МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ   
РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение высшего образования «Самарский национальный исследовательский университет имени академика С.П. Королева (Самарский университет)»

Институт информатики и кибернетики

Кафедра программных систем

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**  
  
 к курсовой работе по дисциплине

«Разработка человеко-машинного интерфейса

в системах реального времени»

по теме «Система моделирования процесса регистрации продукции»

Вариант №11

Обучающийся 19.09.2022 В.А. Артамонов

Обучающийся 19.09.2022 Д.А. Смирнов

Руководитель 19.09.2022 Л.С. Зеленко

Самара 2022

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ   
РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение высшего образования «Самарский национальный исследовательский университет имени академика С.П. Королева (Самарский университет)»

Институт информатики и кибернетики

Кафедра программных систем

ЗАДАНИЕ

на курсовую работу по дисциплине

«Разработка человеко-машинного интерфейса   
в системах реального времени»

обучающимся в группе № 6132-020402D

В.А. Артамонову

Д.А. Смирнову

1. Тема проекта:«Система моделирования процесса регистрации продукции»
2. Задание**:** с помощью SCADA-системы WinCC OA 3.17 разработать программное обеспечение, моделирующее поведение работы конвейера, предусмотреть работу конвейера при возникновении нештатных ситуаций
3. Исходные данные к проекту**:** см. приложение к заданию
4. Перечень вопросов, подлежащих разработке:
   1. Произвести анализ предметной области: изучить основные принципы работы конвейера, состав оборудования и его характеристики, а также поведение оборудования в режиме реального времени
   2. Разработать информационную модель реального объекта, определить точки данных, команды управления, аварийные ситуации
   3. Разработать прототипы экранных форм (мнемосхем)
   4. Разработать алгоритмы управления конвейера
   5. Реализовать скрипты для элементов управления, провести тестирование и отладку
   6. Оформить документацию курсовой работы
   7. Подготовить презентацию по разработанной системе
5. Перечень графических разработок:
   1. Мнемосхема системы
   2. Диаграмма вариантов использования
6. Календарный план выполнения работ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № п/п | Содержание работы по этапам | Объем этапа в % к общему объему проекта | Срок  окончания | Фактическое выполнение |
| 1 | Оформление технического задания и его утверждение | 5 | 25.02.2022 |  |
| 2 | Описание и анализ предметной области | 15 | 04.03.2022 |  |
| 3 | Проектирование системы | 25 |  |  |
| 3.1 | Разработка информационной модели объекта | 10 | 11.03.2022 |  |
| 3.2 | Разработка прототипа интерфейса пользователя | 10 | 25.03.2022 |  |
| 3.3 | Разработка диаграммы вариантов использования | 5 | 25.03.2022 |  |
| 4 | Реализация проекта, разработка контрольных примеров. Предъявление реализации руководителю | 45 | 13.05.2022 |  |
| 5 | Корректировка проекта и оформление документации проекта. Защита проекта с представлением презентации. | 10 | 20.05.2022 |  |

Задание принял  
 к исполнению 15.02.2022 В.А. Артамонов

15.02.2022 Д.А. Смирнов

ПРИЛОЖЕНИЕ  
к заданию на курсовую работу  
обучающимся в группе № 6132-020402D

В.А. Артамонову

Д.А. Смирнову

Тема проекта: «Система моделирования процесса регистрации продукции»

Исходные данные к проекту:

1. Характеристика объекта автоматизации:

## объект автоматизации: конвейер;

## виды автоматизируемой деятельности:

* + процесс авторизации пользователей системы;
  + процесс моделирования работы оборудования конвейера;
  + процесс управления оборудованием конвейера при возникновении нештатных ситуаций;
  + процесс визуализации работы оборудования конвейера;
  + процесс формирования исторических значений;
  + процесс обработки аварийных сообщений;

## количество ролей пользователей – 2 (администратор и оператор);

## минимальная длина пароля – 4 (символа);

## максимальная длина пароля – 16 (символа);

## минимальная длина логина – 4 (символа);

## максимальная длина логина – 16 (символа);

## количество режимов работы оборудования – 2;

## количество типов потока – 2;

## количество законов распределения для моделирования потока изделий – 1;

## минимальное значение интенсивности поступления изделий – 5 (шт/мин);

## максимальное значение интенсивности поступления изделий – 25 (шт/мин);

## количество типов изделий – 3;

## количество элементов оборудования – 5;

## количество конвейерных сканеров – 1;

## количество несущих лент – 1;

## ширина несущей ленты – 600 (мм);

## длина несущей ленты – 10 (м);

## минимальное значение скорости движения ленты – 1,0 (м/с);

## максимальное значение скорости движения ленты – 2,0 (м/с);

## количество конвейерных сталкивателей – 3;

## количество транспортных тележек – 3;

## максимальное количество изделий первого типа, вмещаемых в транспортную тележку, – 200 (шт);

## максимальное количество изделий второго типа, вмещаемых в транспортную тележку, – 400 (шт);

## максимальное количество изделий третьего типа, вмещаемых в транспортную тележку, – 600 (шт);

## количество приводных устройств – 1;

## мощность приводного устройства – 0,5 – 2,2 (кВт).

1. Требования к информационному обеспечению:
2. информационное обеспечение разрабатывается на основе следующих источников:
   * описание принципов работы конвейера [Электронный ресурс]. URL: https://tehpribory.ru/glavnaia/oborudovanie/konvejer.html (дата обращения: 17.02.2022);
   * конвейерные сканеры [Электронный ресурс]. URL: <https://id-russia.ru/hardware/conveyor-scanners> (дата обращения: 17.02.2022);
   * ленточный конвейер [Электронный ресурс]. URL: <https://www.isuct.ru/sites/default/files/department/ightu/ktmio/36.pdf> (дата обращения: 17.02.2022);
3. структуры точек данных определяются в процессе проектирования;
4. структура мнемосхемы определяется в процессе проектирования.
5. Требования к техническому обеспечению:
6. тип ЭВМ – IBM PC совместимый;
7. монитор с разрешающей способностью не ниже 800 х 600;
8. манипулятор – мышь;
9. технические характеристики определяются в процессе выполнения проекта.
10. Требования к программному обеспечению:
11. тип операционной системы – Windows 7 и выше;
12. SCADA-система – WinCC OA 3.17;
13. язык программирования – CTRL;
14. среда программирования – Vision;
15. среда проектирования – StarUML 4.1.6.
16. Общие требования к проектируемой системе:

5.1 Функции, реализуемые системой:

1. общесистемные функции:
   * авторизация пользователя в системе (ввод логина/пароля);
   * аутентификация пользователя в системе, настройка интерфейса пользователя на заданную роль;
   * моделирование работы оборудования в автоматическом режиме:
   * генерирование изделий разного типа в соответствии с заданными параметрами;
   * перемещение изделия по транспортной ленте;
   * сканирование изделия и определение его типа;
   * выработка управляющего сигнала от сканера к толкателям;
   * сталкивание изделия в транспортную тележку;
   * контроль наполнения транспортной тележки;
   * моделирование возникновения нештатных ситуаций (отказов) в режиме реального времени и выдача аварийных сообщений;
   * визуализация процесса штатной работы оборудования и возникновения нештатных ситуаций:
   * отображение аварийных сообщений (алармов);
   * отображение состояния оборудования;
   * отображение параметров оборудования;
2. функции администратора:
   * регистрация в системе (ввод логина и пароля);
   * настройка параметров оборудования (точек данных):
   * задать мощность приводного устройства;
   * выбор режима моделирования для потока изделий;
   * задание интенсивности потока изделий;
   * работа с оборудованием при возникновении нештатных ситуаций (ручной режим):
   * включение стопорных механизмов для сталкивателей;
   * остановка конвейерной ленты;
   * запуск конвейерной ленты;
   * отправка конвейерной ленты в ремонт;
   * отправка приводного устройства в ремонт;
   * остановка конвейерного сканера;
   * запуск конвейерного сканера штрих-кода;
   * отправка сканера штрих-кода в ремонт;
   * остановка конвейерного сталкивателя;
   * запуск конвейерного сталкивателя;
   * отправка сталкиватель в ремонт;
   * квитирование аварийных сообщений;
   * выбор параметров для просмотра исторических данных;
   * просмотр архивных параметров за требуемый интервал времени;
3. функции оператора:
   * регистрация в системе (ввод логина и пароля);
   * работа с оборудованием при возникновении нештатных ситуаций (ручной режим):
   * остановка приводного устройства;
   * запуск приводного устройства;
   * остановка конвейерного сканера штрих-кода;
   * запуск конвейерного сканера штрих-кода;
   * остановка конвейерного сталкивателя;
   * запуск конвейерного сталкивателя;
   * квитирование аварийных сообщений;
   * выбор параметров для просмотра исторических данных;
   * просмотр архивных параметров за требуемый интервал времени.

5.2 Технические требования к системе:

1. режим работы ‑ диалоговый;
2. система должна удовлетворять санитарным правилам и нормам  
    СанПин 2.2.2./2.4.2198-07;
3. условия работы средств вычислительной техники (содержание вредных веществ, пыли и подвижность воздуха) должны соответствовать ГОСТ 12.1.005, 12.01.007;
4. температура окружающего воздуха – 15-35°С;
5. влажность воздуха – 45-75%.

Руководитель   
проекта 23.03.2022 Л.С. Зеленко

Задание принял  
к исполнению 18.02.2022 В.А. Артамонов

18.02.2022 Д.А. Смирнов

РЕФЕРАТ

Пояснительная записка 100 с, 56 рисунков, 7 таблиц, 11 источников,  
2 приложения.

КОНВЕЙЕР, SCADA-СИСТЕМЫ, ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ, НЕСУЩАЯ ЛЕНТА, СИСТЕМА МОДЕЛИРОВАНИЯ, КОНВЕЙЕРНЫЙ СКАНЕР, РЕГИСТРАЦИЯ ПРОДУКЦИИ, КОНВЕЙЕРНЫЕ СТАЛКИВАТЕЛИ.

Объектом автоматизации является конвейер.

Во время курсового проектирования разработана система моделирования работы конвейер с помощью SCADA-системы WinCC OA 3.17. В системе реализованы две роли пользователей: администратор и оператор. Оператор наблюдает за ходом процесса и при возникновении аварийных ситуаций включается в процесс работы оборудования. Администратор дополнительно может задать параметры оборудования и задавать поток интенсивности изделий перед началом технологического процесса.

В системе имеется возможность смоделировать поведение работы конвейера и предусмотреть работу конвейера при возникновении нештатных ситуаций.

Программное обеспечение разработано на языке CTRL в среде WinCC OA 3.17 и функционирует под управлением операционной системы Windows 7 и выше.

СОДЕРЖАНИЕ

[Введение 12](#_Toc114468411)

[1 Описание предметной области 14](#_Toc114468412)

[1.1 Основные понятия и определения 14](#_Toc114468413)

[1.1.1 Виды конвейеров по способу перемещения 14](#_Toc114468414)

[1.1.2 Типы конвейеров 15](#_Toc114468415)

[1.1.3 Конвейерные сканеры 20](#_Toc114468416)

[1.1.4 Конвейерные сталкиватели 22](#_Toc114468417)

[1.1.5 Транспортные тележки 23](#_Toc114468418)

[1.2 Описание процесса функционирования системы 24](#_Toc114468419)

[1.3 Описание оборудования, задействованного в работе 24](#_Toc114468420)

[1.3.1 Описание конвейерного сканера штрих-кода Datalogic Axiom 400 25](#_Toc114468421)

[1.3.2 Описание горизонтальный ленточный конвейер КЛГ-10-600-90-2,2-1100 26](#_Toc114468422)

[1.3.3 Описание конвейерного сталкивателя 28](#_Toc114468423)

[1.3.4 Описание транспортировочной тележки 30](#_Toc114468424)

[1.4 Постановка задачи 31](#_Toc114468425)

[2 Проектирование системы 35](#_Toc114468426)

[2.1 Разработка информационной модели объекта 35](#_Toc114468427)

[2.1.1 Описание точек данных (Data Points) 35](#_Toc114468428)

[2.2 Проектирование интерфейса пользователя 40](#_Toc114468429)

[2.3 Диаграмма вариантов использования 44](#_Toc114468430)

[3 Реализация системы 49](#_Toc114468431)

[3.1 Настройка элементов точек данных 49](#_Toc114468432)

[3.2 Настройка исходных значений 50](#_Toc114468433)

[3.3 Общие настройки объекта 50](#_Toc114468434)

[3.4 Обработка сообщений (alert) 51](#_Toc114468435)

[3.5 Разработка и описание графического интерфейса 51](#_Toc114468436)

[3.5.1 Описание графических элементов мнемосхемы 52](#_Toc114468437)

[3.5.2 Начало работы 54](#_Toc114468438)

[3.5.3 Режим администратора 56](#_Toc114468439)

[3.5.4 Режим оператора 58](#_Toc114468440)

[3.5.5 Обработка внештатных ситуаций 58](#_Toc114468441)

[3.5.6 Просмотр параметров оборудования 60](#_Toc114468442)

[Заключение 62](#_Toc114468443)

[Список использованных источников 63](#_Toc114468444)

[Приложение А Описание точек данных 65](#_Toc114468445)

[Приложение Б Сценарии (скрипты управления) 68](#_Toc114468446)

[Приложение В Руководство пользователя 95](#_Toc114468447)

ВВЕДЕНИЕ

SCADA (аббр. от англ. supervisory control and data acquisition, диспетчерское управление и сбор данных) – программный пакет, предназначенный для разработки или обеспечения работы в реальном времени систем сбора, обработки, отображения и архивирования информации об объекте (мониторинг) или управления им [1].

Диспетчерское управление и сбор данных является основным и в настоящее время остается наиболее перспективным методом автоматизированного управления сложными динамическими системами (процессами) в жизненно важных и критичных с точки зрения безопасности и надежности областях. Именно на принципах диспетчерского управления строятся крупные автоматизированные системы в промышленности и энергетике, на транспорте, в космической и военной областях, в различных государственных структурах [1].

SCADA-системы решают следующие задачи:

* Обмен данными с «устройствами связи с объектом», (то есть с промышленными контроллерами и платами ввода/вывода) в реальном времени через драйверы.
* Обработка информации в реальном времени.
* Логическое управление.
* Отображение информации на экране монитора в удобной и понятной для человека форме.
* Ведение базы данных реального времени с технологической информацией.
* Аварийная сигнализация и управление тревожными сообщениями.
* Подготовка и генерирование отчетов о ходе технологического процесса.
* Осуществление сетевого взаимодействия между SCADA ПК.
* Обеспечение связи с внешними приложениями (СУБД, электронные таблицы, текстовые процессоры и т. д.). В системе управления предприятием такими приложениями чаще всего являются приложения, относимые к уровню MES (от англ. manufacturing execution system, система управления производственными процессами).

Таким образом, SCADA-системы используются во всех отраслях хозяйства, где требуется обеспечивать операторский контроль за технологическими процессами в реальном времени.

Во время курсового проектирования необходимо разработать систему моделирования работы конвейера, с помощью которой можно будет регистрировать изделия разного типа.

Система будет разрабатываться с помощью SCADA-системы SIMATIC WinCC OA 3.17, которая разработана для применения в приложениях, требующих гибкой и адаптивной платформы для решения индивидуальных задач клиента, а также в больших или сложных проектах, которые предъявляют специфические требования к функциональности и архитектуре системы.

При проектировании системы будет использоваться язык моделирования UML (Unified Modeling Language), который является стандартным инструментом для разработки «чертежей» программного обеспечения [2].

1. Описание предметной области
   1. Основные понятия и определения

Конвейер – специализированная производственная машина для непрерывной транспортировки объектов. В зависимости от специализации может перемещать сыпучие, кусковые грузы или штучные изделия. Является частью поточного производства [3].

Рассмотрим основные виды конвейеров в зависимости от различных признаков классификации.

* + 1. Виды конвейеров по способу перемещения

Конструктивные особенности конвейеров позволяют их использовать для транспортировки грузов в различном направлении. Они бывают [3]:

* Вертикальные.
* Горизонтальные.
* Наклонные.

Вертикальные конвейеры поднимают груз вверх или опускают вниз. Такой способ перемещения является наиболее удачным в плане эффективности использования производственного пространства. Машина перемещает грузы между оборудованием, размещенным на разных уровнях. Кроме этого она может применяться в качестве погрузчика, для выгрузки в бункер, контейнер железнодорожного вагона, кузов грузовика и т.п.

Горизонтальные конвейеры перемещают груз вдоль производственной линии. Они зачастую передвигают его в морозильных камерах, туннельных печах и т.д.

Наклонные поднимают или опускают груз под углом. По сути это гибрид горизонтальной и вертикальной машины. Устройства этого типа могут использовать для загрузки, поднятия сырья на уровень выше и решения прочих производственных задач [3].

* + 1. Типы конвейеров

Конвейер может использоваться для транспортировки грузов разного размера и степени прочности. Для каждого из них требуется специфическая машина, способная обеспечить сохранность переносимого изделия. Для сыпучих изделий нужны одни конвейеры, для штучных совсем другие. В связи с этим на производствах применяются машины разного типа.

Самые популярные из них следующие:

* Ленточный.
* Шнековый или винтовой.
* Нория.
* Пластинчатый.
* Роликовый.
* Вибрационный.

1. Ленточный конвейер – это наиболее распространенный тип транспортировочных производственных машин. Они являются универсальными. Могут работать с любыми типами грузов, но чаще всего применяются со штучными изделиями (рисунок 1).

  
Рисунок 1 – Пример ленточного транспортера



В таблице 1 описаны технические характеристики ленточных транспортеров.

Таблица 1 – Технические характеристики ленточных транспортеров

|  |  |  |
| --- | --- | --- |
| Наименование показателей | КЛГ-10-600-90-2,2-1100 | ЛТ-4000 |
| Длина, мм | 10000 | 10000 |
| Ширина, мм | 600 | 500 |
| Скорость движения ленты, м/с | 1,0 – 2,0 | 0,5 – 1,5 |
| Мощность привода, кВт | 0,5 – 2,2 | 0,5 – 2,0 |

Устройство представляет собой закольцованную широкую ленту. Она натягивается между ведущим и ведомыми роликами. За счет вращения ведущего лента перекатывается. Подобная конструкция используется на кассах супермаркетов. Лента может двигать грузы горизонтально, наклонно или вертикально. Также бывают конвейеры, двигающиеся по кругу, или под разными углами. Усложненные устройства состоят из нескольких лент. Для вертикального поднятия груза лента конвейера оснащается зацепами. Преимущество таких машин в бережной транспортировке.

Ленточные конвейеры могут двигать груз на различную дистанцию. На больших производствах встречаются машины длиной 300 м. Применяемая для этого лента делается из резины или обрезиненной ткани. Она прочная и долговечная [3].

1. Шнековые конвейеры состоят из шнека, расположенного в корпусе в виде трубы. Сам шнек представляет собой стержень с винтовой поверхностью по всей длине. Его привод осуществляется с помощью шкива, связанного с электродвигателем, чаще всего посредством ременной передачи (рисунок 2).

Такие конвейеры часто называют погрузчиками. Они предназначены для транспортировки сыпучих грузов. Шнек, вращаясь, затягивает сырье и перемещается его внутри корпуса. Сам груз попадает в него за счет гравитации. Шнеками выполняется перемещение вертикально вверх до 30 м, под наклоном, а также горизонтально до 40 м.

   
Рисунок 2 – Примеры шнековых транспортеров

Данное оборудование является очень продуктивным, надежным, недорогим. При этом оно имеет и важный недостаток – повреждает груз. Перемещаемое винтом сырье может сильно сжиматься и перетираться. В связи с этим шнековые машины лучше не применять для транспортировки посевного зерна [3].

1. Нория – это ковшевой конвейер, также его называют элеватор. Нория предназначена для перемещения жидкостей и сыпучих грузов вертикально вверх. Она имеет подобное строение с ленточными конвейерами. Между ее барабанами натянута лента или роликовые цепи. На них закрепляются ковши. По мере вращения они зачерпывают жидкость или сыпучий груз, подымают его вверх и опрокидывают (рисунок 3).

  
Рисунок 3 – Пример транспортера нория

Этот тип конвейера очень популярный, так как совершенно не повреждает перемещаемый груз. При этом устройство отличается высокой производительностью за счет величины и количества ковшей. Скорость его движения может составлять больше чем 4 метра в секунду. Крупные нории подымают до 500 м3 грузов в час [3].

1. Пластинчатый транспортер, практически полностью копирующий конструкцию ленточного конвейера, но используемое в качестве тягового органа соединенные в большое кольцо металлические или пластиковые пластины. Они скрепляются между собой на шарниры, что позволяет им поворачивать относительно друг друга (рисунок 4).

  
Рисунок 4 – Пример пластинчатого транспортера

Пластины более крепкие, чем ленты. Они износоустойчивые и могут перевозить в разы более тяжелые грузы. Пластины могут быть гладкими или оснащаться скребками. Первые применяются для горизонтального перемещения, а последние для наклонного под углом до 70°.

Конвейер пластинчатого типа имеет производительность до 2 тыс. тонн в час. Так как они предназначены для перемещения тяжестей, то толщина пластин составляет минимум 3 мм, что делает такое оборудование очень тяжелым, к тому же дорогостоящим. Пластины не могут быстро переворачиваться, поэтому скорость такой машины составляет максимум 1,25 м/сек [3].

1. Роликовый конвейер для горизонтального перемещения, также известный как рольганг. Он представляет собой подобие уложенной садовой лестницы, вместо перекладин в которой применяются широкие ролики. Они проворачиваются, за счет чего сдвигают опираемый на них сверху груз. Они применяются для перемещения штучных изделий, а также различной тары. Ролики ставят на лесопилках, предприятиях, занимающихся производством металлопроката, железобетонных плит и столбов, а также в цехах упаковки (рисунок 5).

  
Рисунок 5 – Пример роликового конвейера

Рольганги бывают гравитационные и приводные. Ролики последних оснащаются приводом, что позволяет перемещать груз без прикладывания усилий. Гравитационные машины имеют свободные ролики. То есть, чтобы груз по ним проехал, его нужно толкать. Это делается вручную, или он съезжает под уклоном просто за счет гравитации [3].

1. Вибрационный транспортер обеспечивает транспортировку мелкого сыпучего груза и кускового материала путем воздействия вибрацией на рабочую платформу желоба. За счет колебаний осуществляется подпрыгивание груза. Данное оборудование преимущественно применяются для поднятия взорванной породы. То есть устройство практически полностью находиться под слоем сыпучего груза и постепенно его перемещает. За счет давящей массы другие типы конвейеров с данной задачей не справляются без вспомогательного оборудования для погрузки, тем более при столь слабом приводе (рисунок 6).

  
Рисунок 6 – Пример вибрационного транспортера

Несмотря на кажущуюся малую продуктивность такого способа перемещения, вибрационный конвейер способен транспортировать до 450 т груза в час. Однако длина его платформы редко бывает больше 9-10 м [3].

В курсовой работе будет использоваться транспортер КЛГ-10-600-90-2,2-1100 ленточного типа.

* + 1. Конвейерные сканеры

Конвейерные сканеры – это специальные компактные устройства, которые позволяют организовать считывание штрих-кода без участия человека при перемещении товаров, продукции, запасных частей на складе или производстве по конвейерной ленте (рисунок 7). Такие устройства «видят» штрих-коды на продукции, считывают его и передают информацию в систему автоматизации [4].

  
Рисунок 7 – Пример конвейерных сканеров 2D штрих-кода

В таблице 2 описаны технические характеристики конвейерных сканеров 2D штрих-кодов

Таблица 2 – Технические характеристики конвейерных сканеров 2D штрих-кодов

|  |  |  |
| --- | --- | --- |
| Наименование показателей | Datalogic ACCULAZR AL5010 | Datalogic Axiom 400 |
| Скорость сканирования, s/p | 1200 | 1400 |

Примеры применения конвейерных сканеров:

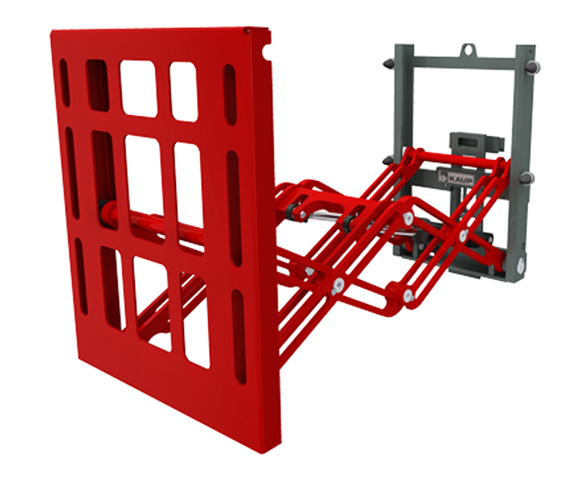
* Валидация штрих-кода. Считывание штрихового кода (уникальный код или артикул) с одиночного изделия или групповой упаковки для контроля правильности сортировки потока изделий.
* Агрегация. Считывание уникальных кодов с группы изделий с целью определения состава группы и последующей привязки этих кодов к коду групповой упаковки.
* Учет выпуска готовой продукции. Сканирование уникальных кодов или артикулов продукции на участке автоматической отгрузки.
* Сериализация. Считывание уникального штрихового кода с одиночного изделия с целью построения виртуальной очереди изделий в системе.
* Учет перемещений внутри предприятия. Считывание уникальных кодов продукции для организации прослеживаемости её перемещений между цехами или складами.
* Контроль маркировки. Контроль читаемости кода и соответствия маркировки продукции. Возможен контроль отдельных параметров качества печати штриховых кодов и выбраковка подозрительных образцов для дальнейшей проверки с помощью верификатора.

В курсовой работе будет использоваться конвейерный сканер Datalogic Axiom 400.

* + 1. Конвейерные сталкиватели

Сортировочные сталкиватели встраиваются в определенных местах в роликовый или ленточный конвейер (рисунок 8). Сталкиватели могут быть простыми и портальными – в последнем варианте возможна двунаправленная сортировка. Сталкиватели приводятся в движение с помощью пневматических цилиндров. Сортировка производится под углом 90 градусов к основному потоку груза на выходные конвейеры или в лотки.

  
Рисунок 8 – Пример конвейерных сталкивателей



Ограничения по размеру продукции, которая может обрабатываться на сортировщике, составляет 10 мм в высоту, максимальный вес единицы продукции тоже ограничен [5].

В курсовой работе будет использоваться пневматический конвейерный сталкиватель СПК.

* + 1. Транспортные тележки

Тележка транспортер – это устройство, которое устанавливается на горизонтальных ленточных конвейерах и служит для организации операций по разгрузке системы (рисунок 9). В зависимости от особенностей определенного устройства подготавливается и конструкция тележки. Например, они могут служить для работы с зерном, комбикормом, углем, строительным камнем и пр. материалами [8].

  
Рисунок 9 – Пример транспортных тележек



Устройства, предназначенные для организации разгрузочных операций, могут характеризоваться различными габаритами и размерами. Механизм передвижения тележки у каждого отдельного приспособления имеет свой уровень мощности, что нужно учитывать при выборе конструкции.

В настоящее время можно найти оборудование подобного типа как отечественного производства, так и, соответственно, иностранного. Естественно, что при поиске разгрузочной тележки следует в первую очередь обращать внимание на особенности имеющегося конвейера, учитывать параметры установленной ленты и т. д. Только так можно подобрать самый лучший вариант механизма, который будет способен справиться с поставленными задачами [8].

В таблице 3 описаны технические характеристики транспортных тележек.

Таблица 3 ‒ Технические характеристики транспортной тележки

|  |  |  |
| --- | --- | --- |
| Наименование показателей | ТП-4 | КПО-300-С |
| Высота платформы, мм | 1200 | 1000 |
| Ширина платформы, мм | 600 | 600 |
| Грузоподъёмность, кг | 450 | 400 |

В курсовой работе будет использоваться транспортная тележка ТП-4.

* 1. Описание процесса функционирования системы

Система сортировки упаковочных бумажных пакетов начинается с того, что на конвейерную ленту подаются коробки габаритами 40\*20\*50 см, которые необходимо рассортировать на маленькие, средние и большие пакеты, на каждой коробке присутствует штрих-код, который несёт в себе информацию о производителе, дате изготовления и тип продукции и ее количества. Проходя через сканер, с коробок считывается штрих-код информация о котором поступает на компьютер, где сигнал со сканера обрабатывается и передается сталкивателям, которые в нужный момент сталкивают короб с конвейера в приемную транспортную тележку.

* 1. Описание оборудования, задействованного в работе

В работе предприятия будут задействованы ниже перечисленные оборудования:

1. конвейерный сканер штрих-кода Datalogic Axiom 400;
2. горизонтальный ленточный конвейер КЛГ-10-600-90-2,2-1100;
3. сталкиватель пневматический конвейерный (СПК);
4. тележка-транспортер ТП-4.
   * 1. Описание конвейерного сканера штрих-кода Datalogic Axiom 400

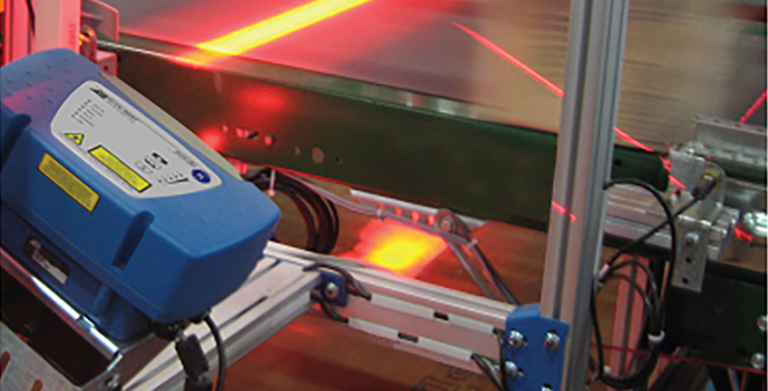
В данной работе будет использоваться конвейерный сканер Datalogic Axiom 400 (рисунок 10).

Cканер Datalogic Axiom 400, разработан с учетом особенностей промышленных и производственных областей. AXIOM 400 - это сканер с высокой скоростью считывания и с широкой глубиной поля считывания, позволяющий точное распознавание штрих кода с экстремальных дистанций. Компактная модульная конструкция содержит четыре лазерных диода разной мощности для высокопроизводительного сканирования штрих кодов разного размера на скоростных конвейерах. Несмотря на свою технологическую и функциональную насыщенность, сканер весьма прост в установке и обслуживании.

Любое оптическое устройство имеет свою зону оптимального фокусного расстояния на которой в полной мере проявляются его характеристики. Встроенные в AXIOM 400 четыре лазера с разными фокусными зонами позволяют сканеру осуществлять автофокусировку, переключаясь между зонами наилучшего фокуса, и таким образом выбирать зону сканирования в зависимости от расположения коробки или палеты по ширине конвейера. Таким образом автоматически перекрываются очень большие участки расстояний, сканер практически не имеет мертвых зон в диапазоне расстояний от 28 до 216 см.

Типовые возможности применения:

* Сканирование этикеток с коробок и упаковок, движущихся по конвейерной ленте.
* Сортировка посылок.
* Считывание штрих кода с багажных бирок в аэропорту
* Автоматическое распознавание и трассировка упаковок с продукцией.

  
Рисунок 10 – Пример конвейерного сканера Datalogic Axiom 400

Особенности модели:

* Скорость считывания до 1400 сканирований в секунду.
* Защита по классу IP65.
* Программа "AXESS" обеспечивает удобную установку, настройку и диагностику в среде Windows.
* Обеспечивает соединение по Ethernet и промышленным протоколам Profibus, DeviceNet (опционально).
  + 1. Описание горизонтальный ленточный конвейер   
       КЛГ-10-600-90-2,2-1100

Ленточный конвейер (англ. belt conveyor) – транспортирующее устройство непрерывного действия с рабочим органом в виде ленты [6].

Ленточный конвейер является наиболее распространённым типом транспортирующих машин, он служит для перемещения насыпных или штучных грузов. Применяется на промышленных производствах, в рудниках и шахтах, в сельском хозяйстве. В зависимости от свойств и природы перемещаемого груза угол наклона рабочей стороны ленты может быть установлен до 90°.

Часто конвейерная лента является одной из частей транспортирующего устройства. Например, зернопогрузчик, применяющийся на механизированном току для сбора зерновой массы с площадки, имеет щёточные скребки, далее зерно поднимается норией и попадает на ленточный конвейер, который забрасывает зерно в кузов грузового автомобиля.

Ленточные конвейеры бывают передвижными, переносными, поворотными и стационарными. Стационарные машины применяют для перемещения большого количества материалов на расстояние от 3 до 300[источник не указан 3219 дней] м., а передвижные и переносные машины — для перемещения небольшого количества материала на расстояние от 2 до 20 м. В практике применяют последовательно расположенные конвейеры для перемещения материала на десятки километров.

Ленточные конвейеры различаются по видам. Виды ленточных конвейеров:

* По типу трассы:

1. горизонтальные ленточные конвейеры;
2. наклонные ленточные конвейеры;
3. крутонаклонные ленточные конвейера;
4. с изменяющимся углом наклона;
5. Z – образные ленточные конвейера;
6. L – образные;
7. V – образные.

* По типу несущей поверхности:

1. с прямой гладкой поверхностью ленты;
2. желобчатые ленточные конвейеры и транспортеры;
3. с перегородками (поперечинами) на ленте;
4. с гофробортам (бортиками) на ленте;
5. с модульной лентой модульные конвейеры.

Ленточные конвейеры состоят из основных узлов: приводной барабан, привод конвейера (мотор-редуктор), натяжной барабан, узел натяжения, несущая ленточная часть, опорные и поддерживающие ролики или поддерживающий листовой стол и, собственно, рама конвейера, изготовленная из сваренного металлопроката. Конвейеры, в которых используется вместо ленты металлическое сетчатое полотно (сетка конвейерная), называются «сетчатый конвейер». Используется в основном в местах с высокими температурами груза, либо в жарких печах. Например, для выпекания основы для пиццы [6].

В данной работе будет использоваться горизонтальный ленточный конвейер КЛГ-10-600-90-2,2-1100 (рисунок 11).

Рисунок 11 – Пример горизонтального ленточного конвейера  
КЛГ-10-600-90-2,2-1100



* + 1. Описание конвейерного сталкивателя

Перемещение груза в конвейерной линии может осуществляться не только за счёт создания разветвлённой сети транспортировочных трасс, но и за счёт использования дополнительного оборудования для перераспределения грузопотоков. В качестве дополнительного оборудования для сортировки перемещаемых грузов на конвейерных линиях может быть установлен сталкиватель пневматический.

Сталкиватель пневматический конвейерный (рисунок 12) предназначается для двунаправленной сортировки перемещаемых грузов на конвейерной линии под углом 90 градусов с основного потока на выходные ленточные конвейеры, рольганги или в лотки.

  
Рисунок 12 – Пример пневматического конвейерного сталкивателя

Сталкиватель пневматический устанавливается на раме над транспортировочной трассой конвейера параллельно его продольной оси. Устройство имеет сталкивающую планку и связанный с приводом механизм плоскопараллельного перемещения сталкивающей планки в горизонтальной плоскости. Также в устройстве имеется пневматический стопор, которых способен приостановить перемещаемый груз, что при совместном использовании со сканером штрих-кодов позволяет автоматизировать процесс сортировки и распределения грузов по конвейерной линии склада или производства продукции. Ограничения по размеру продукции, которая может обрабатываться с помощью пневматического сталкивателя, составляет 10 мм в высоту при максимальном весе единицы продукции до 70 килограмм.

* + 1. Описание транспортировочной тележки

Тележка транспортер — это устройство, которое устанавливается на горизонтальных ленточных конвейерах и служит для организации операций по разгрузке системы.

Подобного рода механизмы монтируются на специальном треке, по которому они перемещаются благодаря рельсам. Собственно, сам трек одновременно с тем является еще и центральной частью всего конвейера, где имеют место быть соответствующие роликоопоры.

Разгрузочные тележки могут изготавливаться в нескольких вариантах, каждый из которых предполагает взаимодействие с определенными типами конвейеров. Так, в этом отношении особо значимыми становятся такие параметры, как уровень производительности, ширина транспортерной ленты и колеи самого устройства, общий вес. Разумеется, что каждый отдельный вариант пригоден для работы с соответствующими типами грузов и в конкретных условиях.

  
Рисунок 13 – Транспортная тележка ТП-04

* 1. Постановка задачи

Во время курсового проектирования необходимо разработать систему моделирования работы конвейера с помощью SCADA-системы WinCC OA 3.17, которая будет управлять процессом сортировки коробок с пакетами в заданных режимах и с помощью, которой оператор сможет наблюдать за данным процессом посредством мнемосхемы, отображающей текущее состояние оборудования на разных этапах.

В системе должно быть реализовано две роли пользователей: администратор и оператор конвейера.

Каждый пользователь должен авторизоваться в системе (ввести логин и пароль), система должна проверить ученые данные пользователей и настроить интерфейс пользователя на заданную роль.

В системе должно быть предусмотрено два режима работы: ручной и автоматический. В автоматическом режиме генерирование изделий разных типов в соответствии с заданными параметрами и параметры оборудования (мощность приводного устройства, скорость конвейерной ленты, а также количество изделий, находящихся в тележках) являются значениями по умолчанию. В ручном режиме администратор должен вводить данные параметры самостоятельно.

Режим администратора

Администратор отвечает за: настройку всех видов оборудования, настройку параметров потока изделий, поступающих на конвейерную ленту; включение и выключение оборудования; работу с архивными данными и переход в ручной режим при возникновении нештатных ситуаций.

Администратор задает такие параметры оборудования, как: скорость конвейерной ленты (1,0-2,0 м/c), мощность приводного устройства (0,5-2,2 кВт), тележка. Кроме этого, администратор может выбрать модель поступления изделий на конвейер: выбрать тип потока (детерминированный или случайный) и настроить параметры этого потока (интенсивность 10-40 шт/мин).

При отказе оборудования администратор должен принять решение о замене или отправке его в ремонт, а при возникновении аварийных ситуаций остановить или запустить оборудование. Администратор должен квитировать аварийные сообщения, просматривать исторические данные за требуемый промежуток времени.

Система должна моделировать возникновение отказов оборудования. Необходимо предусмотреть возможность экстренной остановки приводного устройства, работу сталкивателя, а также всей системы при возникновении аварийных ситуаций.

Режим оператора

В обязанности оператора входит наблюдение за исправной работой оборудования и при необходимости устрание возникших нештатных ситуаций с оборудованием (в ручном режиме). Также, оператор имеет возможность остановить или запустить работу оборудования. Оператор должен квитировать аварийные сообщения, просматривать исторические данные за требуемый промежуток времени.

Любой пользователь должен иметь возможность наблюдать за технологическим процессом, переходить в ручной режим при необходимости и работать архивными параметрами.

Также, в системе должна быть предусмотрена выдача справочной информация о системе и о разработчиках.

Таким образом, в системе должны быть реализованы следующие функции:

1. общесистемные функции:
   * авторизация пользователя в системе (ввод логина/пароля);
   * аутентификация пользователя в системе, настройка интерфейса пользователя на заданную роль;
   * моделирование работы оборудования в автоматическом режиме:
   * генерирование изделий разного типа в соответствии с заданными параметрами;
   * перемещение изделия по транспортной ленте;
   * сканирование изделия и определение его типа;
   * выработка управляющего сигнала от сканера к толкателям;
   * сталкивание изделия в транспортную тележку;
   * контроль наполнения транспортной тележки;
   * моделирование возникновения нештатных ситуаций (отказов) в режиме реального времени и выдача аварийных сообщений;
   * визуализация процесса штатной работы оборудования и возникновения нештатных ситуаций:
   * отображение аварийных сообщений (алармов);
   * отображение состояния оборудования;
   * отображение параметров оборудования;
2. функции администратора:
   * регистрация в системе (ввод логина и пароля);
   * настройка параметров оборудования (точек данных):
   * задать скорости движения ленты;
   * задать мощность приводного устройства;
   * выбор режима моделирования для потока изделий;
   * задание интенсивности потока изделий;
   * работа с оборудованием при возникновении нештатных ситуаций (ручной режим):
   * включение стопорных механизмов для сталкивателей;
   * остановка конвейерной ленты;
   * запуск конвейерной ленты;
   * отправка конвейерной ленты в ремонт;
   * отправка приводного устройства в ремонт;
   * остановка конвейерного сканера;
   * запуск конвейерного сканера штрих-кода;
   * отправка сканера штрих-кода в ремонт;
   * остановка конвейерного сталкивателя;
   * запуск конвейерного сталкивателя;
   * отправка сталкиватель в ремонт;
   * квитирование аварийных сообщений;
   * выбор параметров для просмотра исторических данных;
   * просмотр архивных параметров за требуемый интервал времени;
3. функции оператора:
   * регистрация в системе (ввод логина и пароля);
   * работа с оборудованием при возникновении нештатных ситуаций (ручной режим):
   * остановка приводного устройства;
   * запуск приводного устройства;
   * остановка конвейерного сканера штрих-кода;
   * запуск конвейерного сканера штрих-кода;
   * остановка конвейерного сталкивателя;
   * запуск конвейерного сталкивателя;
   * квитирование аварийных сообщений;
   * выбор параметров для просмотра исторических данных;
   * просмотр архивных параметров за требуемый интервал времени.
4. Проектирование системы
   1. Разработка информационной модели объекта

Отправной точкой при создании проекта в WinCC OA является разработка информационной модели будущей системы, включающей иерархию и структуры данных, алармов и их свойств. Основой для формирования информационной модели может служить (в зависимости от назначения и масштаба системы) техническая структура объектов, включенных в SCADA-систему (аналогия «элемент – устройство – агрегат – …»), организационная модель объекта, например, иерархия диспетчерских пунктов (местный уровень – региональный уровень – центральный уровень), или их сочетание. Информационная модель выражается в виде совокупности типов точек данных и созданных на их основе структур для хранения конкретных переменных процесса [7].

* + 1. Описание точек данных (Data Points)

Ключевым элементом при построении информационной модели в WinCC OA является понятие точки данных, связывающей переменные (теги), относящиеся к некоторому устройству или объекту, в единую древовидную структуру с возможностью создания произвольных подуровней. На практике в точку данных, относящуюся к одному устройству, объединяются примерно от 4 до 30 переменных.

В соответствии с идеологией WinCC OA для создания точек данных должен быть определен соответствующий тип точки данных, который будет выполнять роль шаблона при формировании множества точек данных с такой структурой [7].

Тип точки данных (DPT – data point type) – это описание структуры разнородных типизированных элементов, объединенных по признаку принадлежности к одному физическому объекту. Тип точки данных определяет формат представления сущности и показывает, в каком виде информация о процессе будет храниться в точке данных, какими элементами и какого типа будет описан реальный объект. Тип не хранит значение сущностей, конкретные значения будут храниться в соответствующей точке данных – экземпляре типа ТД. Точка данных (ТД, DP – data point) – это переменная, характеризующая состояние объекта, состоящая из элементов ТД, строго определенных структурой ТТД.

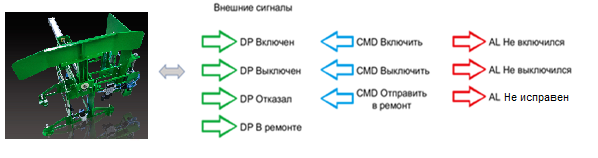
Элемент точка данных (ЭТД, DPE – data point element) – это динамическая переменная, обозначающая состояние объекта или внутренние состояния системы строго определенного типа данных [7].

На рисунках 13-15 представлены информационные модели оборудования разрабатываемой системы, здесь DP – Data Point – Сигналы состояния, CMD – Command – Команды управления, AL – Alert – Аварийные сообщения.

Рисунок 14 – информационная модель ленточного конвейера



  
Рисунок 15– Информационная модель сканера штрих-кода

  
Рисунок 16 – Информационная модель сталкивателя

В таблицах 4 и 5 представлено описание параметров оборудования, участвующих в построении информационной модели «Регистрация продукции».

В системе должно быть предусмотрено:

* выдача ошибок при нарушении технологического процесса;
* остановка приводного устройства при нарушении технологического процесса;
* аварийное ручное отключение.

К таким ситуациям можно отнести следующие случаи:

* сигнал о начале запуска оборудования;
* сигнал о выключении оборудования;
* выдача оповещений при механических отказах оборудования;
* выдача ошибок при сбое в работе оборудования;
* выдача ошибок при повреждении исторических данных;
* выдача ошибок при повреждении архивных параметров;
* выдача ошибок при введении неправильных параметров.
* затор товара на ленте;
* некорректная работа сталкивателя (изменение направления в противоположную сторону от необходимой);
* необходимость проведения ремонтных работ;
* неисполнение команд оператора оборудованием.

Таблица 4 – Описание параметров оборудования модели «регистрация продукции»

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Параметр | Диапазон значений | Тип параметра |
| Ленточный конвейер | длина | [10] м | фиксированный |
| ширина | [600] мм | фиксированный |
| скорость движения ленты | [1,0; 2,0] м/с | настраиваемый |
| статус | [включен; выключен] | логический |
| Приводное устройство | номинальная мощность привода | [0,5; 2,2] кВт | настраиваемый |
| скорость вращения вала | – | измеряемый |
| состояние | [работает; не работает] | логический |
| Сканер штрих-кода | скорость сканирования | [1200] s/p | фиксированный |
| состояние | [работает; не работает] | логический |
| Сталкиватель | статус | [включен; выключен] | логический |
| Транспортировочная тележка | вместимость изделий разных типов | [200; 600] шт | фиксированный |
| текущая вместимость изделий разных типов | –шт | измеряемый |
| Товаровыпускающее устройство | интенсивность | [5,25] шт/мин | настраиваемый |
| Тип товара | [Синий,Зеленый,Желтый] | измеряемый |

Таблица 5 – Описание точек данных модели «регистрация продукции»

| Точка данных | Команды | Состояния | Неисправности/Аварийные сообщения |
| --- | --- | --- | --- |
| Ленточный конвейер | включить,  выключить,  запустить,  остановить,  отправить в ремонт. | включен,  выключен,  в работе,  отказал,  в ремонте. | * не включился; * не выключился; * износ конвейерной ленты. |
| Приводное устройство | включить,  выключить,  запустить,  остановить,  отправить в ремонт. | включен,  выключен,  в работе,  отказал,  в ремонте. | * Приводное устройство неисправно; * не включилось; * не выключилось. |
| Сканер штрих-кода | включить,  выключить,  выключить,  отправить в ремонт. | включен,  выключен,  отказал,  в ремонте. | * не включился; * не выключился; * не считал штрих-код. |
| Сталкиватель | включить,  выключить; | включен,  выключен; | * не включился; * не выключился; * не исправен; * не получен сигнал от сканера. |
| Поток товара | запустить подачу,  остановить подачу. | включен,  отключен. | * мех. повреждение |

* 2. Проектирование интерфейса пользователя

Проектирование пользовательского интерфейса (ПИ) сегодня является очень важным аспектом разработки и требует от разработчиков специальных знаний о том, как разработать хороший интерфейс. Качество пользовательского интерфейса является самостоятельной характеристикой программного продукта, сопоставимо по значимости с такими его показателями, как надежность и эффективность использования вычислительных ресурсов [9].

Эффективный ПИ должен обеспечивать всестороннее использование потенциальных возможностей оператора, технических и информационно-программных средств АРМ, высокие безошибочность и быстродействие деятельности оператора при применении ПИ по назначению.

Отличительной особенностью интерфейса систем реального времени является наличие ограничения времени его реакции на внешние воздействия. В человеко-машинных системах реального времени в процессе управления участвует человек. Он может просто следить за тем, чтобы не случилось каких-нибудь аварийных ситуаций. Если они происходят, он должен предпринимать меры, помогая системе [9].

На рисунке 17 приведен прототип главной мнемосхемы разрабатываемой системы.

На рисунке 18 изображен прототип панели настройки оборудования, на которой администратор может задавать параметры оборудования. Также администратор имеет возможность настроить вероятности отказа оборудования (приводное устройство, транспортная лента, сканер, сталкиватель).

Рисунок 17 – Прототип мнемосхемы системы

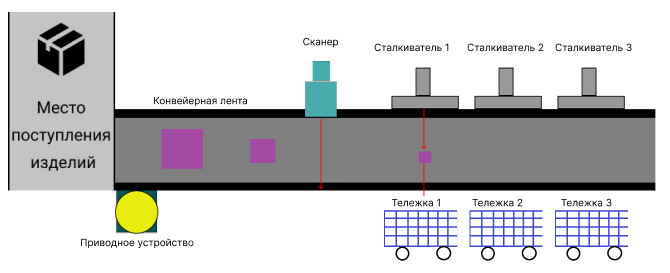
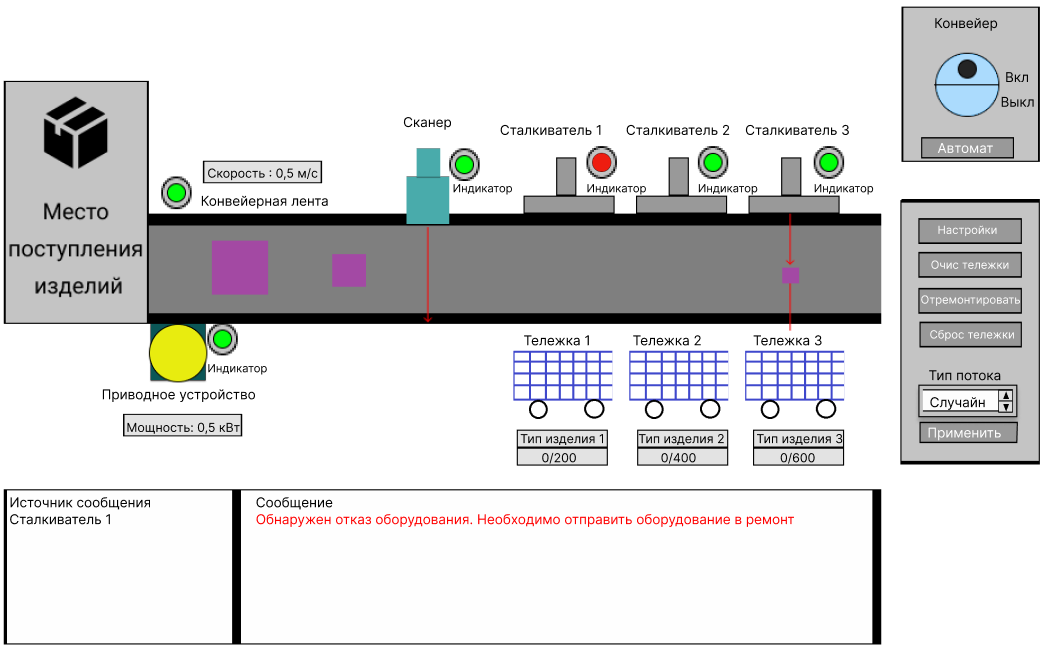


Рисунок 18 – Прототип панели настройки оборудования



Ниже приведен прототип главной панели разрабатываемой системы (рисунок 19). В роли оператора будут доступны: кнопка работы в автоматическом режиме. Администратору будет дополнительно доступна настройка параметров оборудования. А именно: настройка скорости конвейерной ленты, настройка мощности приводного устройства, управление наполнением тележек и их опустошением, а также будет возможность запустить систему в ручном режиме работы. Ниже представлены проектируемые панели для работы оператора и администратора. Аварийные сообщения о возникновении внештатной ситуации будут отображаться в нижней правой части рабочего окна. В списке будет выводиться время возникновения критической ситуации и описание проблемы. В системе будет предусмотрено возникновение внештатных ситуаций (alarm). Внештатная ситуация может быть связана с отказом оборудования или при переполнении корзин. Панель с настройками оборудования будет отображаться при нажатии на кнопку «Настройки», и доступна только администратору.

Рисунок 19 – Прототип главной панели приложения



* 1. Диаграмма вариантов использования

Диаграмма вариантов использования представляет собой наиболее общую концептуальную модель сложной системы, которая является исходной для построения всех остальных диаграмм. На ней изображаются отношения между актерами и вариантами использования [10].

Создание диаграммы вариантов использования имеет следующие цели:

* Определить общие границы и контекст моделируемой предметной области на начальных этапах проектирования системы.
* Сформулировать общие требования к функциональному поведению проектируемой системы.
* Разработать исходную концептуальную модель системы для ее последующей детализации в форме логических и физических моделей.
* Подготовить исходную документацию для взаимодействия разработчиков системы с ее заказчиками и пользователями.

На рисунке 20 приведена диаграмма вариантов использования со стороны системы. Система визуализирует при помощи анимации процесс работы устройств, обозначает статус подключенных устройств, даёт возможность пользователю изменять параметры поступающего товара (скорость, интенсивность), даёт возможность настроить объём контейнеров хранения. Также оповещает пользователя о повреждениях в системе.

На рисунках 21 и 22 приведена диаграмма вариантов использования (основная часть) со стороны пользователя. Работая с системой, пользователь может быть представлен в двух ролях: «Оператор» и «Администратор».

Рисунок 20 – Диаграмма вариантов использования (система)

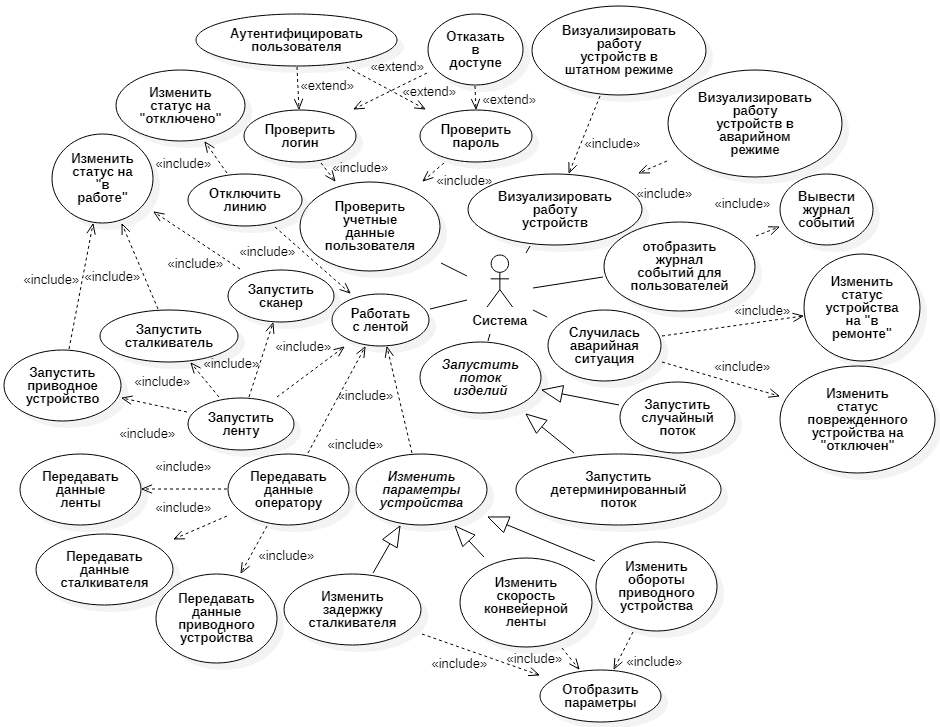


Рисунок 21 – Диаграмма вариантов использования (Оператор)

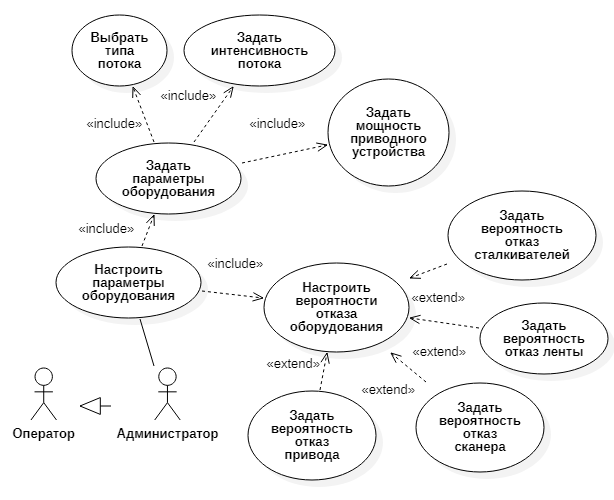
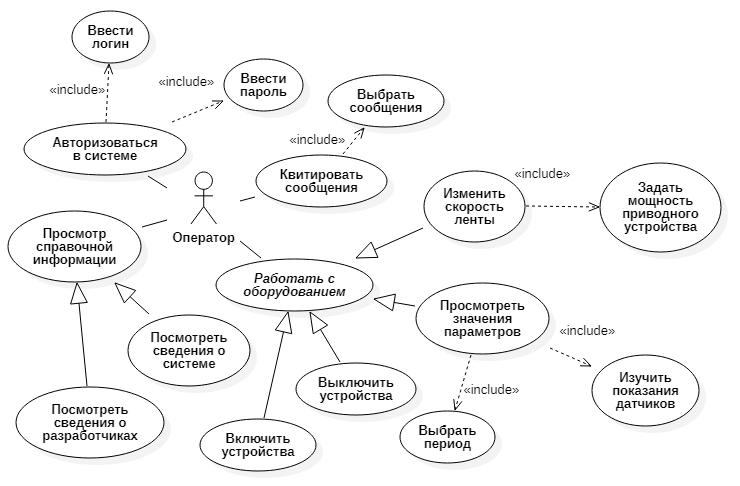


Рисунок 22 – Диаграмма вариантов использования (Администратор)

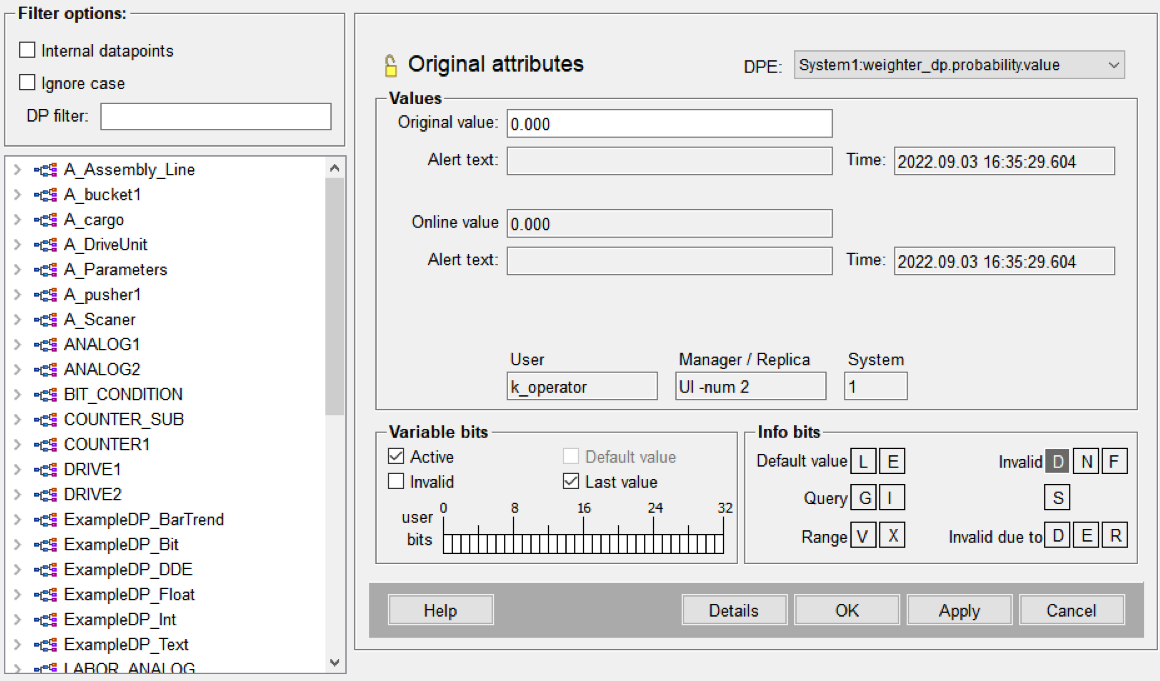
В качестве оператора пользователь может запустить или остановить линию, для этого необходимо запустить или остановить все механизмы (приводное устройство, сталкиватель, сканер, ленту). Также он может изменять скорость ленты, для чего необходимо изменить скорость приводного устройства. В случае непредвиденной ситуации пользователь может отключить линию в аварийном режиме.

Данные системы, полученные с датчиков в текущий момент времени, будут отображены для пользователя, а также их можно сравнить с номинальными данным отображаемыми для пользователя. Администратор может настраивать номинальные данные системы в ограниченном диапазоне (мощность приводного устройства, скорость конвейерной ленты).

1. Реализация системы
   1. **Настройка элементов точек данных**

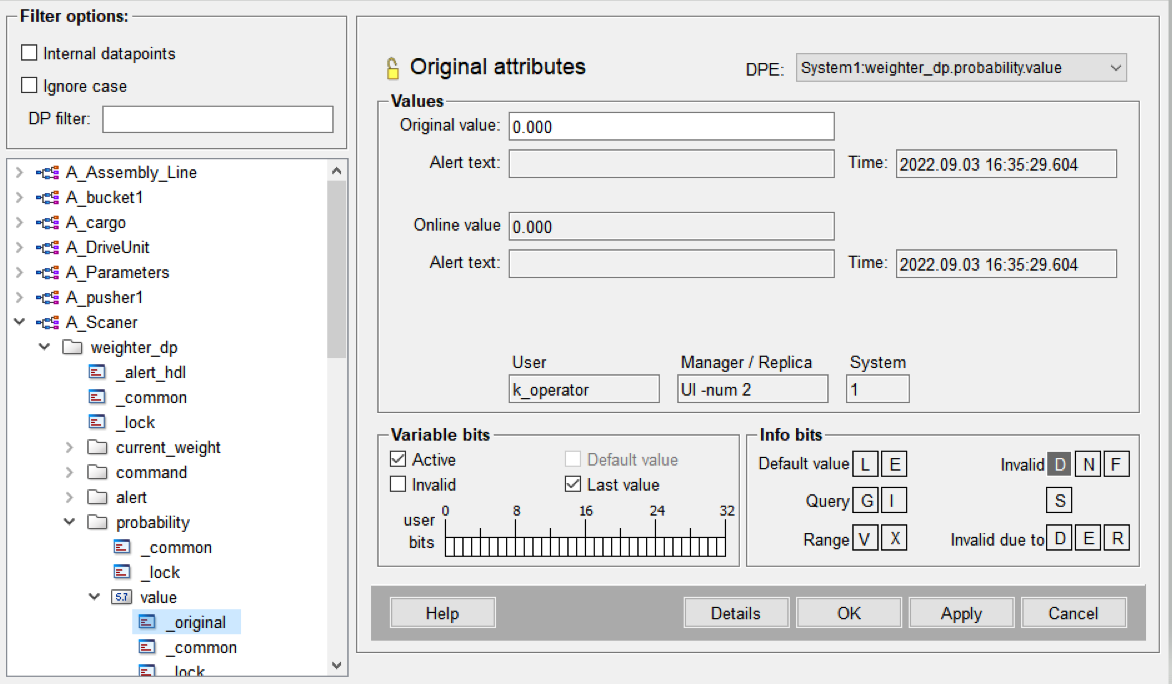
Для точной работы системы требуется организовать точки данных для элементов интерфейса. В WinCC OA точки данных представлены в виде дерева (рисунок 23).

Рисунок 23 – Структура точек данных



В соответствии с информационной моделью системы, авторами были внесены с систему необходимые точки данных, на рисунке 24 приведен пример точки данных «**Сканер**», остальные ТД приведены в приложении А.

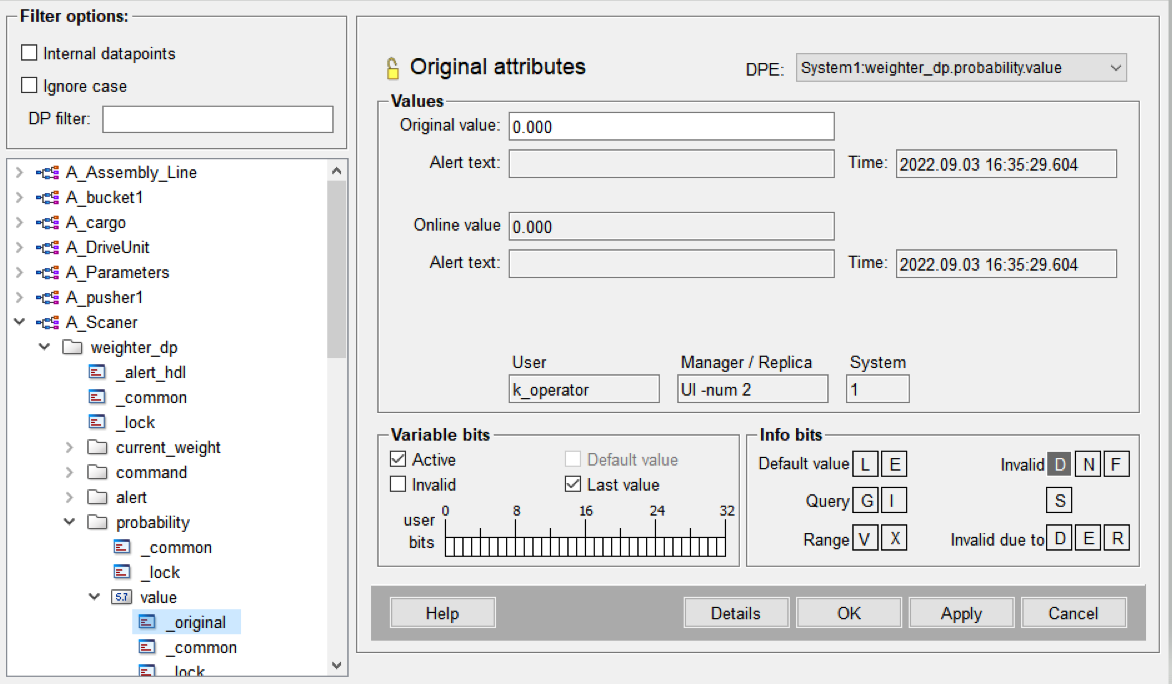
Рисунок 24 – Описание ТД «Сканер»



* 1. **Настройка исходных значений**

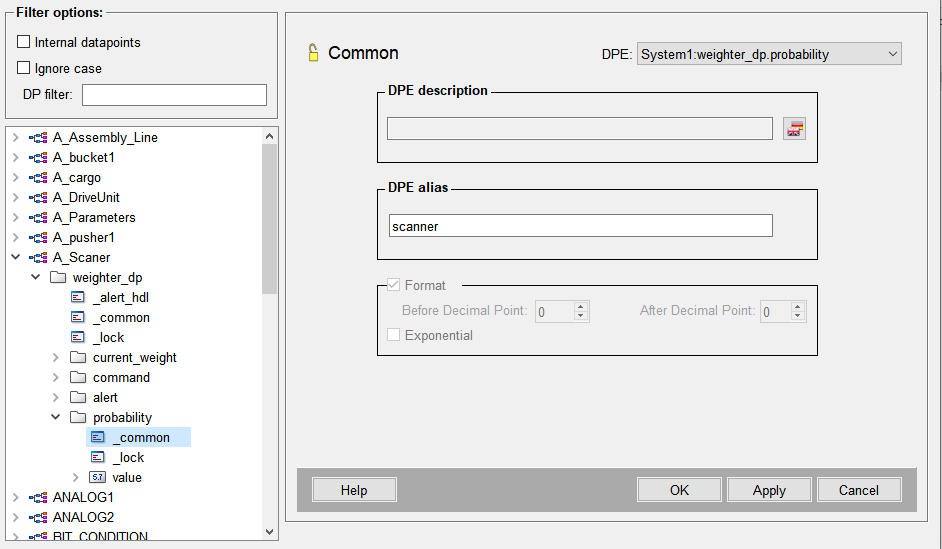
В зависимости от процесса поступления изделий разных типов, значение будет изменяться, поэтому базовое значение 0. На рисунке 25 представлено начальное значение точки данных «Сканер».

Рисунок 25 – Начальное значение \_original ТД «Сканер»



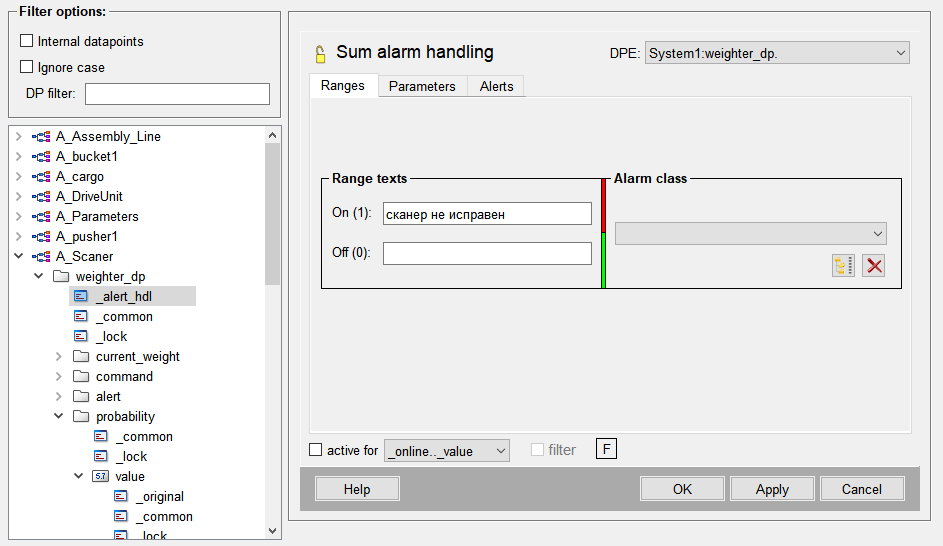
* 1. **Общие настройки объекта**

**Существует возможность также совершить общие настройки. Например, добавить описание для соответствующих типов точек данных (рисунок 26).**

  
Рисунок 26 – Настройка значения \_common для ТД «Сканер»

* 1. **Обработка сообщений (alert)**

Если случилась аварийная ситуация, то можно настроить вывод соответствующего сообщения (рисунок 27).

  
Рисунок 27 – Настройка \_alert\_hdl для ТД «Сканер»

* 1. **Разработка и описание графического интерфейса**

Графический пользовательский интерфейс (GUI) – один из основных элементов WinCC OA, с которым взаимодействуют операторы/ администратор системы для работы и отладки системы. Интерфейс является частью программного обеспечения и спроектирован таким образом, чтобы обеспечить корректную передачу информации [11].

Пользовательский интерфейс может быть графическим, текстовым и аудио-видео, в зависимости от базовой аппаратной и программной комбинации. Пользовательский интерфейс может быть аппаратным или программным, или их комбинацией.

Используя GUI, пользователь интерпретирует программное обеспечение. Как правило, GUI более ресурсоемкий, чем интерфейс командной строки (CLI). С помощью передовых технологий программисты и дизайнеры создают сложные графические интерфейсы, которые работают с большей эффективностью, точностью и скоростью.

GUI предоставляет набор компонентов для взаимодействия с программным или аппаратным обеспечением. Каждый графический компонент предоставляет способ работы с системой. Система GUI имеет следующие элементы, такие как: Окно – область, где отображается содержимое приложения [11].

3.5.1 Описание графических элементов мнемосхемы

В разрабатываемую систему входят элементы, помогающих оператору эффективно использовать возможности системы, их описание приведено в таблицах 6 и 7.

Таблица 6 – Описание пиктограмм

|  |  |
| --- | --- |
| 1 | 2 |
| Пиктограмма | Назначение |
|  | Выключатель (включает/выключает приводное устройство) |
|  | Точкой поступления изделий на транспортер |
|  | Служит для передвижения изделий (транспортер) |
| пушер | Сталкиватели служат для отсеивания изделий |

Продолжение таблицы 6

|  |  |
| --- | --- |
| 1 | 2 |
|  | Сканер служит для определения типа проходящего изделия |
|  | Служит для хранения брака |
|  | Показывает статус устройств |
|  | Служат для хранения изделий |
|  | Ремонтирует поврежденный элемент (доступно только в ручном режиме) |
|  | Включает/выключает требуемый элемент (доступно только в ручном режиме) |

Таблица 7 – Описание кнопок для управления

|  |  |
| --- | --- |
| Кнопка | Назначение |
|  | Запускает систему в автоматическом режиме с базовыми параметрами |
|  | Запускает ручной режим системы. |
|  | Показывает всплывающее окно с кратким руководством пользователя |
|  | Показывает всплывающее окно с информацией о разработчиках |
|  | Показывает доступные исторические данные в виде графиков |
|  | Показывает доступные исторические данные в виде таблицы с возможностью фильтрации |

3.5.2 Начало работы

В системе предусмотрено две роли пользователей: администратор и оператор. Перед началом работы с системой пользователи должны авторизоваться. В зависимости от выбранной роли отличается и набор доступных параметров для просмотра и редактирования, в системе авторизация реализована при помощи стандартных инструментов WinCC OA (рисунок 28).

Также любой пользователь может посмотреть информацию о разработчиках (рисунок 29).

Рисунок 28 – Авторизация в системе

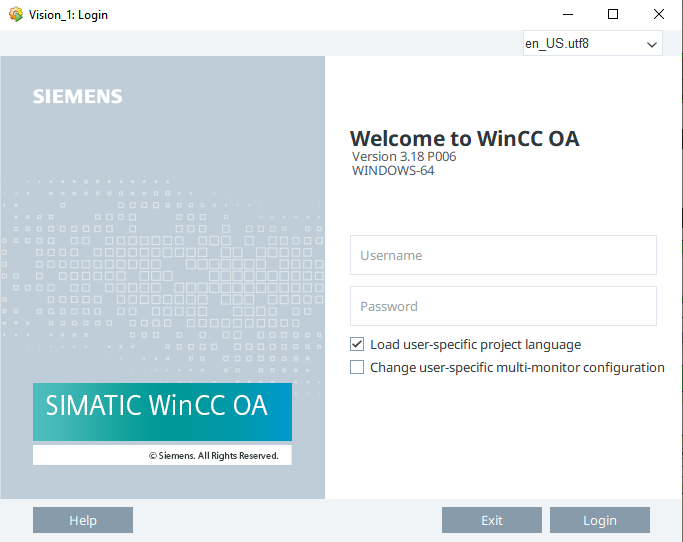
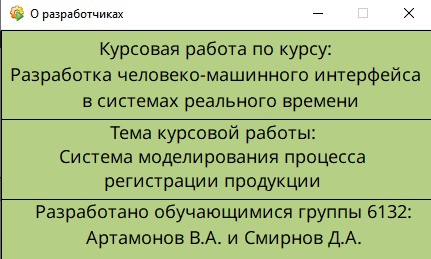


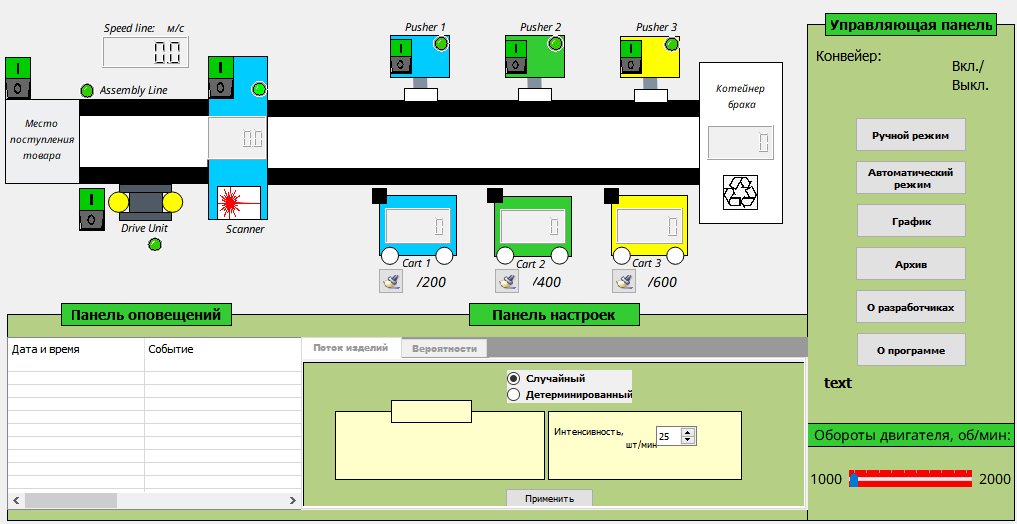
Рисунок 29 – Информация о разработчиках



При авторизации пользователя в роли «Оператора» пользователю доступны кнопка запуска/остановки устройств системы (рисунок 30).

Администратору дополнительно доступна настройка параметров оборудования: настройка типов поступающих изделий, настройка потока изделий, настройка параметров вероятности отказа. Ниже представлены панели для работы оператора (рисунок 30) и для администратора (рисунок 31).

Рисунок 30 – Панель работы оператора



В разрабатываемую систему входят элементы, помогающих оператору/администратору эффективно использовать возможности системы тарификации, их описание приведено в таблицах 6 и 7.

3.5.3 Режим администратора

После входа в систему администратору открывается рабочее окно с мнемосхемой и панелью управления (см. рисунок 31). Администратор может провести настройку начальных параметров в «Панель настроек», которая находится в нижней части экранной формы для детального управления параметров системы (рисунки 32, 33).

Администратор может выбрать поток поступления товара (Случайный/Детерминированный), изменить тип поступающего изделий (детерминированный поток), интенсивность потока изделий, запустить/остановить конвейер, включить работу системы в ручном или автоматическом режиме, а также просматривать журнал сообщений, отремонтировать оборудование (в ручном режиме), очищать корзину, задавать вероятности отказа оборудования (см. рисунок 34).

Рисунок 31 – Панель работы администратора

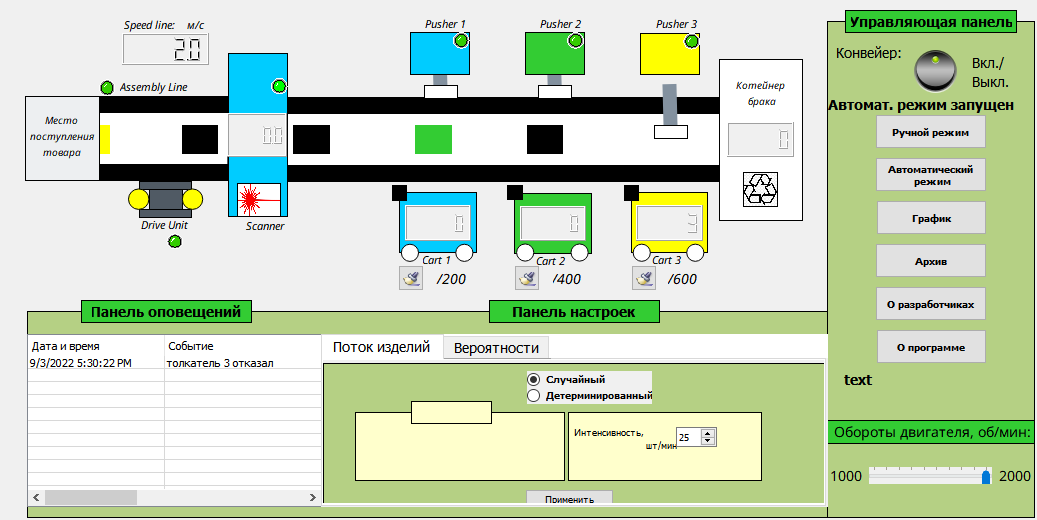
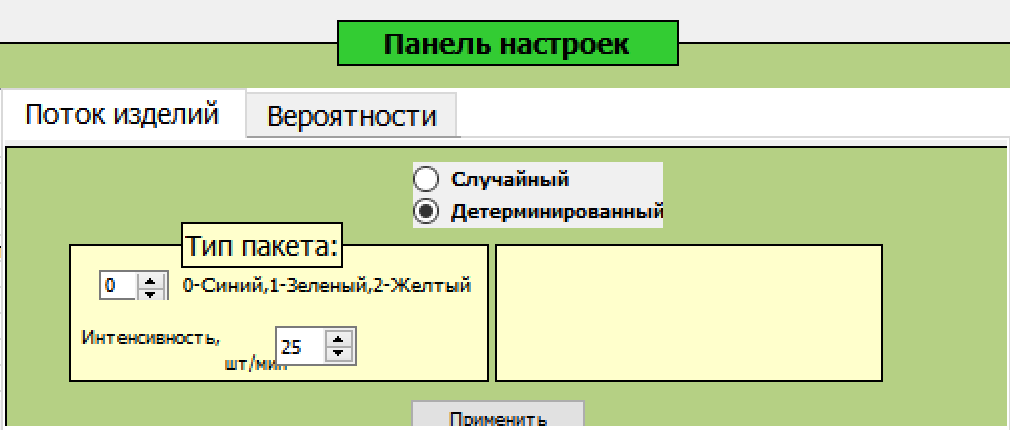
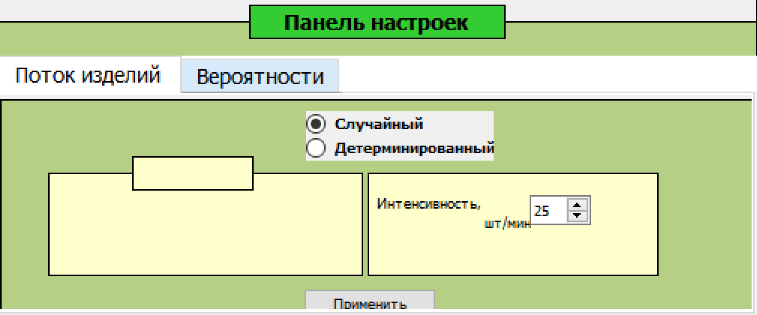
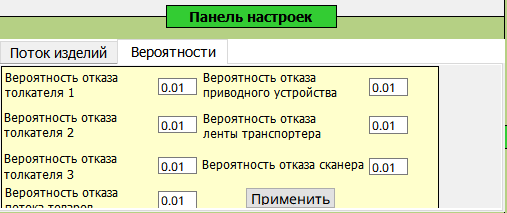


Рисунок 32 – Панель настроек (Поток изделий: «Случайный2)  
  
Рисунок 33 – Панель настроек (Поток изделий: «Детерминированный»)

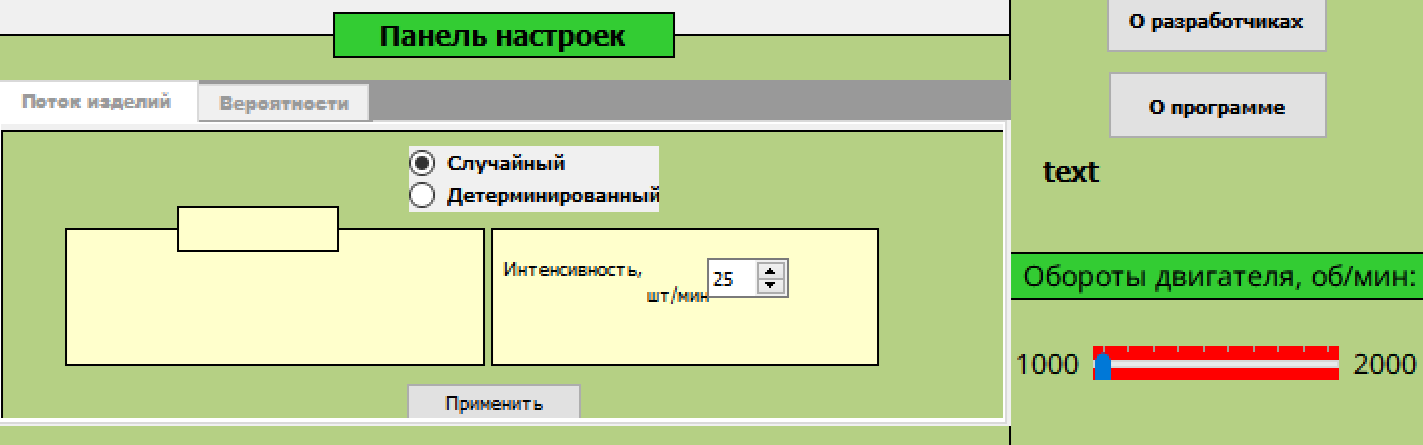


  
Рисунок 34 – Панель настроек («Вероятности»)

3.5.4 Режим оператора

Оператор начинает работу с начального окна программы (см. рисунок 30). Для него раздел с настройками заблокирован, также, как и раздел приводного устройства. Он может использовать систему как в автоматическом режиме, так и в ручном (для работы с внештатными ситуациями), также ему доступна регулировка скорости работы ленты транспортера, функции очистка корзин и полный функционал панели управления.

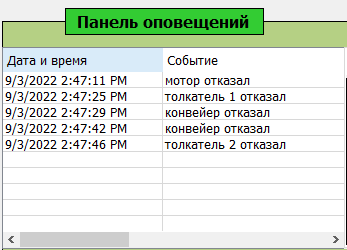
Рисунок 35 – Заблокированные панели настроек и приводного устройства   
для роли «Оператор»



3.5.5 Обработка нештатных ситуаций

В системе предусмотрена обработка нештатных ситуаций (alarm). Нештатная ситуация может быть связана с отказом оборудования или при переполнении корзин. Аварийные сообщения о возникновении нештатной ситуации отображаются в нижней части рабочего окна. Также на рабочей панели красным цветом подсвечивается вышедший из строя компонент системы и появляется кнопка для его починки, также этот элемент потребуется отдельно включить (в ручном режиме), нажав по соответствующей кнопке. В списке выводятся время возникновения критической ситуации, описание проблемы, места, а также возможность квитирования (рисунок 36).

Рисунок 36 – Пример сообщений о внештатных ситуациях



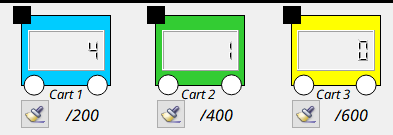
При отказе какого-либо оборудования у данного элемента системы появляется кнопка для выполнения ремонта. При нажатии на нее отказавшее оборудование восстанавливает работоспособность (рисунок 37).



  
Рисунок 37 – Оповещение о необходимости   
выполнения ремонта оборудования

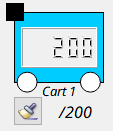
Также есть кнопка включения/отключения элемента  и кнопка , отражающая состояние элемента.

Во время работы системы можно отслеживать изменение значений весов, корзин для готовых изделий и корзины брака (рисунок 38).

  
Рисунок 38 – Пример отображения заполняемости тележек №1, 2, 3

При заполнении тележки Администратор/Оператор имеют возможность очистить необходимую тележку нажатием кнопки «Очистить», также, при необходимости это можно сделать и раньше (рисунок 39).

Рисунок 39 – Пример функции очищения корзины

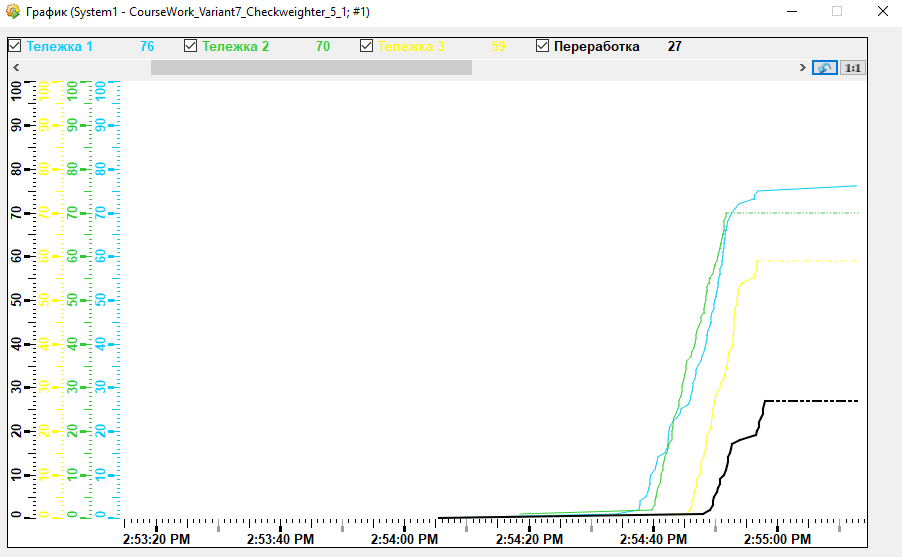


3.5.6 Просмотр параметров оборудования

**Архивные значения отображаются в виде графиков и имеют четыре кривые, показывающие заполняемость тележек и корзины брака от времени**. Таким образом можно отследить изменение значений параметра в определенный период времени.

**Пример графика зависимости значения** **заполняемости тележек и корзины от времени приведен на рисунке 40.**

Рисунок 40 – График зависимости **заполняемости тележек и   
корзины от времени**



**Заключение**

**В ходе выполнения курсового проектирования были изучены возможности SCADA-системы WinCC OA 3.17, с ее помощью была разработана система моделирования процесса регистрации продукции.**

В первой главе приведены основные понятия предметной области, рассмотрены принципы работы систем сортировки грузов, изучены виды оборудования, входящие в состав таких систем, рассмотрены их основные параметры. На основании данной информации были определены основные функции разрабатываемой системы и требования к ней.

Во второй главе приведены сведения о проектировании системы. Они содержат описание точек данных, используемых в ней, и прототип мнемосхемы модели. Так же были разработаны диаграммы вариантов использования для различных ролей пользователей и системы.

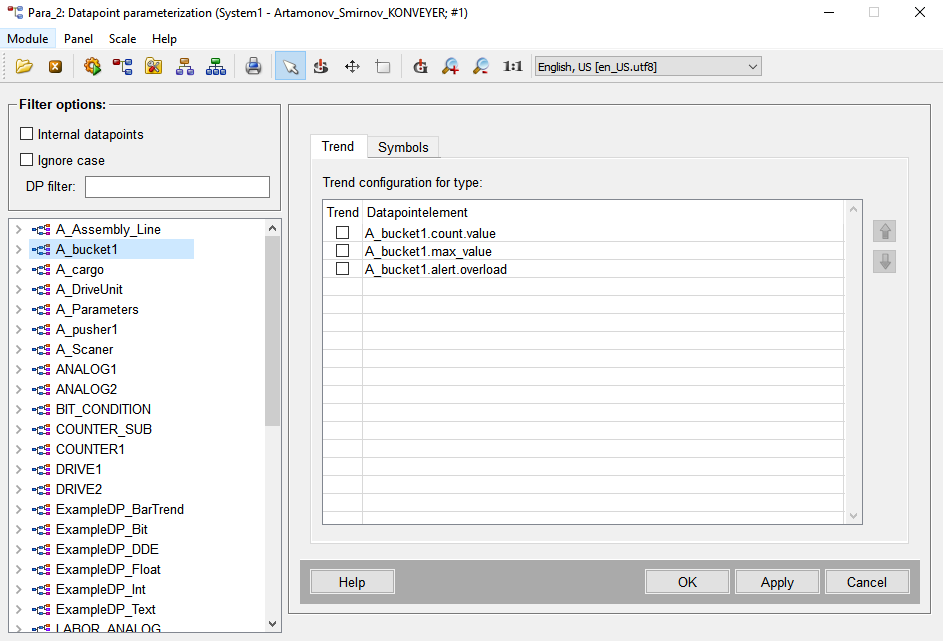
В третьей главе описаны основные возможности разработанной системы: интерфейс пользователя с расположенной на главной панели мнемосхемой, созданы точки данных, написаны скрипты для реализации работы системы (см. приложение В) и ее взаимодействия с пользователем, при нарушении допустимых норм в журнал сообщений приходит предупреждение.

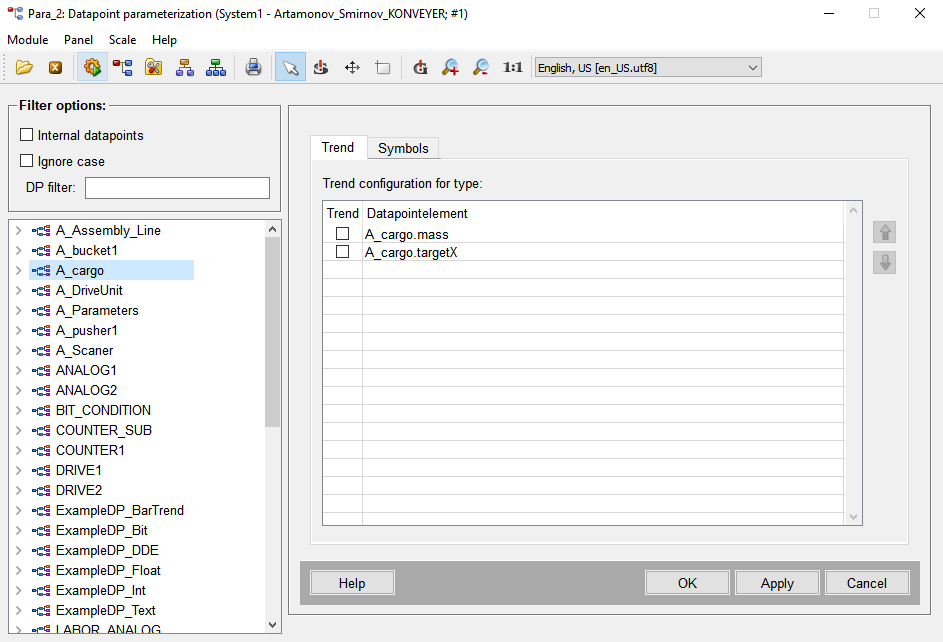
Разработанная во время курсового проектирования система может быть использована на конвейерных линиях регистрации различной продукции.

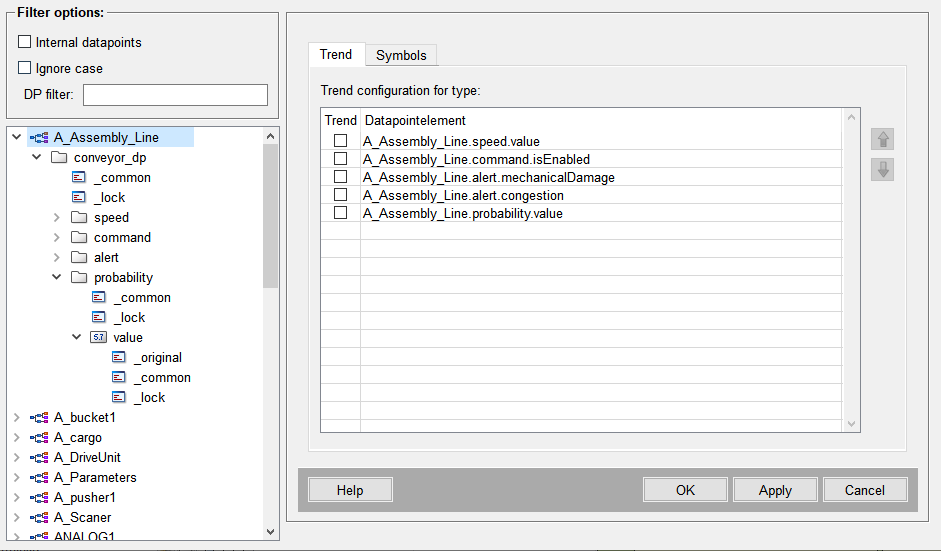
**Список использованных источников**

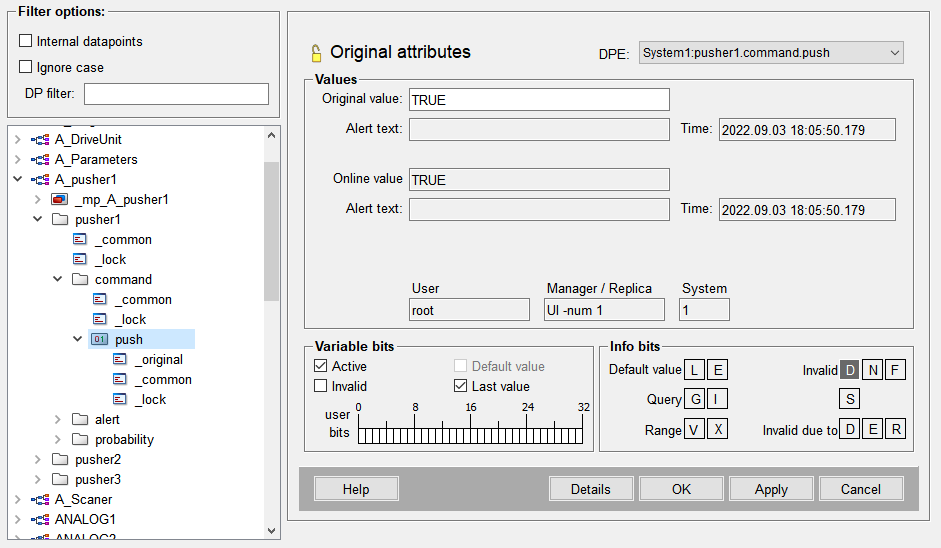
1. SCADA-системы [Электронный ресурс]. URL: https://sfsamgtu.com/epa/docs/ITiOvNGO/SCADA.pdf (дата обращения: 17.02.2022).
2. UML [Электронный ресурс] // Википедия: [сайт]. URL: [https://ru.wikipedia.org/wiki/UML](https://ru.wikipedia.org/wiki/UML#Диаграммы) (дата обращения 17.02.2022).
3. Конвейер. Виды и применение. Назначение и особенности [Электронный ресурс] URL: https://tehpribory.ru/glavnaia/oborudovanie/  
   konvejer.html (дата обращения: 17.02.2022).
4. ИД РАША – Конвейерные сканеры [Электронный ресурс]. URL: <https://id-russia.ru/hardware/conveyor-scanners> (дата обращения: 17.02.2022).
5. Сортировочные конвейеры со сталкивателями | Конвейермаш [Электронный ресурс] URL: <https://conveermash.ru/catalog/sortirovochnye-konveyery-so-stalkivatelyami> (дата обращения: 17.02.2022).
6. Ленточный конвейер [Электронный ресурс] // Википедия: [сайт]. URL: <https://ru.wikipedia.org/wiki/Ленточный_конвейер> (дата обращения: 17.02.2022).
7. Мониторинг производственных линий с SINATIC WinCC Open Architecture [Электронный ресурс]. https://zen.yandex.ru/media/id/  
   5c8b8c6f8cbd2100b0f2f811/monitoring-proizvodstvennyh-linii-s-simatic-wincc-open-architecture-5c93745f573e4700b4e6baaa (дата обращения: 17.02.2022).
8. Грузовая тележка. Виды и типы. Применение и особенности [Электронный ресурс] URL:.https://tehpribory.ru/glavnaia/oborudovanie/  
   gruzovaia-telezhka.html (дата обращения: 21.03.2022);
9. Разработка человеко-машинного интерфейса и его применение в системах управления [Электронный ресурс]. URL: <https://cyberleninka.ru/article/n/razrabotka-cheloveko-mashinnogo-interfeysa-i-ego-primenenie-v-sistemah-upravleniya> (дата обращения 21.03.2022).
10. НОУ ИНТУИТ | Лекция | Элементы графической нотации диаграммы вариантов использования [Электронный ресурс]. URL: <https://intuit.ru/studies/courses/32/32/lecture/1004> (дата обращения 21.03.2022).
11. Графический интерфейс пользователя [Электронный ресурс]. URL: https://www.gamma.spb.ru/novosti-proizvoditelej/microchip/stati-microchip/79-graficheskij-interfejs-polzovatelya#:~:text=Графический%20интерфейс%  
    20пользователя%20(Graphical%20User,ко%20всем%20видимым%20экранным%20объектам (дата обращения: 01.07.2022).

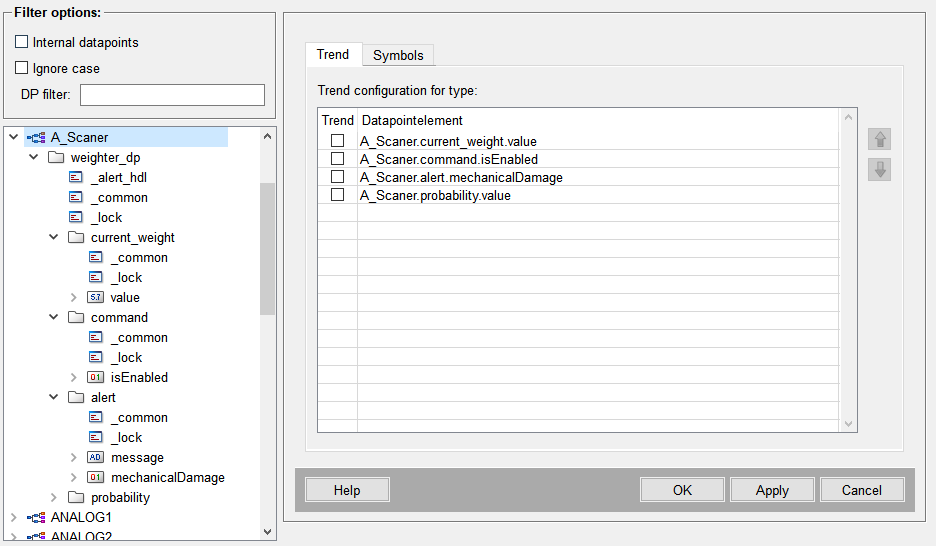
ПРИЛОЖЕНИЕ А  
Описание точек данных

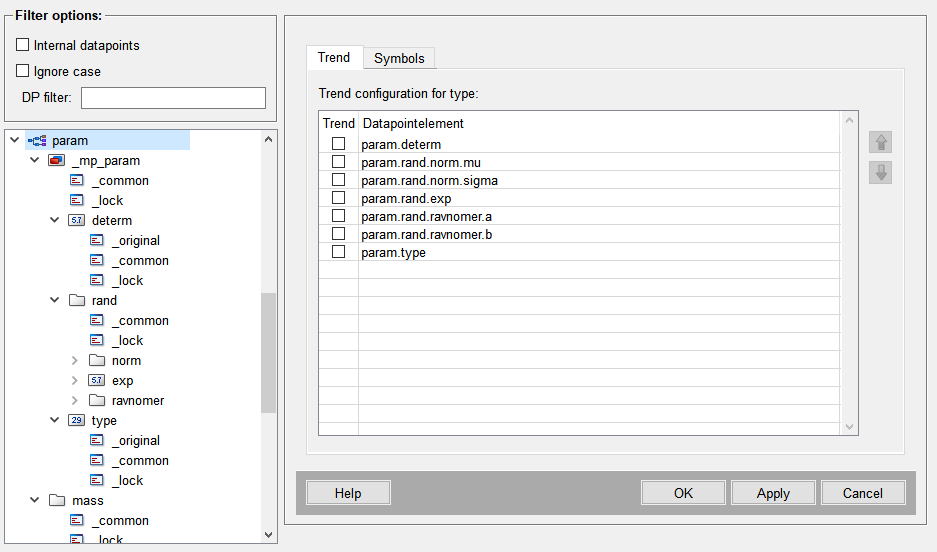
  
Рисунок А.1 – Структура точек данных

  
Рисунок А.2 – Описание ТД «A\_cargo» («Изделие»)

  
Рисунок А.3 – Описание ТД A\_Assembly\_Line («Транспортная лента»)

  
Рисунок А.4 – Описание ТД «A\_pusher1 («Сталкиватель»)

  
Рисунок A.5 – Описание ТД «A\_Scaner» («Сканер»)

  
Рисунок A.6 – Описание ТД «param»   
 («Параметры для задания законов распределения»)

ПРИЛОЖЕНИЕ Б  
Сценарии (скрипты управления)

─// [(Panel)] [0] - [ScopeLib]

bool isConveyorWorking = FALSE;

bool isCargoGenerating = FALSE;

float convSpeed = 1;

int fullBucketNumber = 0;

int genDistribution = 2;

int massDistribution = 1;

int RAND\_MAX = 32767;

float expectedValue = 5;

float dispSqr = 2;

int b = 10;

int a = 3;

double lamb = 0.1;

float weighterThread = 0.0;

float sourceThread = 0.0;

float motorThread = 0.0;

float pusher1Thread = 0.0;

float pusher2Thread = 0.0;

float pusher3Thread = 0.0;

float conveyorThread = 0.0;

float expectedValueTime = 10;

float dispSqrTime = 30;

int bTime = 5;

int aTime = 1;

double lambTime = 0.1;

int determTime = 2;

int determMass = 2;

dyn\_string panelNames;

dyn\_string threadNames;

dyn\_int multyThreadsIdis;

dyn\_int threadIds;

bool isHand = FALSE;

bool isAuto = FALSE;

public int m1 = 1, m2 = 3, m3 = 6, m4 = 10;

void reset(){

for (int i = 1; i <= dynlen(panelNames); i++) {

removeSymbol(myModuleName(), myPanelName(), panelNames[i]);

dpDelete(panelNames[i]);

dynRemove(panelNames, i);

}

for (int i = 1; i <= dynlen(threadIds); i++)

{

DebugN("Th " + threadIds[i]);

stopThread(threadIds[i]);

waitThread(threadIds[i]);

dynRemove(threadIds, i);

}

dpSet("System1:conveyor\_dp.command.isEnabled", true);

dpSet("System1:A\_motor1.command.isEnabled",true);

setValue("ToggleSwitch\_ewo1", "enabled", true);

dpSet("System1:m1.", 1);

dpSet("System1:m2.", 3);

dpSet("System1:m3.", 6);

dpSet("System1:m4.", 10);

dpSet("System1:mass.type", 1);

dpSet("System1:mass.determ", 1);

dpSet("System1:mass.rand.norm.mu", 1);

dpSet("System1:mass.rand.norm.sigma", 1.5);

dpSet("System1:mass.rand.ravnomer.a", 2);

dpSet("System1:mass.rand.ravnomer.b", 5);

dpSet("System1:mass.rand.exp", 7);

dpSet("System1:time.type", 0);

dpSet("System1:time.determ", 1);

dpSet("System1:time.rand.norm.mu", 5);

dpSet("System1:time.rand.norm.sigma", 2);

dpSet("System1:time.rand.ravnomer.a", 2);

dpSet("System1:time.rand.ravnomer.b", 5);

dpSet("System1:time.rand.exp", 7);

setValue("LCD2", "intValue", 0);

setValue("LCD3", "intValue", 0);

setValue("LCD4", "intValue", 0);

setValue("LCD5", "intValue", 0);

}

──────────────────────────────────────────────────────

─// [(Panel)] [0] - [Initialize]

main()

{

panelNames = makeDynString();

multyThreadsIdis = makeDynInt();

dpSet("System1:A\_source1.intensity.value", 25);

dpSet("System1:A\_source1.deviation.d1", 1);

dpSet("System1:A\_source1.deviation.d2", 10);

LCD1.decimals(1);

dpSet("System1:TIP.tip",0);

float current;

dpSet("System1:weighter\_dp.probability.value", 0.01);

dpSet("System1:A\_source1.probability.value", 0.01);

dpSet("System1:A\_motor1.probability.value", 0.01);

dpSet("System1:pusher1.probability.value", 0.01);

dpSet("System1:pusher2.probability.value", 0.01);

dpSet("System1:pusher3.probability.value", 0.01);

dpSet("System1:conveyor\_dp.probability.value", 0.01);

dpGet("System1:weighter\_dp.probability.value", current);

LCD10.decimals(1);

while(true) {

bool motor;

dpGet("System1:A\_motor1.command.isEnabled", motor);

if (isConveyorWorking && motor){

if(convSpeed == 1){

setValue("LCD10", "value",1 );

} else {

setValue("LCD10", "value",1 + convSpeed/10);

}

}else {

setValue("LCD10", "value",0);

}

delay(0,10);

}

}

══════════════════════════════════════════════════════

─// [PUSH\_BUTTON4] [3] - [Clicked]

main(mapping event)

{

string dpName = "System1:A\_bucket" + fullBucketNumber + ".count.value";

dpSet(dpName,0);

setValue("LCD" + (fullBucketNumber + 1), "intValue", 0);

isConveyorWorking = true;

isCargoGenerating = true;

dpSet("System1:conveyor\_dp.command.isEnabled", true);

fullBucketNumber=0;

setValue("PUSH\_BUTTON4", "visible", false);

ToggleSwitch\_ewo1.enabled(true);

ToggleSwitch\_ewo1.checked(true);

}

══════════════════════════════════════════════════════

─// [PUSH\_BUTTON5] [4] - [Clicked]

main(mapping event)

{dpSet("System1:weighter\_dp.alert.mechanicalDamage", false);

dpSet("System1:pusher1.alert.mechanicalDamage", false);

dpSet("System1:pusher2.alert.mechanicalDamage", false);

dpSet("System1:pusher3.alert.mechanicalDamage", false);

dpSet("System1:conveyor\_dp.alert.mechanicalDamage", false);

dpSet("System1:A\_source1.alert.mechanicalDamage", false);

dpSet("System1:A\_motor1.alert.damage", false);

dpSet("System1:pusher1.command.push", true);

dpSet("System1:pusher2.command.push", true);

dpSet("System1:pusher3.command.push", true);

dpSet("System1:conveyor\_dp.command.isEnabled", true);

dpSet("System1:weighter\_dp.command.isEnabled", true);

dpSet("System1:A\_motor1.command.isEnabled", true);

string dpName = "System1:A\_bucket" + fullBucketNumber + ".count.value";

DebugN("dpname="+dpName);

dpSet(dpName,0);

setValue("LCD" + (fullBucketNumber + 1), "intValue", 0);

isConveyorWorking = true;

isCargoGenerating = true;

dpSet("System1:conveyor\_dp.command.isEnabled", true);

fullBucketNumber=0;

setValue("PUSH\_BUTTON5", "visible", false);

ToggleSwitch\_ewo1.enabled(true);

ToggleSwitch\_ewo1.checked(true);

}

══════════════════════════════════════════════════════

─// [LCD2] [10] - [Initialize]

main()

{

float n;

dpGet("System1:A\_bucket1.count.value", n);

if(n==200.0){

dpSet("System1:A\_bucket1.count.value", 0.0);

setValue("LCD2", "intValue", 0.0);}

}

══════════════════════════════════════════════════════

─// [LCD3] [11] - [Initialize]

main()

{

float n;

dpGet("System1:A\_bucket2.count.value", n);

if(n==400.0){

dpSet("System1:A\_bucket2.count.value", 0.0);

setValue("LCD3", "intValue", 0.0);}

}

══════════════════════════════════════════════════════

─// [ToggleSwitch\_ewo1] [14] - [toggled]

toggled(bool on)

{

isConveyorWorking = on;

isCargoGenerating = on;

if(isAuto){

dpSet("System1:A\_motor1.alert.damage", on);

dpSet("System1:conveyor\_dp.command.isEnabled", on);

}

}

══════════════════════════════════════════════════════

─// [SLIDER1] [15] - [Slide]

Slide(int value)

{

convSpeed = value;

DebugN(value);

}

──────────────────────────────────────────────────────

─// [SLIDER1] [15] - [Initialize]

main()

{

if(!getUserPermission(6)){

SLIDER1.transparentForMouse(true);

this.color("red");

SLIDER1.font();

DebugN("authorization for level 4 ok");}

else DebugN("no authorization for level");

}

══════════════════════════════════════════════════════

─// [PUSH\_BUTTON10] [22] - [Clicked]

dyn\_string dpNames;

dyn\_int deletedIds;

int startY = 130;

int startX = 40;

int pistonX5 = 440;

int pistonX1 = 580;

int pistonX3 = 780;

int pistonX4 = 855;

int pistonX2 = 710;

int weightX = 225;

string bucketLabel1 = "LCD2";

string bucketLabel2 = "LCD3";

string bucketLabel3 = "LCD4";

string bucketLabel4 = "LCD5";

mapping cargosToPush;

int amount1, amount2, amount3, amount4;

shape table = getShape("TABLE1");

main()

{

ToggleSwitch\_ewo1.visible = true;

PRIMITIVE\_TEXT62.visible = TRUE;

PRIMITIVE\_TEXT62.text = "Автомат. режим запущен";

PANEL\_REF22.position(2000,2000);

PANEL\_REF21.position(2000,2000);

PANEL\_REF20.position(2000,2000);

PANEL\_REF19.position(2000,2000);

PANEL\_REF18.position(2000,2000);

PANEL\_REF25.position(2000,2000);

isHand = FALSE;

isAuto = TRUE;

reset();

init();

runThreads();

}

void init() {

dpNames = makeDynString();

setValue("ToggleSwitch\_ewo1", "checked", true);

dpSet("System1:A\_bucket1.count.value", 0);

dpSet("System1:A\_bucket2.count.value", 0);

dpSet("System1:A\_bucket3.count.value", 0);

dpSet("System1:A\_bucket4.count.value", 0);

}

void runThreads(){

for (int i = 1; i <= dynlen(multyThreadsIdis); i++)

{DebugN("Th " + multyThreadsIdis[i]);

stopThread(multyThreadsIdis[i]);

waitThread(multyThreadsIdis[i]);

dynRemove(multyThreadsIdis, i);}

int upt = startThread("update");

int gen = startThread("generateCargos");

int br = startThread("breakerThread");

dynAppend(multyThreadsIdis, upt);

dynAppend(multyThreadsIdis, gen);

dynAppend(multyThreadsIdis, br);

}

void updateMass(){

float mass1, mass2, mass3,mass4;

dpGet("System1:A\_bucket1.max\_value", amount1);

dpGet("System1:A\_bucket2.max\_value", amount2);

dpGet("System1:A\_bucket3.max\_value", amount3);

dpGet("System1:A\_bucket4.max\_value", amount4);

dpGet("System1:A\_bucket1.max\_value", mass1);

dpGet("System1:A\_bucket2.max\_value", mass2);

dpGet("System1:A\_bucket3.max\_value", mass3);

dpGet("System1:A\_bucket4.max\_value", mass4);

setValue("PRIMITIVE\_TEXT30", "text", mass1);

setValue("PRIMITIVE\_TEXT34", "text", mass2);

setValue("PRIMITIVE\_TEXT32", "text", mass3);

dpGet("System1:m1.", m1);

dpGet("System1:m2.", m2);

dpGet("System1:m3.", m3);

dpGet("System1:m4.", m4);

}

void generateCargos(){

int id = 0;

while(true) {

if(isHand) {

return;

}

if (!isCargoGenerating) {continue;}

updateMass();

generateCargoWithID(id++);

int del = getDelay() \* 1000;

delay(0, del);

}

}

void generateCargoWithID(int id) {

string name = "CARGO" + id;

dpCreate(name, "A\_cargo");

float mass = generateMass();

int targetX = getTargetX(mass);

dpSet(name + ".mass", mass);

dpSet(name + ".targetX", targetX);

string weightLabel = "$weight:" ;

string colorStr = "$color:";

int color;

float d1,d2;

dpGet("System1:A\_source1.deviation.d1", d1);

dpGet("System1:A\_source1.deviation.d2", d2);

float d1L = 1 - d1/100;

float d1R = 1 + d1/100;

float d2L = 1 - d2/100;

float d2R = 1 + d2/100;

if (mass ==0) {color = 1;}

else if ( mass ==1) {color = 2;}

else if(mass ==2) {color=3;}

else {color = 4;}

switch(color) {

case 1:

colorStr = colorStr + "STD\_value\_invalid";

break;

case 2:

colorStr = colorStr + "STD\_valve\_open";

break;

case 3:

colorStr = colorStr + "Yellow";

break;

case 4:

colorStr = colorStr + "Black";

break;

}

addSymbol(myModuleName(), myPanelName(), "gedi/Cargo.pnl",name,makeDynString(weightLabel, colorStr),startX, startY,0,0,0);

dynAppend(panelNames, name);}

float generateMass(){

int dist;

dpGet("System1:mass.determ",determMass);

dpGet("System1:mass.type",dist);

dpGet("System1:mass.rand.norm.mu",expectedValue);

dpGet("System1:mass.rand.norm.sigma",dispSqr);

dpGet("System1:mass.rand.ravnomer.a",a);

dpGet("System1:mass.rand.ravnomer.b",b);

dpGet("System1:mass.rand.exp",lamb);

switch(dist) {

case 0:

return determMass;

case 1 :

return getNormDistribValue(expectedValue,dispSqr);

}

}

int getTargetX(float mass) {

float d1,d2;

dpGet("System1:A\_source1.deviation.d1", d1);

dpGet("System1:A\_source1.deviation.d2", d2);

if (mass==0) {

return pistonX5;

} else if (mass==1)

{return pistonX1;}

else if(mass==2)

{return pistonX2;}

else {return pistonX3;}

}

int getPusherNum(int pos) {

if (pos == pistonX5) {

return 3;

}

else if (pos == pistonX1) {

return 4;

}

else if (pos == pistonX2) {

return 5;

}

else {

return 0;

}

}

int getLineNum(int pos) {

if (pos == pistonX5) {

return 5;

}

else if (pos == pistonX1) {

return 7;

}

else if (pos == pistonX2) {

return 9;

}

else {

return 0;

}

}

string getBucketLabel(float mass) {

float d1,d2;

dpGet("System1:A\_source1.deviation.d1", d1);

dpGet("System1:A\_source1.deviation.d2", d2);

float d1L = 1 - d1/100;

float d1R = 1 + d1/100;

float d2L = 1 - d2/100;

float d2R = 1 + d2/100;

if(mass ==0)

{return bucketLabel1;

}else if( mass ==1)

{return bucketLabel2;}

else if ( mass ==2)

{return bucketLabel4;}else

{return bucketLabel3;}

}

int getBucketByMass (float mass) {

float d1,d2;

dpGet("System1:A\_source1.deviation.d1", d1);

dpGet("System1:A\_source1.deviation.d2", d2);

float d1L = 1 - d1/100;

float d1R = 1 + d1/100;

float d2L = 1 - d2/100;

float d2R = 1 + d2/100;

if (mass==0) {

return 1;

} else if (mass==1) {

return 2;

}

else if (mass==2) {

return 4;

}else {

return 3;

}

}

int getDelay() {

int intensity;

dpGet("System1:A\_source1.intensity.value", intensity);

return 60/intensity;

}

void update(){

while(true) {

if(isHand) {

return;

}

if (!isConveyorWorking || isSomeBucketFull()) { continue; }

threadIds = makeDynInt();

for (int i = 1; i <= dynlen(panelNames); i++) {

int id = startThread("updateAt", i);

dynAppend(threadIds, id);

}

for (int i = 1; i <= dynlen(threadIds); i++) {

int id = threadIds[i];

if (dynContains(deletedIds, id)) {

waitThread(id);

}

}

if(isConveyorWorking){

if(convSpeed == 1){

dpSet("System1:conveyor\_dp.speed.value", 1);

} else {

dpSet("System1:conveyor\_dp.speed.value", 1 + convSpeed/10);

}

} else {

dpSet("System1:conveyor\_dp.speed.value", 0);

}

delay(0,20);//

}

}

bool isSomeBucketFull(){

int am1, am2, am3, am4;

dpGet("System1:A\_bucket1.count.value", am1);

dpGet("System1:A\_bucket2.count.value", am2);

dpGet("System1:A\_bucket3.count.value", am3);

dpGet("System1:A\_bucket4.count.value", am4);

if (amount1 == am1) {

fullBucketNumber = 1;

}

else if (amount2 == am2) {

fullBucketNumber = 2;

}

else if (amount3 == am3) {

fullBucketNumber = 4;

}

else if (amount4 == am4) {

fullBucketNumber = 3;

}//

if (fullBucketNumber != 0) {

performOverflowBucket();

return true;

}

else {

return false;

}

}

void performOverflowBucket() {

isConveyorWorking = false;

isCargoGenerating = false;

time t1 = getCurrentTime();

string localTime;

localTime = formatTime("%c", t1, "");

setValue("ToggleSwitch\_ewo1", "checked", false);

setValue("ToggleSwitch\_ewo1", "enabled", false);

dpSet("System1:conveyor\_dp.command.isEnabled", false);

string messageBoxText;

messageBoxText = "корзина " + fullBucketNumber + " полная";

table.appendLine("Дата и время", localTime, "Событие", messageBoxText);

setValue("PUSH\_BUTTON4", "visible", true);

if (isConveyorWorking == false || isCargoGenerating == false){

string dpName = "System1:A\_bucket" + fullBucketNumber + ".count.value";

dpSet(dpName,0);

setValue("LCD" + (fullBucketNumber + 1), "intValue", 0);

isConveyorWorking = true;

isCargoGenerating = true;

fullBucketNumber=0;

setValue("PUSH\_BUTTON4", "visible", false);

dpSet("System1:conveyor\_dp.command.isEnabled", true);

ToggleSwitch\_ewo1.enabled(true);

ToggleSwitch\_ewo1.checked(true);

}

}

void updateAt(int i) {

float x = 0, y = 0, targetX, mass;

string name = panelNames[i];

getValue(name, "position", x, y);

float summary = x +(1.5 + convSpeed/7);

setValue(name, "position", summary, startY);

dpGet(name + ".targetX", targetX);

dpGet(name + ".mass", mass);

if(weightX == x){

dpSet("System1:weighter\_dp.current\_weight.value", mass);

}

if (x >= weightX && x<= weightX + 10 ) {

setValue("LCD1", "value", mass);

delay(0,600);

setValue("LCD1", "value", 0);

}

if (x == targetX) {

int count;

string label = getBucketLabel(mass);

int pushern = getPusherNum(targetX);

int lineNum = getLineNum(targetX);

dynInsertAt(deletedIds, i, 0);

dynRemove(threadIds,i);

if (pushern != 0) {

int psh = startThread("pushRunnable", name, pushern, lineNum, i);

dynAppend(multyThreadsIdis, psh);

} else {

removeSymbol(myModuleName(), myPanelName(), name);

}

dynRemove(panelNames, i);

dpDelete(name);

string dpName = "System1:A\_bucket" + getBucketByMass(mass) + ".count.value";

getValue(label, "intValue", mass);

setValue(label, "intValue", ++mass);

int bucketCount;

dpGet(dpName, bucketCount);

dpSet(dpName, ++bucketCount);

}

}

void pushRunnable(string name, int pusherNum, int lineNum, int i) {

int pusherX, pusherY, lineX, lineY, cargoX, cargoY;

string pusher = "PANEL\_REF" + pusherNum;

string line = "LINE" + lineNum;

int count = 0;

while(count < 45) {

getValue(pusher, "position", pusherX, pusherY);

setValue(pusher, "position", pusherX, ++pusherY);

getValue(line, "position", lineX, lineY);

setValue(line, "position", lineX, ++lineY);

count++;

if(count > 20) {

getValue(name, "position", cargoX, cargoY);

cargoY += 4;

setValue(name, "position", cargoX, cargoY);

}

}

delay(0,50);

removeSymbol(myModuleName(), myPanelName(), name);

while (count > 0) {

getValue(pusher, "position", pusherX, pusherY);

setValue(pusher, "position", pusherX, --pusherY);

getValue(line, "position", lineX, lineY);

setValue(line, "position", lineX, --lineY);

count--;

}

}

void breakerThread() {

dpSet("System1:pusher1.command.push", true);

dpSet("System1:pusher2.command.push", true);

dpSet("System1:pusher3.command.push", true);

dpSet("System1:conveyor\_dp.command.isEnabled", true);

dpSet("System1:weighter\_dp.command.isEnabled", true);

while(true) {

if(isHand) {

return;

}

delay(2.5);

if (isConveyorWorking) {

float weighterThreadCurrent;

float sourceThreadCurrent;

float motorThreadCurrent;

float pusher1ThreadCurrent;

float pusher2ThreadCurrent;

float pusher3ThreadCurrent;

float conveyorThreadCurrent;

dpGet("System1:weighter\_dp.probability.value", weighterThreadCurrent);

dpGet("System1:A\_source1.probability.value", sourceThreadCurrent);

dpGet("System1:A\_motor1.probability.value", motorThreadCurrent);

dpGet("System1:pusher1.probability.value", pusher1ThreadCurrent);

dpGet("System1:pusher2.probability.value", pusher2ThreadCurrent);

dpGet("System1:pusher3.probability.value", pusher3ThreadCurrent);

dpGet("System1:conveyor\_dp.probability.value", conveyorThreadCurrent);

float rand = (float)rand()/32767;

string type;

string messageBoxText;

time t1 = getCurrentTime();

string localTime;

if(rand < weighterThreadCurrent){

dpSet("System1:weighter\_dp.command.isEnabled", false);

dpSet("System1:weighter\_dp.alert.mechanicalDamage", true);

type = "сканер";

localTime = formatTime("%c", t1, "");

messageBoxText = type + " отказал";

table.appendLine("Дата и время", localTime,"Событие", messageBoxText);

isConveyorWorking = false;

isCargoGenerating = false;

dpSet("System1:conveyor\_dp.command.isEnabled", false);

setValue("PUSH\_BUTTON5", "visible", true);

}

rand = (float)rand()/32767;

if(rand < pusher1ThreadCurrent){

dpSet("System1:pusher1.command.push", false);

dpSet("System1:pusher1.alert.mechanicalDamage", true);

type = "толкатель 1";

localTime = formatTime("%c", t1, "");

messageBoxText = type + " отказал";

table.appendLine("Дата и время", localTime,"Событие", messageBoxText);

isConveyorWorking = false;

isCargoGenerating = false;

dpSet("System1:conveyor\_dp.command.isEnabled", false);

setValue("PUSH\_BUTTON5", "visible", true);

}

rand = (float)rand()/32767;

if(rand < pusher2ThreadCurrent){

dpSet("System1:pusher2.command.push", false);

dpSet("System1:pusher2.alert.mechanicalDamage", true);

type = "толкатель 2";

localTime = formatTime("%c", t1, "");

messageBoxText = type + " отказал";

table.appendLine("Дата и время", localTime,"Событие", messageBoxText);

isConveyorWorking = false;

isCargoGenerating = false;

dpSet("System1:conveyor\_dp.command.isEnabled", false);

setValue("PUSH\_BUTTON5", "visible", true);

}

rand = (float)rand()/32767;

if(rand < pusher3ThreadCurrent){

dpSet("System1:pusher3.command.push", false);

dpSet("System1:pusher3.alert.mechanicalDamage", true);

type = "толкатель 3";

localTime = formatTime("%c", t1, "");

messageBoxText = type + " отказал";

table.appendLine("Дата и время", localTime,"Событие", messageBoxText);

isConveyorWorking = false;

isCargoGenerating = false;

dpSet("System1:conveyor\_dp.command.isEnabled", false);

setValue("PUSH\_BUTTON5", "visible", true);

}

rand = (float)rand()/32767;

if(rand < conveyorThreadCurrent){

dpSet("System1:conveyor\_dp.command.isEnabled", false);

dpSet("System1:conveyor\_dp.alert.mechanicalDamage", true);

type = "конвейер";

localTime = formatTime("%c", t1, "");

messageBoxText = type + " отказал";

table.appendLine("Дата и время", localTime,"Событие", messageBoxText);

isConveyorWorking = false;

isCargoGenerating = false;

dpSet("System1:conveyor\_dp.command.isEnabled", false);

setValue("PUSH\_BUTTON5", "visible", true);

}

rand = (float)rand()/32767;

if(rand < sourceThreadCurrent){

dpSet("System1:A\_source1.alert.mechanicalDamage", true);

type = "товаровыпускающее устройство";

localTime = formatTime("%c", t1, "");

messageBoxText = type + " отказал";

table.appendLine("Дата и время", localTime,"Событие", messageBoxText);

isConveyorWorking = false;

isCargoGenerating = false;

dpSet("System1:conveyor\_dp.command.isEnabled", false);

setValue("PUSH\_BUTTON5", "visible", true);

}

rand = (float)rand()/32767;

if(rand < motorThreadCurrent){

dpSet("System1:A\_motor1.command.isEnabled", false);

dpSet("System1:A\_motor1.alert.damage", true);

type = "приводное устройство";

localTime = formatTime("%c", t1, "");

messageBoxText = type + " отказал";

table.appendLine("Дата и время", localTime,"Событие", messageBoxText);

isConveyorWorking = false;

isCargoGenerating = false;

dpSet("System1:conveyor\_dp.command.isEnabled", false);

setValue("PUSH\_BUTTON5", "visible", true);

}

if(isConveyorWorking == false || isCargoGenerating == false){

delay(2);

dpSet("System1:weighter\_dp.command.isEnabled", true);

dpSet("System1:weighter\_dp.alert.mechanicalDamage", false);

dpSet("System1:pusher1.command.push", true);

dpSet("System1:pusher1.alert.mechanicalDamage", false);

dpSet("System1:pusher2.command.push", true);

dpSet("System1:pusher2.alert.mechanicalDamage", false);

dpSet("System1:pusher3.command.push", true);

dpSet("System1:pusher3.alert.mechanicalDamage", false);

dpSet("System1:conveyor\_dp.command.isEnabled", true);

dpSet("System1:conveyor\_dp.alert.mechanicalDamage", false);

dpSet("System1:A\_motor1.command.isEnabled", true);

dpSet("System1:A\_motor1.alert.damage", false);

string dpName = "System1:A\_bucket" + fullBucketNumber + ".count.value";

dpSet(dpName,0);

setValue("LCD" + (fullBucketNumber + 1), "intValue", 0);

isConveyorWorking = true;

isCargoGenerating = true;

dpSet("System1:conveyor\_dp.command.isEnabled", true);

fullBucketNumber=0;

setValue("PUSH\_BUTTON5", "visible", false);

ToggleSwitch\_ewo1.enabled(true);

ToggleSwitch\_ewo1.checked(true);

}

}

}

}

float getNormDistribValue(float aa, float bb) {

float sum =0,x;

for(int i=0;i<8;i++)

sum+=1.0\*rand()/RAND\_MAX;

x= (sqrt(2.0)\*(bb)\*(sum-6.95))/2.0+aa;

if(x<0){x=x\*(-1);}

return floor(x);

}

══════════════════════════════════════════════════════

─// [PUSH\_BUTTON11] [23] - [Clicked]

dyn\_string dpNames;

dyn\_int deletedIds;

int startY = 130;

int startX = 40;

int pistonX5 = 440;

int pistonX1 = 580;

int pistonX3 = 780;

int pistonX4 = 855;

int pistonX2 = 710;

int weightX = 225;

string bucketLabel1 = "LCD2";

string bucketLabel2 = "LCD3";

string bucketLabel3 = "LCD4";

string bucketLabel4 = "LCD5";

mapping cargosToPush;

int amount1, amount2, amount3, amount4;

shape table = getShape("TABLE1");

main()

{

ToggleSwitch\_ewo1.visible = true;

PRIMITIVE\_TEXT62.visible = TRUE;

PRIMITIVE\_TEXT62.text = "Ручной режим запущен";

PANEL\_REF22.position(28,60);

PANEL\_REF21.position(113,213);

PANEL\_REF20.position(264,59);

PANEL\_REF18.position(477,42);

PANEL\_REF19.position(611,42);

PANEL\_REF25.position(745,42);//

isHand = TRUE;

isAuto = FALSE;

reset();

init();

runThreads();

}

void init() {

setValue("ToggleSwitch\_ewo1", "checked", true);

dpNames = makeDynString();

threadNames = makeDynString();

dpSet("System1:A\_bucket1.count.value", 0);

dpSet("System1:A\_bucket2.count.value", 0);

dpSet("System1:A\_bucket3.count.value", 0);

dpSet("System1:A\_bucket4.count.value", 0);

}

void runThreads(){

for (int i = 1; i <= dynlen(multyThreadsIdis); i++) {

DebugN("Th " + multyThreadsIdis[i]);

stopThread(multyThreadsIdis[i]);

waitThread(multyThreadsIdis[i]);

dynRemove(multyThreadsIdis, i);

}

int upt = startThread("update");

int gen = startThread("generateCargos");

int br = startThread("breakerThread");

dynAppend(multyThreadsIdis, upt);

dynAppend(multyThreadsIdis, gen);

dynAppend(multyThreadsIdis, br);

}

void updateMass(){

float mass1, mass2, mass3,mass4;

dpGet("System1:A\_bucket1.max\_value", amount1);

dpGet("System1:A\_bucket2.max\_value", amount2);

dpGet("System1:A\_bucket3.max\_value", amount3);

dpGet("System1:A\_bucket4.max\_value", amount4);

dpGet("System1:A\_bucket1.max\_value", mass1);

dpGet("System1:A\_bucket2.max\_value", mass2);

dpGet("System1:A\_bucket3.max\_value", mass3);

dpGet("System1:A\_bucket4.max\_value", mass4);

setValue("PRIMITIVE\_TEXT30", "text", mass1);

setValue("PRIMITIVE\_TEXT34", "text", mass2);

setValue("PRIMITIVE\_TEXT32", "text", mass3);

dpGet("System1:m1.", m1);

dpGet("System1:m2.", m2);

dpGet("System1:m3.", m3);

dpGet("System1:m4.", m4);

}

void generateCargos(){

int id = 0;

while(true) {

if(isAuto) {

return;

}

bool sour;

bool motor;

dpGet("System1:conveyor\_dp.command.isEnabled",motor);

dpGet("System1:conveyor\_dp.command.isEnabled",sour);

if (!isCargoGenerating || !sour || !motor) { continue; }

updateMass();

generateCargoWithID(id++);

int del = getDelay() \* 1000;

delay(0, del);

}

}

void generateCargoWithID(int id) {

string name = "CARGO" + id;

dpCreate(name, "A\_cargo");

float mass = generateMass();

int targetX = getTargetX(mass);

dpSet(name + ".mass", mass);

dpSet(name + ".targetX", targetX);

string weightLabel = "$weight:";

string colorStr = "$color:";

int color;

float d1,d2;

dpGet("System1:A\_source1.deviation.d1", d1);

dpGet("System1:A\_source1.deviation.d2", d2);

float d1L = 1 - d1/100;

float d1R = 1 + d1/100;

float d2L = 1 - d2/100;

float d2R = 1 + d2/100;

if (mass ==0) {color = 1;}

else if ( mass ==1) {color = 2;}

else if (mass==2) {color = 3;}

else {color = 4;}

int someInt = getBucketByMass(mass);

switch(color) {

case 1:

colorStr = colorStr + "STD\_value\_invalid";

break;

case 2:

colorStr = colorStr + "STD\_valve\_open";

break;

case 3:

colorStr = colorStr + "Yellow";

break;

case 4:

colorStr = colorStr + "Black";

break;

}

addSymbol(myModuleName(), myPanelName(), "gedi/Cargo.pnl",name,makeDynString(weightLabel, colorStr),startX, startY,0,0,0);

dynAppend(panelNames, name);

}

float generateMass(){

int dist;

dpGet("System1:mass.determ",determMass);

dpGet("System1:mass.type",dist);

dpGet("System1:mass.rand.norm.mu",expectedValue);

dpGet("System1:mass.rand.norm.sigma",dispSqr);

dpGet("System1:mass.rand.ravnomer.a",a);

dpGet("System1:mass.rand.ravnomer.b",b);

dpGet("System1:mass.rand.exp",lamb);

switch(dist) {

case 0:

return determMass;

case 1 :

return getNormDistribValue(expectedValue,dispSqr);

}

}

int getTargetX(float mass) {

float d1,d2;

bool p1,p2,p3;

dpGet("System1:A\_source1.deviation.d1", d1);

dpGet("System1:A\_source1.deviation.d2", d2);

dpGet("System1:pusher1.command.push", p1);

dpGet("System1:pusher2.command.push", p2);

dpGet("System1:pusher3.command.push", p3);

bool we;

dpGet("System1:weighter\_dp.command.isEnabled", we);

if (mass==0) {return pistonX5;}

else if ( mass==1){return pistonX1;}

else if(mass==2){return pistonX2;}

else {return pistonX3;}}

int getPusherNum(int pos) {

if (pos == pistonX5) {

return 3;

}

else if (pos == pistonX1) {

return 4;

}

else if (pos == pistonX2) {

return 5;

}

else {

return 0;

}

}

int getLineNum(int pos) {

if (pos == pistonX5) {

return 5;

}

else if (pos == pistonX1) {

return 7;

}

else if (pos == pistonX2) {

return 9;

}

else {

return 0;

}

}

string getBucketLabel(float mass) {

float d1,d2;

bool p1,p2,p3;

dpGet("System1:A\_source1.deviation.d1", d1);

dpGet("System1:A\_source1.deviation.d2", d2);

dpGet("System1:pusher1.command.push", p1);

dpGet("System1:pusher2.command.push", p2);

dpGet("System1:pusher3.command.push", p3);

bool we;

dpGet("System1:weighter\_dp.command.isEnabled", we);

if (mass==0){return bucketLabel1;} else if

(mass==1){return bucketLabel2;} else if

(mass==2){return bucketLabel4;}

else { return bucketLabel3;}

}

int getBucketByMass (float mass) {

float d1,d2;

bool p1,p2,p3;

dpGet("System1:A\_source1.deviation.d1", d1);

dpGet("System1:A\_source1.deviation.d2", d2);

dpGet("System1:pusher1.command.push", p1);

dpGet("System1:pusher2.command.push", p2);

dpGet("System1:pusher3.command.push", p3);

bool we;

dpGet("System1:weighter\_dp.command.isEnabled", we);

if (mass==0){return 1;} else if

(mass==1){return 2;} else if

(mass==2){return 4;}

else { return 3;}

}

int getDelay() {

int intensity;

dpGet("System1:A\_source1.intensity.value", intensity);

return 60/intensity;

}

void update(){

while(true) {

if(isAuto){

return;

}

bool conv;

bool motor;

dpGet("System1:conveyor\_dp.command.isEnabled", conv);

dpGet("System1:A\_motor1.command.isEnabled", motor);

if(isConveyorWorking && motor){

if(convSpeed == 1){

dpSet("System1:conveyor\_dp.speed.value", 1);

} else {

dpSet("System1:conveyor\_dp.speed.value", 1 + convSpeed/10);

}

} else {

dpSet("System1:conveyor\_dp.speed.value", 0);

DebugN(motor);

}

if ((!isConveyorWorking || !motor) || isSomeBucketFull()) { continue; }

threadIds = makeDynInt();

for (int i = 1; i <= dynlen(panelNames); i++) {

int id = startThread("updateAt", i);

dynAppend(threadIds, id);

}

for (int i = 1; i <= dynlen(threadIds); i++) {

int id = threadIds[i];

if (dynContains(deletedIds, id)) {

waitThread(id);

}

}

delay(0,20);

}

}

bool isSomeBucketFull(){

int am1, am2, am3,am4;

dpGet("System1:A\_bucket1.count.value", am1);

dpGet("System1:A\_bucket2.count.value", am2);

dpGet("System1:A\_bucket3.count.value", am3);

dpGet("System1:A\_bucket4.count.value", am4);

if (amount1 == am1) {

fullBucketNumber = 1;

}

else if (amount2 == am2) {

fullBucketNumber = 2;

}

else if (amount3 == am3) {

fullBucketNumber = 3;

}

else if (amount4 == am4) {

fullBucketNumber = 4;

}

if (fullBucketNumber != 0) {

performOverflowBucket();

return true;

}

else {

return false;

}

}

void performOverflowBucket() {

isConveyorWorking = false;

isCargoGenerating = false;

time t1 = getCurrentTime();

string localTime;

localTime = formatTime("%c", t1, "");

setValue("ToggleSwitch\_ewo1", "checked", false);

setValue("ToggleSwitch\_ewo1", "enabled", false);

dpSet("System1:conveyor\_dp.command.isEnabled", false);

string messageBoxText;

messageBoxText = "корзина " + fullBucketNumber + " полная";

table.appendLine("Дата и время", localTime, "Событие", messageBoxText);

if(fullBucketNumber == 1){

setValue("PUSH\_BUTTON18", "visible", true);

}

if(fullBucketNumber == 2){

setValue("PUSH\_BUTTON19", "visible", true);

}

if(fullBucketNumber == 3){

setValue("PUSH\_BUTTON20", "visible", true);

}

if(fullBucketNumber == 4){

setValue("PUSH\_BUTTON26", "visible", true);

}

}

void updateAt(int i) {

float x = 0, y = 0, targetX, mass;

string name = panelNames[i];

getValue(name, "position", x, y);

float summary = x +(1.5 + convSpeed/7);

setValue(name, "position",summary, startY);

dpGet(name + ".targetX", targetX);

dpGet(name + ".mass", mass);

bool we;

dpGet("System1:weighter\_dp.command.isEnabled", we);

if(weightX == x){

dpSet("System1:weighter\_dp.current\_weight.value", mass);

}

if ((x >= weightX && x<= weightX + 10 )&& we) {

setValue("LCD1", "value", mass);

delay(0,600);

setValue("LCD1", "value", 0);

}

if (x == targetX) {

int count;

string label = getBucketLabel(mass);

int pushern = getPusherNum(targetX);

int lineNum = getLineNum(targetX);

dynInsertAt(deletedIds, i, 0);

dynRemove(threadIds,i);

if (pushern != 0) {

int psh = startThread("pushRunnable", name, pushern, lineNum, i);

dynAppend(multyThreadsIdis, psh);

} else {

removeSymbol(myModuleName(), myPanelName(), name);

}

dynRemove(panelNames, i);

dpDelete(name);

string dpName = "System1:A\_bucket" + getBucketByMass(mass) + ".count.value";

getValue(label, "intValue", mass);

setValue(label, "intValue", ++mass);

int bucketCount;

dpGet(dpName, bucketCount);

dpSet(dpName, ++bucketCount);

}

}

void pushRunnable(string name, int pusherNum, int lineNum, int i) {

int pusherX, pusherY, lineX, lineY, cargoX, cargoY;

string pusher = "PANEL\_REF" + pusherNum;

string line = "LINE" + lineNum;

int count = 0;

while(count < 45) {

getValue(pusher, "position", pusherX, pusherY);

setValue(pusher, "position", pusherX, ++pusherY);

getValue(line, "position", lineX, lineY);

setValue(line, "position", lineX, ++lineY);

count++;

if(count > 20) {

getValue(name, "position", cargoX, cargoY);

cargoY += 4;

setValue(name, "position", cargoX, cargoY);

}

}

delay(0,50);

removeSymbol(myModuleName(), myPanelName(), name);

while (count > 0) {

getValue(pusher, "position", pusherX, pusherY);

setValue(pusher, "position", pusherX, --pusherY);

getValue(line, "position", lineX, lineY);

setValue(line, "position", lineX, --lineY);

count--;

}

}

void breakerThread() {

dpSet("System1:pusher1.command.push", true);

dpSet("System1:pusher2.command.push", true);

dpSet("System1:pusher3.command.push", true);

dpSet("System1:conveyor\_dp.command.isEnabled", true);

dpSet("System1:weighter\_dp.command.isEnabled", true);

while(true) {

if(isAuto){

return;

}

delay(2.5);

bool motor;

dpGet("System1:A\_motor1.command.isEnabled", motor);

if (isConveyorWorking && motor) {

float weighterThreadCurrent;

float sourceThreadCurrent;

float motorThreadCurrent;

float pusher1ThreadCurrent;

float pusher2ThreadCurrent;

float pusher3ThreadCurrent;

float conveyorThreadCurrent;

dpGet("System1:weighter\_dp.probability.value", weighterThreadCurrent);

dpGet("System1:A\_source1.probability.value", sourceThreadCurrent);

dpGet("System1:A\_motor1.probability.value", motorThreadCurrent);

dpGet("System1:pusher1.probability.value", pusher1ThreadCurrent);

dpGet("System1:pusher2.probability.value", pusher2ThreadCurrent);

dpGet("System1:pusher3.probability.value", pusher3ThreadCurrent);

dpGet("System1:conveyor\_dp.probability.value", conveyorThreadCurrent);

bool p1,p2,p3;

dpGet("System1:pusher1.command.push", p1);

dpGet("System1:pusher2.command.push", p2);

dpGet("System1:pusher3.command.push", p3);

float rand = (float)rand()/32767;

string type;

string messageBoxText;

time t1 = getCurrentTime();

string localTime;

if(rand < weighterThreadCurrent){

dpSet("System1:weighter\_dp.command.isEnabled", false);

dpSet("System1:A\_motor1.command.isEnabled", false);

dpSet("System1:weighter\_dp.alert.mechanicalDamage", true);

type = "сканер";

localTime = formatTime("%c", t1, "");

messageBoxText = type + " отказал";

table.appendLine("Дата и время", localTime,"Событие", messageBoxText);

isConveyorWorking = false;

isCargoGenerating = false;

setValue("ToggleSwitch\_ewo1", "checked", false);

setValue("ToggleSwitch\_ewo1", "enabled", false);

dpSet("System1:conveyor\_dp.command.isEnabled", false);

dynAppend(threadNames, "We");

setValue("PUSH\_BUTTON22", "visible", true);

}

rand = (float)rand()/32767;

if((rand < pusher1ThreadCurrent) && p1){

dpSet("System1:pusher1.command.push", false);

dpSet("System1:A\_motor1.command.isEnabled", false);

dpSet("System1:pusher1.alert.mechanicalDamage", true);

type = "толкатель 1";

localTime = formatTime("%c", t1, "");

messageBoxText = type + " отказал";

table.appendLine("Дата и время", localTime,"Событие", messageBoxText);

isConveyorWorking = false;

isCargoGenerating = false;

setValue("ToggleSwitch\_ewo1", "checked", false);

setValue("ToggleSwitch\_ewo1", "enabled", false);

dpSet("System1:conveyor\_dp.command.isEnabled", false);

dynAppend(threadNames, "P1");

setValue("PUSH\_BUTTON15", "visible", true);

}

rand = (float)rand()/32767;

if((rand < pusher2ThreadCurrent)&& p2){

dpSet("System1:pusher2.command.push", false);

dpSet("System1:A\_motor1.command.isEnabled", false);

dpSet("System1:pusher2.alert.mechanicalDamage", true);

type = "толкатель 2";

localTime = formatTime("%c", t1, "");

messageBoxText = type + " отказал";

table.appendLine("Дата и время", localTime,"Событие", messageBoxText);

isConveyorWorking = false;

isCargoGenerating = false;

setValue("ToggleSwitch\_ewo1", "checked", false);

setValue("ToggleSwitch\_ewo1", "enabled", false);

dpSet("System1:conveyor\_dp.command.isEnabled", false);

dynAppend(threadNames, "P2");

setValue("PUSH\_BUTTON16", "visible", true);

}

rand = (float)rand()/32767;

if((rand < pusher3ThreadCurrent) && p3){

dpSet("System1:pusher3.command.push", false);

dpSet("System1:A\_motor1.command.isEnabled", false);

dpSet("System1:pusher3.alert.mechanicalDamage", true);

type = "толкатель 3";

localTime = formatTime("%c", t1, "");

messageBoxText = type + " отказал";

table.appendLine("Дата и время", localTime,"Событие", messageBoxText);

isConveyorWorking = false;

isCargoGenerating = false;

setValue("ToggleSwitch\_ewo1", "checked", false);

setValue("ToggleSwitch\_ewo1", "enabled", false);

dpSet("System1:conveyor\_dp.command.isEnabled", false);

dynAppend(threadNames, "P3");

setValue("PUSH\_BUTTON25", "visible", true);

}

rand = (float)rand()/32767;

if(rand < conveyorThreadCurrent){

dpSet("System1:conveyor\_dp.command.isEnabled", false);

dpSet("System1:A\_motor1.command.isEnabled", false);

dpSet("System1:conveyor\_dp.alert.mechanicalDamage", true);

type = "конвейер";

localTime = formatTime("%c", t1, "");

messageBoxText = type + " отказал";

table.appendLine("Дата и время", localTime,"Событие", messageBoxText);

isConveyorWorking = false;

isCargoGenerating = false;

setValue("ToggleSwitch\_ewo1", "checked", false);

setValue("ToggleSwitch\_ewo1", "enabled", false);

dpSet("System1:conveyor\_dp.command.isEnabled", false);

dynAppend(threadNames, "Conv");

setValue("PUSH\_BUTTON23", "visible", true);

}

rand = (float)rand()/32767;

if(rand < sourceThreadCurrent){

dpSet("System1:A\_source1.alert.mechanicalDamage", true);

dpSet("System1:A\_motor1.command.isEnabled", false);

type = "товаровыпускающее устройство";

localTime = formatTime("%c", t1, "");

messageBoxText = type + " отказал";

table.appendLine("Дата и время", localTime,"Событие", messageBoxText);

isConveyorWorking = false;

isCargoGenerating = false;

setValue("ToggleSwitch\_ewo1", "checked", false);

setValue("ToggleSwitch\_ewo1", "enabled", false);

dpSet("System1:conveyor\_dp.command.isEnabled", false);

dynAppend(threadNames, "Sour");

setValue("PUSH\_BUTTON21", "visible", true);

}

rand = (float)rand()/32767;

if(rand < motorThreadCurrent){

dpSet("System1:A\_motor1.command.isEnabled", false);

dpSet("System1:A\_motor1.alert.damage", true);

type = "приводное устройство";

localTime = formatTime("%c", t1, "");

messageBoxText = type + " отказал";

table.appendLine("Дата и время", localTime,"Событие", messageBoxText);

isConveyorWorking = false;

isCargoGenerating = false;

setValue("ToggleSwitch\_ewo1", "checked", false);

setValue("ToggleSwitch\_ewo1", "enabled", false);

dpSet("System1:conveyor\_dp.command.isEnabled", false);

dynAppend(threadNames, "Mot");

setValue("PUSH\_BUTTON17", "visible", true);

}

}

}

}

float getNormDistribValue(float aa, float bb) {

float sum =0,x;

for(int i=0;i<8;i++)

sum+=1.0\*rand()/RAND\_MAX;

x= (sqrt(2.0)\*(bb)\*(sum-6.95))/2.0+aa;

if(x<0){x=x\*(-1);}

return floor(x);

}

══════════════════════════════════════════════════════

─// [TAB1] [26] - [Initialize]

main()

{

if(!getUserPermission(6)){

TAB1.transparentForMouse(true);

langString f = "Arial Black,8,-1,5,50,0,0,0,0,0";

this.color("grey");

this.font(f);

DebugN("authorization for level 4 ok");}

else DebugN("no authorization for level");

}

══════════════════════════════════════════════════════

─// [PUSH\_BUTTON12] [29] - [Clicked]

main()

{

EP\_childPanelOn();

}

void EP\_childPanelOn()

{

ChildPanelOnCentralModal("gedi/process\_trends.pnl",

"График",

makeDynString(""));

}

══════════════════════════════════════════════════════

─// [PUSH\_BUTTON13] [32] - [Clicked]

main()

{

EP\_childPanelOn();

}

void EP\_childPanelOn()

{

ChildPanelOnCentralModal("gedi/DataView.pnl", "Архив точек данных", makeDynString(""));

}

══════════════════════════════════════════════════════

─// [PUSH\_BUTTON14] [33] - [Clicked]

main()

{

EP\_childPanelOn();

}

void EP\_childPanelOn()

{

ChildPanelOnCentralModal("gedi/AboutProgramm.pnl",

"О разработчиках",

makeDynString(""));

}

══════════════════════════════════════════════════════

─// [PUSH\_BUTTON15] [34] - [Clicked]

main(mapping event)

{

dpSet("System1:pusher1.alert.mechanicalDamage", false);

for (int i = 1; i <= dynlen(threadNames); i++) {

if(threadNames[i] == "P1"){

dynRemove(threadNames, i);

}

}

if(dynlen(threadNames)==0){

ToggleSwitch\_ewo1.enabled(true);

ToggleSwitch\_ewo1.checked(true);

}

setValue("PUSH\_BUTTON15", "visible", false);

}

══════════════════════════════════════════════════════

─// [PUSH\_BUTTON16] [35] - [Clicked]

main(mapping event)

{

dpSet("System1:pusher2.alert.mechanicalDamage", false);

for (int i = 1; i <= dynlen(threadNames); i++) {

if(threadNames[i] == "P2"){

dynRemove(threadNames, i);

}

}

if(dynlen(threadNames)==0){

ToggleSwitch\_ewo1.enabled(true);

ToggleSwitch\_ewo1.checked(true);

}

setValue("PUSH\_BUTTON16", "visible", false);

}

══════════════════════════════════════════════════════

─// [PUSH\_BUTTON17] [36] - [Clicked]

main(mapping event)

{

dpSet("System1:A\_motor1.alert.damage", false);

for (int i = 1; i <= dynlen(threadNames); i++) {

if(threadNames[i] == "Mot"){

dynRemove(threadNames, i);

}

}

if(dynlen(threadNames)==0){

ToggleSwitch\_ewo1.enabled(true);

ToggleSwitch\_ewo1.checked(true);

}

setValue("PUSH\_BUTTON17", "visible", false);

}

══════════════════════════════════════════════════════

─// [PUSH\_BUTTON18] [37] - [Clicked]

main()

{

dpSet("System1:A\_bucket1.count.value", 0);

setValue("LCD2", "intValue", 0);

}

══════════════════════════════════════════════════════

─// [PUSH\_BUTTON19] [38] - [Clicked]

main(mapping event)

{

dpSet("System1:A\_bucket2.count.value", 0);

setValue("LCD3", "intValue", 0);

}

══════════════════════════════════════════════════════

─// [PUSH\_BUTTON20] [39] - [Clicked]

main()

{

dpSet("System1:A\_bucket3.count.value", 0);

setValue("LCD4", "intValue", 0);

}

══════════════════════════════════════════════════════

─// [PUSH\_BUTTON22] [40] - [Clicked]

main(mapping event)

{

dpSet("System1:weighter\_dp.alert.mechanicalDamage", false);

for (int i = 1; i <= dynlen(threadNames); i++) {

if(threadNames[i] == "We"){

dynRemove(threadNames, i);

}

}

if(dynlen(threadNames)==0){

ToggleSwitch\_ewo1.enabled(true);

ToggleSwitch\_ewo1.checked(true);

}

setValue("PUSH\_BUTTON22", "visible", false);

}

══════════════════════════════════════════════════════

─// [PUSH\_BUTTON23] [41] - [Clicked]

main(mapping event)

{

dpSet("System1:conveyor\_dp.alert.mechanicalDamage", false);

for (int i = 1; i <= dynlen(threadNames); i++) {

if(threadNames[i] == "Conv"){

dynRemove(threadNames, i);

}

}

if(dynlen(threadNames)==0){

ToggleSwitch\_ewo1.enabled(true);

ToggleSwitch\_ewo1.checked(true);

}

setValue("PUSH\_BUTTON23", "visible", false);

}

══════════════════════════════════════════════════════

─// [PUSH\_BUTTON24] [43] - [Clicked]

main()

{

EP\_childPanelOn();

}

void EP\_childPanelOn()

{

ChildPanelOnCentralModal("gedi/AboutProg.pnl",

"О системе",

makeDynString(""));

}

══════════════════════════════════════════════════════

─// [PUSH\_BUTTON25] [46] - [Clicked]

main(mapping event)

{

dpSet("System1:pusher3.alert.mechanicalDamage", false);

for (int i = 1; i <= dynlen(threadNames); i++) {

if(threadNames[i] == "P3"){

dynRemove(threadNames, i);

}

}

if(dynlen(threadNames)==0){

ToggleSwitch\_ewo1.enabled(true);

ToggleSwitch\_ewo1.checked(true);

}

setValue("PUSH\_BUTTON25", "visible", false);

}

══════════════════════════════════════════════════════

─// [LCD5] [47] - [Initialize]

main()

{

float n;

dpGet("System1:A\_bucket4.count.value", n);

if(n==600.0){

dpSet("System1:A\_bucket4.count.value", 0.0);

setValue("LCD5", "intValue", 0.0);}

}

══════════════════════════════════════════════════════

─// [PUSH\_BUTTON21] [107] - [Clicked]

main(mapping event)

{

dpSet("System1:A\_source1.alert.mechanicalDamage", false);

for (int i = 1; i <= dynlen(threadNames); i++) {

if(threadNames[i] == "Sour"){

dynRemove(threadNames, i);

}

}

if(dynlen(threadNames)==0){

ToggleSwitch\_ewo1.enabled(true);

ToggleSwitch\_ewo1.checked(true);

}

setValue("PUSH\_BUTTON21", "visible", false);

}

ПРИЛОЖЕНИЕ В  
Руководство пользователя

А.1 Назначение системы

Система предназначена для моделирования процесса регистрации различной продукции. Администратор:

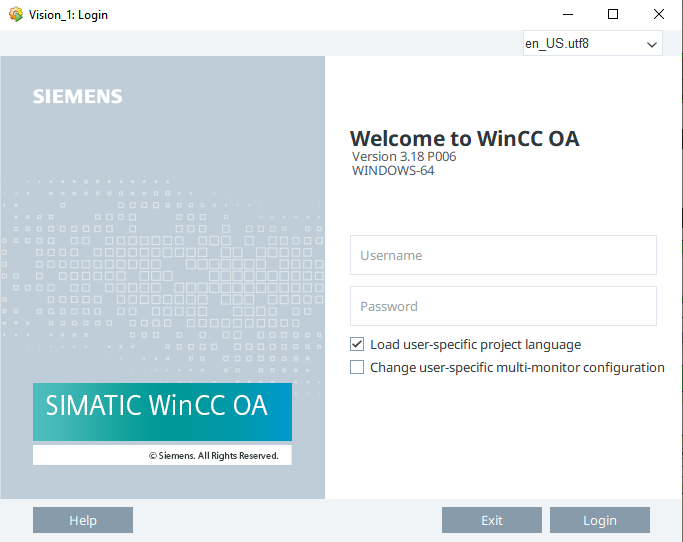
* может изменять параметры элементов системы
* имеет право запускать автоматический режим для проверки работы при этих параметрах, также может запускать ручной режим
* ремонтировать элементы при нажатии на специальную кнопку на элементе системы
* очищать корзину при переполнении нажатием на кнопку на выбранной корзине

Администратор и оператор могут квитировать ошибки, просматривать исторические данные, фильтровать эти данные по промежуткам времени.

А.2 Окно авторизации

При запуске система отобразит окно авторизации (рисунок А.1).

Рисунок А.1 – Экранная форма для окна авторизации в системе

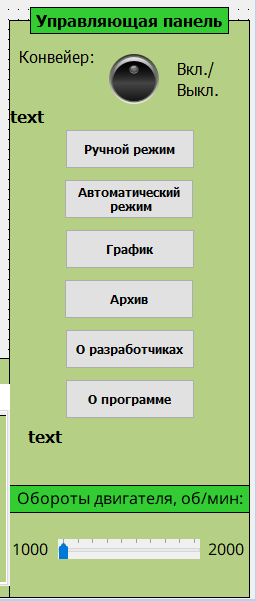


Для авторизации необходимо ввести логин (k\_operator/k\_admin) и пароль (Main\_123), после чего нажать на кнопку "Login". При входе пользователь может выбрать опцию использования его специфичного языка или загрузить настройки монитора пользователя. Также пользователь может воспользоваться помощью или выйти из системы.

А.3 Управляющая панель

Рисунок А.2 описывает управляющую панель.

Рисунок А.2 – Управляющая панель



Основные функции:

* Для того чтобы запустить ручной режим, необходимо нажать на кнопку "Ручной режим".
* Для автоматического режима "Автоматический режим".
* При нажатии "График" отобразится график зависимости параметров от времени, изображенный на рисунке 3.
* При нажатии "Архив" откроется окно с архивом исторических данных, изображенный на рисунке 4.
* При нажатии "О программе" откроется текущее руководство
* При нажатии "О разработчиках" выйдет окно, изображенное на рисунке 5

Также ниже располагается ползунок, позволяющий Администратору менять скорость ленты конвейера по средствам изменения мощности приводного устройства.

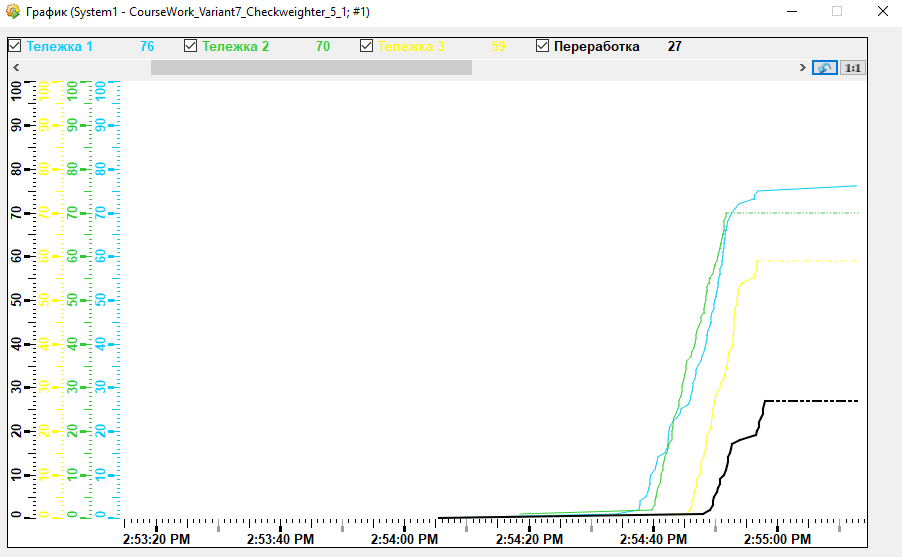
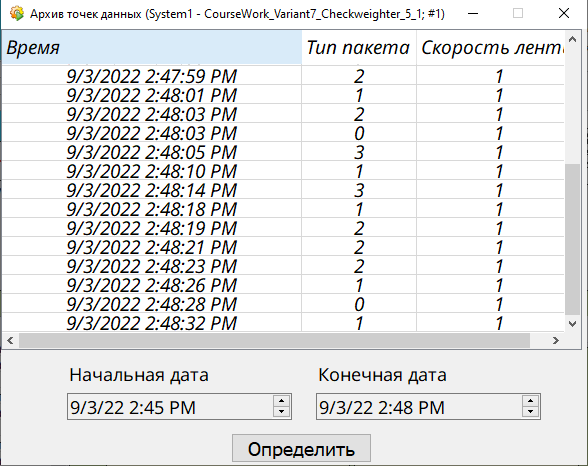
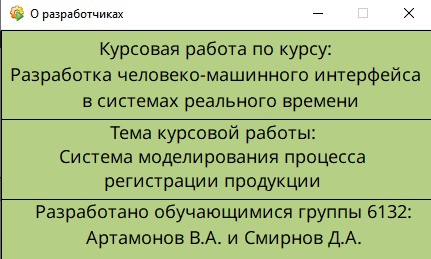
  
Рисунок А.3 – Окно «График»

Рисунок А.4 - Окно «Архив»



А.4 Панель настроек

Рисунок А.5 - Окно «О разработчиках»



На изображенных ниже рисунках А.6-7 отображены режимы задания потока изделий в панели настроек.

* Случайный

Поток товаров будет идти случайным образом, с заданной интенсивностью, которая задается в соответствующем поле

* Детерминированный режим

Поток товаров будет идти предопределенно, с заданной интенсивностью, которая задается в соответствующем поле

В верхней вкладке задается тип изделий, где:

* 0 тип - синий пакет;
* 1тип - зеленый пакет;
* 2тип - желтый пакет.

Рисунок А.6 - Панель настроек (Поток изделий: Случайный режим)

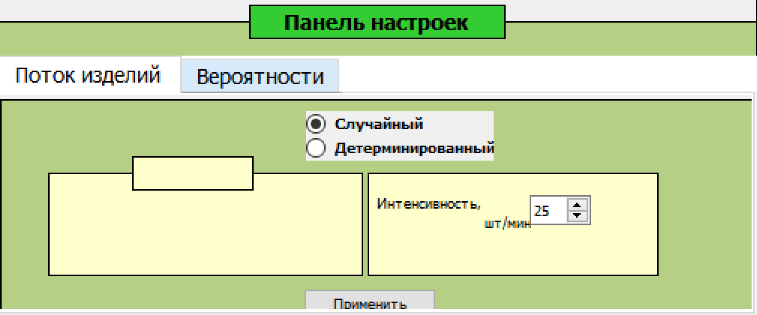
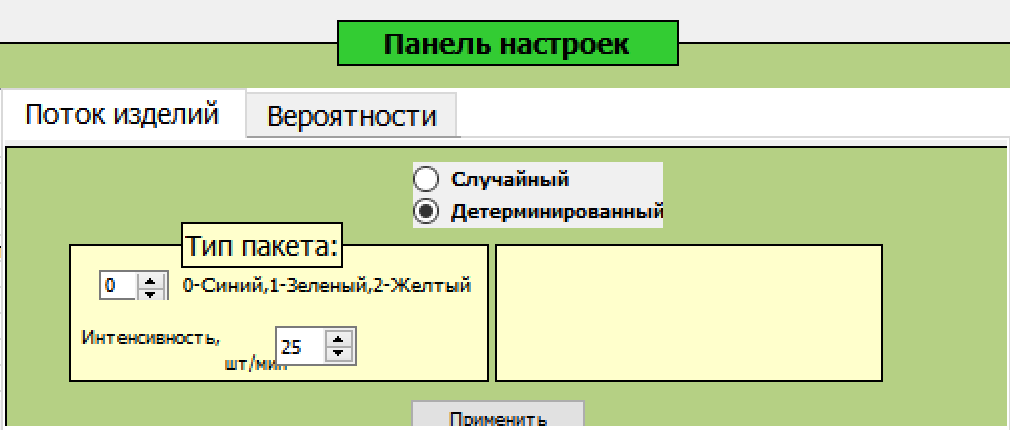


Рисунок А.7 - Панель настроек (Поток изделий: Детерминированный режим)

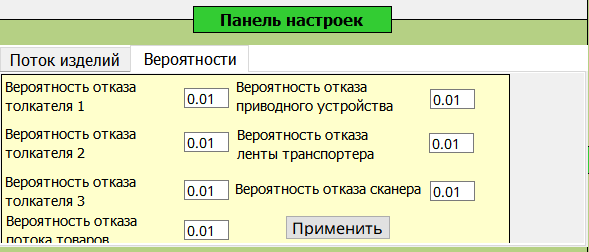


На рисунке А.8 представлена вкладка для настройки параметров вероятности отказа. Можно изменить вероятность отказа:

* Сталкивателей 1-3.
* Потока товаров.
* Приводного устройства.
* Ленты транспортера.
* Сканера.

Для сохранения изменений следует нажать «Применить».

Рисунок А.8 - Панель настроек (Вероятности)



А.5 Описание работы мнемосхемы в ручном режиме

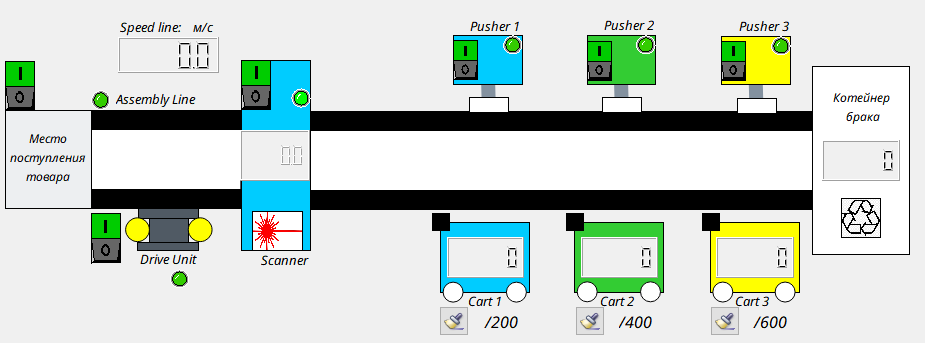
На рисунке А.9 представлена экранная форма для работы мнемосхемы в ручном режиме. Для того чтобы починить сломавшийся элемент системы необходимо нажать на красно-зеленые диоды показывают работоспособность элемента. Для того чтобы включить/выключить элемент системы используются переключатели слева от каждого из устройств.

Циферблат на Сканере отображает тип продукции:

* 0.0 - первый тип продукции (синий пакет)
* 1.0 - второй тип продукции (зеленый пакет)
* 2.0 - третий тип продукции (желтый пакет)
* 3.0 - брак на производстве (черный пакет)

**«**Speed line: м/с» - отображает действительную скорость ленты транспортера. Циферблаты на «Cart 1-3» - отображают текущее количество товаров в тележках. А циферблат «Контейнер брака» отображает количество брака.

Рисунок А.9 – Экранная форма для ручного режима



Не содержит ни одного элемента, с которым возможно взаимодействовать, точки данных чинятся сами по себе.

А.6 Описание работы мнемосхемы в автоматическом режиме

Рисунок А.10 - Экранная форма для автоматического режима

