

    s_temp:=FormatFloat('#.#',ThisWeight[1])+' '+FormatFloat('#.#',LastWeight[1]);
    Form_developer.Memo_SaveWeight.Lines.Add(s_temp);

    CreateWRecord;
    ShowWInTable; //show weighting in stringgrid table
//    SaveWToFile; //save weighting to output file
  //отправляем текущие показания на выносное табло если порт установлен
    if ScDisplay.DisplayPort>0 then begin
      SendWghtToDisplay(0,ThisWeight[1]);
      end;
    end;
end;

//функция преобразования чтроки в символьный набор байт
function ConvertStrToHEX(s:string):string;
var
  i,n:integer;
  s_temp:string;
begin
  s_temp:='';
  Result:='';
  n:=length(s);
  if n<1 then Exit;
  for i:=1 to n do
    s_temp:=s_temp+' '+IntToHex(ord(s[i]),2);
  Result:=s_temp;
end;

procedure ReadWeight;
var
  s_temp:string;
  fromscales:string;
  port_n:byte;
```

```
  n,i:integer;
  tw:real;
begin
  if Paused then Exit;
  port_n:=Main.ScCOM[0];
  if Form_Main.CheckBox_Emulation.Checked then begin
      s_temp:=Form_Emulation.LabeledEdit_Value.Text;
      //s_temp:=StringReplace(s_temp, ',', '.', [rfReplaceAll, rfIgnoreCase]);
      fromscales:=EmulateWeight(ScAddr[0], s_temp); end
      else begin
        //fromscales:=Scales.GetScaleData(Main.ScAddr[0],port_n);
        fromscales:=Scales.GetScaleData(0);
        end;
  //LastWeight:=ThisWeight;
  n:=length(PltWghts1);
  if n>0 then LastWeight[1]:=PltWghts1[n-1].WFullMassaInt/1000
    else LastWeight[1]:=0;
  if fromscales='' then Exit;
  ThisWeight[1]:=Scales.GetWeightVN(fromscales);
  tw:=ThisWeight[1];
  Last5W[1]:=Last5W[2];
  Last5W[2]:=Last5W[3];
  Last5W[3]:=Last5W[4];
  Last5W[4]:=Last5W[5];
  Last5W[5]:=round(tw*1000);
  Form_Developer.Memo_ScaleData.Lines.Add(fromscales+' HEX:'+ConvertStrToHEX(fromscales));//!!!
  Developer.ShowScaleParameters;
  s_temp:=FormatFloat('#0.000',ThisWeight[1]);
  Form_Main.LabeledEdit_SC01Weight.Text:=s_temp;
  Form_Developer.Label_OutputFile.Caption:=ScOutFolder[0]+'\вес.txt';
  //show indicators
  if (Scales.stab_ind) then Form_Main.Image_Stab01.Picture.LoadFromFile(MyDir+'\images\green_ring.bmp')
     else Form_Main.Image_Stab01.Picture.LoadFromFile(MyDir+'\images\red_ring.bmp');
  if (Scales.tara_ind) then Form_Main.Image_Tara01.Picture.LoadFromFile(MyDir+'\images\green_ring.bmp')
     else Form_Main.Image_Tara01.Picture.LoadFromFile(MyDir+'\images\red_ring.bmp');
  if (Scales.zero_ind) then Form_Main.Image_Zero01.Picture.LoadFromFile(MyDir+'\images\green_ring.bmp')
     else Form_Main.Image_Zero01.Picture.LoadFromFile(MyDir+'\images\red_ring.bmp');
  //test weight and save if it need
  if (PalletteStarted[1]) and (Last5W[5]=Last5W[4]) and (Last5W[4]=Last5W[3])
    and (Last5W[3]=Last5W[2]) and (Last5W[2]=Last5W[1])
    then SaveWeight;
end;

procedure TForm_Main.Button_GetWeight01Click(Sender: TObject);
var
  s_temp:string;
  fromscales:string;
  port_n:byte;
  n,i,c:integer;
  tw:real;
  tempLastW:real;
begin
  if Paused then Exit;

  n:=length(PltWghts1);
  LastWeight[1]:=0;
  if (n>0) then LastWeight[1]:=PltWghts1[n-1].WFullMassaInt/1000;

  fromscales:=Form_Main.LabeledEdit_SC01Weight.Text;
  fromscales:=StringReplace(fromscales, ',', '.', [rfReplaceAll, rfIgnoreCase]);
  if fromscales='' then Exit;
  val(fromscales,ThisWeight[1],c);
  tw:=ThisWeight[1];
  Last5W[1]:=Last5W[2];
  Last5W[2]:=Last5W[3];
```

```
  Last5W[3]:=Last5W[4];
  Last5W[4]:=Last5W[5];
  Last5W[5]:=round(tw*1000);
  //Form_Developer.Label_OutputFile.Caption:=ScOutFolder[0]+'\вес.txt';
  //test weight and save if it need
  if (PalletteStarted[1]) and (Last5W[5]=Last5W[4]) and (Last5W[4]=Last5W[3])
    and (Last5W[3]=Last5W[2]) and (Last5W[2]=Last5W[1])
    then begin
      if (WasZero[1]) and (Main.ScModeType[0]=1) then LastWeight[1]:=0;
      if (tw*1000>Limits.Septum_Min) then WasZero[1]:=false else WasZero[1]:=true;
      SaveWeight;
      end;
end;

procedure TForm_Main.BitBtn_Summa01Click(Sender: TObject);
begin
  Scales.SummaClick(Main.ScAddr[0]);
end;

procedure TForm_Main.BitBtn_LongSumma01Click(Sender: TObject);
begin
  Scales.LongSummaClick(Main.ScAddr[0]);
end;

procedure TForm_Main.BitBtn_Zero01Click(Sender: TObject);
begin
  Scales.ZeroClick(Main.ScAddr[0]);
end;

procedure TForm_Main.BitBtn_Tara01Click(Sender: TObject);
begin
  Scales.TaraClick(Main.ScAddr[0]);
end;

procedure TForm_Main.Button_New_Pallete01Click(Sender: TObject);
var
  i,n,tabind:integer;
  s_temp3:string;
begin
  if (Form_Main.Label_Port01.Color=clRed) and (not Form_Main.CheckBox_Emulation.Checked)
  then begin
    showmessage('Нет связи с терминалом');
    Exit;
    end;

  Form_Main.BitBtn_Pause01.Visible:=true;
  Paused:=false;
  for i:=1 to 5 do
    Last5W[i]:=0;
  PrevBox[1]:='0';
  LastBox[1]:='0';
  FileWriteStarted:=false; //ставим признак что начало паллеты еще не записано
  setlength(RedRows,0);//обнуляем список красных строк
  setlength(YellowRows,0);//обнуляем список красных строк
  Main.LoadTask(ScInpFolder[0]+'\'+ScInFName[0]);
  if Form_Main.Label_CargoID01.Caption='Label_CargoID01' then begin
    showmessage('Не загружен наряд - проверьте существование файла наряда или настройки пути');
    Exit;
    end;
    PalletteStarted[1]:=true;
    LastWeight[1]:=0;
    ThisWeight[1]:=0;

    s_temp3:='SELECT MAX(Pallette) AS MaxPlt FROM Weightings;';
    Form_Main.ADOQuery1.Active:=false;
```

```
    Form_Main.ADOQuery1.SQL.Text:=s_temp3;
    //Form_Main.ADOQuery1.ExecSQL;
    Form_Main.ADOQuery1.Active:=true;
    if Form_Main.ADOQuery1.RecordCount>0 then
      n:=Form_Main.ADOQuery1.FieldByName('MaxPlt').AsInteger
      else n:=0;
    Plt_ID:=n+1;

    Form_Main.Button_New_Pallete01.Enabled:=false;
    Form_main.Button_End_Pallette01.Enabled:=true;
end;

procedure TForm_Main.N_SavesClick(Sender: TObject);
begin
  Form_StoreDest.Show;
end;

procedure TForm_Main.N_UsersSetupClick(Sender: TObject);
begin
  Form_Users.Show;
end;

procedure TForm_Main.N_ParametersClick(Sender: TObject);
begin
  Form_Limits.Show;
end;

//Запись в БД параметров смены пользователя
procedure ChangeUserWrite;
var
  s_temp3:string;
  eid:integer;
  edate:string;
begin
  s_temp3:='SELECT MAX(Id_enterance) AS ENT_MAX FROM Enterance;';
  Form_Main.ADOQuery1.Active:=false;
  Form_Main.ADOQuery1.SQL.Text:=s_temp3;
  Form_Main.ADOQuery1.Active:=true;
  if Form_Main.ADOQuery1.RecordCount=0 then eid:=0 else
    eid:=Form_Main.ADOQuery1.FieldByName('ENT_MAX').AsInteger+1;

  edate:=DateToStr(Now);
  edate:=copy(edate,7,4)+'-'+copy(edate,4,2)+'-'+copy(edate,1,2);
  edate:=""+edate+' '+TimeToStr(Now)+""; //формат записи "2018-07-10 20:55:00"
  s_temp3:='INSERT INTO Enterance VALUES ('+IntToStr(eid)+', '+IntToStr(user_id);
  s_temp3:=s_temp3+', '+edate+', 3);';

  Form_Main.ADOQuery1.Active:=false;
  Form_Main.ADOQuery1.SQL.Text:=s_temp3;
  Form_Main.ADOQuery1.ExecSQL;
//  Form_Main.ADOQuery1.Active:=true;
end;

procedure TForm_Main.N_UserChangeClick(Sender: TObject);
begin
  ChangeUserWrite;
  Form_Authorisation.Show;
  Form_Main.Hide;
end;

procedure TForm_Main.FormCloseQuery(Sender: TObject;
  var CanClose: Boolean);
var
  i:integer;
begin
```

```
  CanClose := false;
  for i:=1 to Main.ScalesCount do
    if Main.PalletteStarted[i] then showmessage('У Вас не закрыта паллета на участке №'+IntToStr(i));
  ExitProgram;
end;

procedure TForm_Main.N_ShowSetupClick(Sender: TObject);
begin
  Form_ScalesSetup.Show;
end;

//читаем файл наряда
procedure LoadTask(taskfile:string);
var
  fname:textfile;
  s_temp,s_temp3:string;
  i,c,n,rc:integer;
  minbox,maxbox:integer;
  minb,maxb:real;
  date_s,wshift,wdate,cname,ccode:string;
  ws_i:integer;
begin
  if (not fileexists(taskfile)) then begin
    showmessage('Не найден указанный в настройках файл наряда');
    Exit;
    end;
  assignfile(fname,taskfile);
  reset(fname);
  readln(fname,s_temp); //read Date
  date_s:=s_temp;
  Form_Main.Label_TaskDate01.Caption:=date_s;
  readln(fname,s_temp); //read workshift
  wshift:=s_temp;
  Form_Main.Label_WorkShift01.Caption:=wshift;
  readln(fname,s_temp); //read cargo code
  ccode:=trim(s_temp);
  readln(fname,s_temp); //read cargo name
  cname:=s_temp;
  readln(fname,s_temp); //read min box
  val(s_temp,minb,c);
  minbox:=round(minb*1000);
  BoxMin[1]:=minbox;
  readln(fname,s_temp); //read max box
  val(s_temp,maxb,c);
  maxbox:=round(maxb*1000);
  BoxMax[1]:=maxbox;
  closefile(fname);

  Form_Main.Label_CargoType01.Caption:=cname;
  //экранируем кавычки если они есть в имени товара
  cname:=StringReplace(cname, '"', ' ', [rfReplaceAll, rfIgnoreCase]);
  cname:=StringReplace(cname, '"', ' ', [rfReplaceAll, rfIgnoreCase]);

  Form_Main.ADOQuery1.Active:=false;  //read DB
  s_temp3:='SELECT Id_cargo FROM Cargos WHERE Cargo_Code='+ccode+';';//test if we not have this cargo we add it;';
//test if we not have this cargo we add it
  Form_Main.ADOQuery1.SQL.Text:=s_temp3;
  Form_Main.ADOQuery1.Active:=true;
  n:=Form_Main.ADOQuery1.RecordCount;
  //if record's count is zero we add cargo to catalog
  if n=0 then begin
    s_temp3:='SELECT Id_cargo FROM Cargos;';//find count of records in cargo table
    Form_Main.ADOQuery1.Active:=false;
    Form_Main.ADOQuery1.SQL.Text:=s_temp3;
    Form_Main.ADOQuery1.Active:=true;
```

```
    rc:=Form_Main.ADOQuery1.RecordCount;

    s_temp3:='INSERT INTO Cargos VALUES ('+inttostr(rc+1)+', '"+cname+'", ';//add record with new cargo code (readed
from file)
    s_temp3:=s_temp3+'NULL, '+ccode+', '+inttostr(minbox)+', '+inttostr(maxbox)+', ';
    s_temp3:=s_temp3+'NULL, NULL, NULL, NULL, 1);';
    Form_Main.ADOQuery1.Active:=false;
    Form_Main.ADOQuery1.SQL.Text:=s_temp3;
    Form_Main.ADOQuery1.ExecSQL;
    //Form_Main.ADOQuery1.Active:=true;
    CargoCatalog.LoadCargos;//refresh cargo list in CargoCatalog Form
    end else begin rc:=Form_Main.ADOQuery1.FieldByName('Id_cargo').AsInteger; rc:=rc-1; end;

  Form_Main.Label_CargoID01.Caption:=inttostr(rc+1);
  Form_Main.Label_MinBox01.Caption:=inttostr(minbox);
  Form_Main.Label_MaxBox01.Caption:=inttostr(maxbox);
end;

procedure TForm_Main.FormShow(Sender: TObject);
var
  s_temp:string;
begin
  s_temp:=Datetostr(Now);
  Form_Main.Label_Date01.Caption:=s_temp;
  Form_Main.Label_Date02.Caption:=s_temp;
  Form_Main.Label_Date03.Caption:=s_temp;
  Form_Main.Label_Date04.Caption:=s_temp;
  Form_Main.Label_Date05.Caption:=s_temp;
  Form_Main.Label_Date06.Caption:=s_temp;
  Form_Main.Label_Date07.Caption:=s_temp;
  Form_Main.Label_Date08.Caption:=s_temp;
  Form_Main.Label_Date09.Caption:=s_temp;
end;

procedure WShiftButVisibility(vis:boolean);
begin
  Form_Main.Button_1WShift01.Visible:=vis;
  Form_Main.Button_2WShift01.Visible:=vis;
  Form_Main.Button_3WShift01.Visible:=vis;
end;

procedure TForm_Main.Button_ChangeWShift01Click(Sender: TObject);
begin
  Form_Main.Button_ChangeWShift01.Visible:=false;
  WShiftButVisibility(true);
end;

procedure TForm_Main.Button_1WShift01Click(Sender: TObject);
begin
  Form_Main.Label_WorkShift01.Caption:='Первая';
  Form_Main.Button_ChangeWShift01.Visible:=true;
  WShiftButVisibility(false);
end;

procedure TForm_Main.Button_2WShift01Click(Sender: TObject);
begin
  Form_Main.Label_WorkShift01.Caption:='Вторая';
  Form_Main.Button_ChangeWShift01.Visible:=true;
  WShiftButVisibility(false);
end;

procedure TForm_Main.Button_3WShift01Click(Sender: TObject);
begin
  Form_Main.Label_WorkShift01.Caption:='Третья';
  Form_Main.Button_ChangeWShift01.Visible:=true;
```

```
  WShiftButVisibility(false);
end;

//показываем состояние индикаторов
procedure ShowIndicators(sc_numb:integer);
var
  Image_Stab,Image_Tara,Image_Zero:TImage;
  ImageName:string;
begin
  ImageName:='Image_Stab0'+IntToStr(sc_numb+1);
  Image_Stab:=TImage(Form_Main.FindComponent(ImageName));
  ImageName:='Image_Tara0'+IntToStr(sc_numb+1);
  Image_Tara:=TImage(Form_Main.FindComponent(ImageName));
  ImageName:='Image_Zero0'+IntToStr(sc_numb+1);
  Image_Zero:=TImage(Form_Main.FindComponent(ImageName));
  if (Scales.stab_ind) then Image_Stab.Picture.LoadFromFile(MyDir+'\images\green_ring.bmp')
     else Image_Stab.Picture.LoadFromFile(MyDir+'\images\red_ring.bmp');
  if (Scales.tara_ind) then Image_Tara.Picture.LoadFromFile(MyDir+'\images\green_ring.bmp')
     else Image_Tara.Picture.LoadFromFile(MyDir+'\images\red_ring.bmp');
  if (Scales.zero_ind) then Image_Zero.Picture.LoadFromFile(MyDir+'\images\green_ring.bmp')
     else Image_Zero.Picture.LoadFromFile(MyDir+'\images\red_ring.bmp');
end;

//Чтение данных с весов номер n и установка соответствующих индикаторов и значения веса в поле
//n от 0
procedure ReadScalesTCP(sc_numb:integer);
var
  fromscales:string;
  ThWght:real;
  s_temp:string;
  port_n:byte;
  LabeledEdit_Weight:TLabeledEdit;
  Button_GetWeight:TButton;
  LEName:string;
  BName:string;
begin
  LEName:='LabeledEdit_SC0'+IntToStr(sc_numb+1)+'Weight';
  BName:='Button_GetWeight0'+IntToStr(sc_numb+1);
  LabeledEdit_Weight:=TLabeledEdit(Form_Main.FindComponent(LEName));
  Button_GetWeight:=TButton(Form_Main.FindComponent(BName));
  if Form_Main.CheckBox_Emulation.Checked then begin
    s_temp:=Form_Emulation.LabeledEdit_Value.Text;
    fromscales:=EmulateWeight(ScAddr[sc_numb], s_temp); end
    else begin
      fromscales:=Scales.GetScaleDataTCP(sc_numb);
      Form_Main.Label_FromTCP.Caption:=fromscales;
      end;
  if (fromscales='-1') or (fromscales='') then Exit;
  if ScType[sc_numb]=1 then ThWght:=Scales.GetWeightVN(fromscales);
  if ScType[sc_numb]=2 then begin
    if length(fromscales)<16 then Exit;
    ThWght:=Scales.GetWeightT1(fromscales);
    end;
  //Form_Developer.Label_DataFrom.Caption:=fromscales; //!!!
  s_temp:=FormatFloat('#0.000',ThWght);
  LabeledEdit_Weight.Text:=s_temp;
  //show indicators
  ShowIndicators(sc_numb);
  //Form_Developer.Memo_ScaleData.Lines.Add(fromscales+' HEX:'+ConvertStrToHEX(fromscales));//!!!
  //Developer.ShowScaleParameters;
  if (PalletteStarted[sc_numb+1]) and (Scales.stab_ind) then Button_GetWeight.Click else;
end;

//Чтение данных с весов номер n и установка соответствующих индикаторов и значения веса в поле
//n от 0
```

```
procedure ReadScales(n:integer);
var
  fromscales:string;
  ThWght:real;
  s_temp:string;
  port_n:byte;
begin
  if (Form_Main.Label_Port01.Font.Color<>clRed) or (Form_Main.CheckBox_Emulation.Checked) then begin
    port_n:=Main.ScCOM[0];
    if Form_Main.CheckBox_Emulation.Checked then begin
      s_temp:=Form_Emulation.LabeledEdit_Value.Text;
      fromscales:=EmulateWeight(ScAddr[0], s_temp); end
      else begin
        fromscales:=Scales.GetScaleData(0);
        //fromscales:=Scales.GetScaleDataTCP(0);
        Form_Main.Label_FromTCP.Caption:=fromscales;
        end;
    if fromscales='-1' then Exit;
    if ScType[0]=1 then ThWght:=Scales.GetWeightVN(fromscales);
    if ScType[0]=2 then begin
      if length(fromscales)<16 then Exit;
      ThWght:=Scales.GetWeightT1(fromscales);
      end;
    Form_Developer.Label_DataFrom.Caption:=fromscales; //!!!

    s_temp:=FormatFloat('#0.000',ThWght);
    Form_Main.LabeledEdit_SC01Weight.Text:=s_temp;
    //show indicators
    ShowIndicators(n);
    Form_Developer.Memo_ScaleData.Lines.Add(fromscales+' HEX:'+ConvertStrToHEX(fromscales));//!!!
    Developer.ShowScaleParameters;
    end;
   if (PalletteStarted[n+1]) and (Scales.stab_ind) then Form_Main.Button_GetWeight01.Click else;
end;

procedure TForm_Main.Timer1Timer(Sender: TObject);
var
  i:integer;
begin
  if not Authorisation.autorized then begin Authorisation.LogIn;
    Authorisation.autorized:=true;
    end;
  //ReadScales(0);
  for i:=0 to Main.ScalesCount-1 do
    ReadScalesTCP(i);
end;

procedure TForm_Main.CheckBox_Emulation01_1Click(Sender: TObject);
begin
  if Form_Main.CheckBox_Emulation.Checked then Form_Emulation.Show;
end;

procedure TForm_Main.N_AboutClick(Sender: TObject);
begin
  Form_About.Show;
end;

//указываем в БД что паллета закрыта
procedure ClosePallete(pid:integer);
var
  s_temp3:string;
begin
  s_temp3:='UPDATE Weightings SET Closed=0 WHERE Pallette='+IntToStr(pid)+';';
  Form_Main.ADOQuery1.Active:=false;
  Form_Main.ADOQuery1.SQL.Text:=s_temp3;
```

```
  Form_Main.ADOQuery1.ExecSQL;
  //Form_Main.ADOQuery1.Active:=true;
end;

procedure TForm_Main.Button_End_Pallette01Click(Sender: TObject);
label m1;
var
  i,n,tabind:integer;
  fname:textfile;
  s_temp:string;
begin
  Form_Main.BitBtn_Pause01.Visible:=false;
  Paused:=false;
  PalletteStarted[1]:=false;
  //пишем в файл последнюю запись
  assignfile(fname,ScOutFolder[0]+'\вес.txt');
  if not fileexists(ScOutFolder[0]+'\вес.txt') then rewrite(fname) else append(fname);
  n:=length(PltWghts1);
  //текущая дата
  s_temp:=DateToStr(Now);
  s_temp:=copy(s_temp,1,6)+copy(s_temp,9,2)+';';
  //текущее время
  s_temp:=s_temp+TimeToStr(Now)+';';
  if n>0 then s_temp:=s_temp+'Закрыли полету '+PltWghts1[n-1].WThisMassa+';'
    else s_temp:=s_temp+'Закрыли полету 0;';
  writeln(fname,s_temp);
  closefile(fname);
  //сбрасываем массив
  setlength(PltWghts1,0);
  Form_Main.Button_New_Pallete01.Enabled:=true;
  Form_Main.Button_End_Pallette01.Enabled:=false;
  n:=Form_Main.StringGrid_SC01.RowCount;
  for i:=1 to n-1 do
    Form_Main.StringGrid_SC01.Rows[i].Clear;
  Form_Main.StringGrid_SC01.RowCount:=2;

  Form_Main.Label_SummaWeight01.Caption:='0';
  Form_Main.Label_LastBox01.Caption:='0';
  Form_Main.Label_BoxCount01.Caption:='0';
  Form_Main.Label_EmptyPallette01.Caption:='0';
  Form_Main.Label_Septum01.Caption:='0';
  Form_main.Label_BadBoxCount01.Font.Color:=clGreen;
  Form_main.Label_BadBoxCount01.Caption:='0';

  //сохраняем последние логи в файл
  Form_Developer.Button_ToScalesToFile.Click;

//  Plt_ID  указываем в БД что паллета закрыта
  ClosePallete(Plt_ID);

//  Form_main.Button_GetWeight01.Click;

  s_temp:='ID001w  0.00L 0.00P 0.00c  0e1';
  ScDisplay.DataForDisplay:=s_temp;
  ScDisplay.SendToDisplay;

  //Form_Developer.Button_SDataToFile.Click; //сохраняем лог данных с весов
  //Form_Developer.Button_ToScalesToFile.Click; //сохраняем лог весов
  Form_Developer.Button_ErrorsToFile.Click;//сохраняем лог ошибок
  //  ScDisplay.COM_Display_Close(DisplayPort);
end;

procedure ShowAllPallettes;
var
  s_temp3:string;
```

```
  i,n:integer;
  wsh,wd:string;
  d_temp:TDateTime;
begin
  //очищаем таблицу
  Form_Main.StringGrid_SC01.RowCount:=2;
  Form_Main.StringGrid_SC01.Rows[1].Clear;
  //Достаем из БД записи по паллетам за текущую дату и с текущей сменой
  wd:=DateToStr(Now);
  wd:='#'+copy(wd,7,4)+'/'+copy(wd,4,2)+'/'+copy(wd,1,2)+'#';
  wsh:=Form_Main.Label_WorkShift01.Caption;
  if wsh='Первая' then wsh:='1';
  if wsh='Вторая' then wsh:='2';
  if wsh='Третья' then wsh:='3';
  s_temp3:='SELECT * FROM Weightings WHERE WDateTime LIKE '+wd;
//  s_temp3:='SELECT * FROM Weightings WHERE WDateTime BETWEEN '+wd+' AND '+wd;
  s_temp3:=s_temp3+' AND WorkShift='+wsh+' ORDER BY WDateTime;';
  Form_Main.ADOQuery1.Active:=false;
  Form_Main.ADOQuery1.SQL.Text:=s_temp3;
  //Form_Main.ADOQuery1.ExecSQL;
  Form_Main.ADOQuery1.Active:=true;
  n:=Form_Main.ADOQuery1.RecordCount;
  if n=0 then Exit;
  for i:=1 to n do begin
    Form_Main.StringGrid_SC01.RowCount:=i+1;
    d_temp:=Form_Main.ADOQuery1.FieldByName('WDateTime').AsDateTime;
    Form_Main.StringGrid_SC01.Cells[0,i]:=DateToStr(d_temp);//Дата
    Form_Main.StringGrid_SC01.Cells[1,i]:=TimeToStr(d_temp);//Время
    //Наименование товара (изначально пишем ИД)
    Form_Main.StringGrid_SC01.Cells[2,i]:=Form_Main.ADOQuery1.FieldByName('Cargo_Id').AsString;
    //Вес нормального и плохого ящика,г

    //Вес полный,г
    Form_Main.StringGrid_SC01.Cells[4,i]:=Form_Main.ADOQuery1.FieldByName('Massa').AsString;
    //Вес пустой паллеты,г

    //Вес перестила,г

    //Дата наряда
    d_temp:=Form_Main.ADOQuery1.FieldByName('TaskDate').AsDateTime;
    Form_Main.StringGrid_SC01.Cells[7,i]:=DateToStr(d_temp);
    //Номер смены
    Form_Main.StringGrid_SC01.Cells[8,i]:=Form_Main.ADOQuery1.FieldByName('WorkShift').AsString;
    //Кол-во ящиков

    Form_Main.ADOQuery1.Next;
    end;
end;

procedure TForm_Main.Button_ShowAllPallettes01Click(Sender: TObject);
begin
  if Form_Main.Button_ShowAllPallettes01.Caption='Показать все паллеты' then begin
     Form_Main.Button_ShowAllPallettes01.Caption:='Показать текущую';
     ShowAllPallettes;
     end
     else begin
     Form_Main.Button_ShowAllPallettes01.Caption:='Показать все паллеты';
     ShowWInTable;
     end;
end;

procedure TForm_Main.StringGrid_SC01DrawCell(Sender: TObject; ACol,
  ARow: Integer; Rect: TRect; State: TGridDrawState);
label m1;
var
```

```
  i,nr,ne:integer;
begin
  nr:=length(RedRows);
  ne:=length(YellowRows);
  if nr=0 then goto m1;
  Form_Main.StringGrid_SC01.Canvas.Brush.Color:=clRed;
  for i:=0 to nr-1 do
    if (ARow=RedRows[i]) then begin
      Form_Main.StringGrid_SC01.Canvas.FillRect(Rect);
      Form_Main.StringGrid_SC01.Canvas.TextRect(Rect,    Rect.Left,    Rect.Top,    Form_Main.StringGrid_SC01.Cells[aCol,
aRow]);
      end;
m1:
  if ne=0 then Exit;
  Form_Main.StringGrid_SC01.Canvas.Brush.Color:=clYellow;
  for i:=0 to ne-1 do
    if (ARow=YellowRows[i]) then begin
      Form_Main.StringGrid_SC01.Canvas.FillRect(Rect);
      Form_Main.StringGrid_SC01.Canvas.Font.Color:=clBlack;
      Form_Main.StringGrid_SC01.Canvas.TextRect(Rect,    Rect.Left,    Rect.Top,    Form_Main.StringGrid_SC01.Cells[aCol,
aRow]);
      end;
end;

procedure TForm_Main.N_CreateFileClick(Sender: TObject);
begin
  Form_CreateFile.Show;
end;

procedure TForm_Main.BitBtn_Pause01Click(Sender: TObject);
begin
  if Paused then begin
    Paused:=false;
    Form_Main.BitBtn_Pause01.Glyph.LoadFromFile(MyDir+'\images\pause.bmp');
    end
    else begin
    Paused:=true;
    Form_Main.BitBtn_Pause01.Glyph.LoadFromFile(MyDir+'\images\restore.bmp');
    end;
end;

procedure TForm_Main.N_CreateForLstPltClick(Sender: TObject);
var
  s_temp,s_temp2,s_temp3,s_temp4:string;
begin
  Form_CreateForPalette.Show;
  //Plt_ID:integer;//ИД паллеты
  s_temp2:=IntToStr(Plt_ID);
  //процедура создания или добавления в конец файла данных о последней паллете
  //CreateForLstPlt(scid,Plt_ID,outfname:string;addto:boolean);
end;

procedure TForm_Main.N_DataTransferClick(Sender: TObject);
begin
  Form_Developer.Show;
end;

procedure TForm_Main.N_DB_FilesClick(Sender: TObject);
begin
  Form_DBnFiles.show;
end;


procedure TForm_Main.N_EnteranceLogClick(Sender: TObject);
begin
```

```
   Form_EnteranceLog.Show;
end;

procedure TForm_Main.CheckBox_EmulationClick(Sender: TObject);
begin
  if Form_Main.CheckBox_Emulation.Checked then Form_Emulation.Show;
end;

end.
```

ДОДАТОК Д

ПРЕЗЕНТАЦІЯ