МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«НОВОСИБИРСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
(НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ, НГУ)

Факультет информационных технологий
Кафедра компьютерных технологий

Направление подготовки 09.03.01 Информатика и вычислительная техника
Направленность (профиль): Программная инженерия и компьютерные науки

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА БАКАЛАВРА

Харченко Александра Дмитриевича

Тема работы:

РАЗРАБОТКА ИНТЕРФЕЙСА ПОЛЬЗОВАТЕЛЯ В ОБЛАЧНОМ ПРАКТИКУМЕ ПО ЯЗЫКУ POST

«К защите допущена»
Заведующий кафедрой,
д.т.н., доцент
Зюбин В. Е. /……..
(ФИО) / (подпись)
31 мая 2024г.

Руководитель ВКР
д.т.н., доцент,
Зав. КафКТ ФИТ НГУ
Зюбин В. Е. /……...
(ФИО) / (подпись)
31 мая 2024г.

Соруководитель ВКР
к.ф.-м.н.,
доцент КафСИ ФИТ НГУ
Ануреев И. С. /……..
(ФИО) / (подпись)
31 мая 2024г.

Новосибирск, 2024

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«НОВОСИБИРСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
(НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ, НГУ)

Факультет информационных технологий

Кафедра компьютерных технологий
(название кафедры)

Направление подготовки 09.03.01 Информатика и вычислительная техника

Направленность (профиль): Программная инженерия и компьютерные науки

УТВЕРЖДАЮ

Зав. кафедрой Зюбин В.Е.
(фамилия, И., О.)

…….
(подпись)

31 октября 2023г.

ЗАДАНИЕ
НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ БАКАЛАВРА

Студенту Харченко Александру Дмитриевичу, группы 20201
(фамилия, имя, отчество, номер группы)

Тема: Разработка интерфейса пользователя в облачном практикуме по языку poST
(полное название темы выпускной квалификационной работы)

утверждена распоряжением проректора по учебной работе от 31 октября 2023г. №0394

Срок сдачи студентом готовой работы 20 мая 2024 г.

Исходные данные (или цель работы): разработать интерфейс программиста алгоритма управления в облачном практикуме по языку poST.

Структурные части работы:

Анализ предметной области, изучение интерфейсов существующих программ для программирования на языке poST, определение внешнего вида и наполнения интерфейса, имплементация и тестирование интерфейса на веб-сервере и тестирование его работы на модельной задаче.

Консультанты по разделам ВКР (при необходимости, с указанием разделов)

Руководитель ВКР
Зав. КафКТ ФИТ НГУ,
д.т.н., доцент
Зюбин В. Е. /……..
(ФИО) / (подпись)
31 октября 2023г.

Задание принял к исполнению
Харченко А. Д. /……..
(ФИО студента) / (подпись)
31 октября 2023г.

Соруководитель ВКР
к.ф.-м.н.,
доцент КафСИ ФИТ НГУ
Ануреев И. С. /……..
(ФИО) / (подпись)
31 октября 2023г.

СОДЕРЖАНИЕ

ОПРЕДЕЛЕНИЯ, ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ

ПЛК – программируемый логический контроллер.

ОУ – объект управления.

АУ – алгоритм управления.

poST – process-oriented Structured Text.

ЛС – лабораторный стенд.

POU – Program Organization Unit, программный компонент.

ФБ – функциональный блок.

ИАиЭ – Институт Автоматики и Электрометрии.

СО РАН – Сибирское Отделение Российской Академии Наук.

IDE – Integrated Development Environment, интегрированная среда разработки.

JS – JavaScript

CSS — Cascading Style Sheets;

HTML – Hyper Text Markup Language

JSON — JavaScript Object Notation;

ВВЕДЕНИЕ

В ИАиЭ СО РАН ведут активную разработку языковых и инструментальных средств процесс-ориентированного программирования. Один из результатов – язык poST, предназначенный для спецификации управляющих программ в рамках концепции IEC 61131-3.

Язык poST изучается в курсе НГУ “Процесс-ориентированное программирование”. Во время образовательного процесса студент должен не только получить теоретические знания, но и научиться применять их на практике. Этого можно достичь, проводя семинарские занятия и выполняя лабораторные работы.

При изучении языков программирования для создания управляющих алгоритмов качественная подготовка даже важнее, чем при работе с языками высокого уровня, где есть возможность работы с абстрактными объектами и структурами данных, потому что, в зависимости от объекта управления, ошибка в алгоритме управления может повлечь большие финансовые потери, представлять опасность жизни и здоровью людей и даже вызвать экологические проблемы. Для достижения независимости от специфического оборудования, ограничения пространства или бюджета в ИАиЭ СО РАН была сформулирована концепция Виртуальных лабораторных стендов (ВЛС), реализующих в программном виде некие объекты управления (ОУ), для которых можно создать алгоритм управления (АУ) и исполнить, отслеживая изменения.

Ранее наиболее привлекательной выглядела организация лабораторных работ на базе одного из существующих IDE IEC 61131-3[1] CODESYS[3], но в связи с последними событиями данная среда разработки стала недоступна в России.

Исходя из вышесказанного группе практикантов была поставлена задача: создать web-приложение для программирования ВЛС на языке poST.

В прошлом году Масеевским Антоном Михайловичем был разработан транслятор языка post в Python и реализована серверная часть симулятора ПЛК на основе интерпретатора Python.

Цель дипломной работы – разработать интерфейс программиста алгоритма управления в облачном практикуме по языку poST.

Для достижения этой цели были поставлены следующие задачи:

1. Анализ особенностей курса «Процесс-ориентированное программирование» и интерфейса пользователя в IDE CODESYS
2. Формирование списка требований к разрабатываемому интерфейсу
3. Проектирование интерфейса программиста
4. Прототипирование интерфейса программиста
5. Имплементация интерфейса пользователя облачной среды программирования ВЛС и интеграция в веб-приложение
6. Апробация на тестовой ВЛС

Новизна данной работы состоит в создании веб-интерфейса программиста АУ для облачного практикума по языку poST, предоставляющего возможности для создания, запуска и отладки АУ, взаимодействия с симулятором ПЛК.

Практическая ценность работы по итогам данной работы получен веб-интерфейс программиста АУ для облачного практикума по языку poST, который можно использовать для написания, запуска, проверки корректности работы и тестирования алгоритмов управления для программируемых логических контроллеров, а также для подключения к виртуальным ОУ. Созданный инструмент позволяет проводить практические занятия по курсу «Процесс-ориентированное программирование» на языке poST не смотря на невозможность их проведения старыми способами.

Структура и объем работы. Работа состоит из введения, 3 глав и заключения. В первой главе рассматриваются особенности курса «Процесс – ориентированное программирование», существующий подход к организации практических занятий, изучается интерфейс IDE CODESYS, проводится их анализ и формируется список требований к разрабатываемому интерфейсу пользователя. Во второй главе описан процесс разработки интерфейса программиста АУ с учётом сформулированных в первой главе требований.

Третья глава посвящена имплементации веб-интерфейса в веб-приложение и тестированию технологии на модельной задаче.

ГЛАВА 1. Анализ специфики разработки программ для ПЛК в рамках курса “Процесс-ориентированное программирование”

1.1 Особенности курса “Процесс-ориентированное программирование”

Курс «Процесс ориентированное программирование» создан для ознакомления с методами разработки алгоритмов управления сложными техническими объектами на технологическом уровне, знакомство как с типовыми задачами программирования алгоритмов управления и основными моделями и методами их решения, так и с современными, постоянно развивающимися методами процесс-ориентированного программирования; изучаются методы верификации на компьютерных моделях, необходимые для организации итерационной разработки управляющих алгоритмов; большое внимание уделяется работе с временными интервалами и логическим параллелизмом реальных алгоритмов управления, обработке событий и вопросам структуризации.

В ходе курса изучается специфика задач промышленной автоматизации. В рамках изучения промышленной автоматизации студенты знакомятся с таким понятием как программируемый логический контроллер. Узнают его особенности и возможности. По сути своей ПЛК это конечный автомат с набором входов и выходов.

ПЛК не может находиться в «вакууме», что приводит к необходимости понимания такого понятия как объект управления. Который по сути своей является физическим объектом, выполняющим роль динамического и статического окружения ПЛК.

Одной из главных особенностей алгоритма управления ПЛК является его цикличность. Получая входные данные с датчиков ПЛК обрабатывает их и выдаёт на выход сигналы для объекта управления, согласно которым тот должен действовать. При окончании этот порядок действий запускается по новой, данный цикл называется циклом сканирования [5].

Так как объект управления является реальным физическим окружением ПЛК, в нём параллельно происходит множество процессов. Эта особенность приводит к изучению на курсе событийности и логического параллелизма[6] алгоритма управления.

Исходя из того, что ОУ физическая система, студентов обучают тому, как синхронизовать работу АУ с процессами ОУ, что приводит к работе с временными интервалами.

После знакомства с ПЛК, ОУ и пониманием студентами особенностей их работы студентов знакомят с процесс-ориентированным подходом программирования, разработанным в ИАиЭ СО РАН. Объясняют такое понятие как гиперпроцесс [6]. Гиперпроцесс состоит из процессов имеющих общую память и взаимодействующих друг с другом, благодаря средствам синхронизации.

Сами процессы представляют из себя расширенные конечные автоматы с состояниями остановки и возможностью влиять на состояния других процессов и средства синхронизации исполнения.

Программы реализуются циклом, на каждой итерации которого процессы выполняют одно из своих состояний или простаивают, если находятся в состоянии остановки или остановки по ошибке. В начале исполнения инициализирующий процесс находится в одном из своих состояний. Остальные процессы находятся в состоянии нормальной остановки.

1.2 Подход к организации практических занятий при обучении программированию на курсе

Подразумевается, что, получив знания о ПЛК, ОУ, особенностях процесс-ориентированного подхода студенты, с помощью преподавателя освоившие в дальнейшем язык для программирования алгоритма управления ПЛК смогут в программе, предназначенной для программного моделирования ОУ отрабатывать навыки на заранее созданных виртуальных ОУ, создавая для управления ими алгоритм управления, основанный на процесс-ориентированном подходе. Для возможности выполнения студентами практических работ

изначально предполагался язык Reflex и среда разработки “LabView”. Позднее было решено переориентироваться на язык poST[4], созданный ИАиЭ на основе языка ST – Structural Text, и вобравший в себя особенности процесс-ориентированного подхода. Для работы с ним были использованы транслятор из poST в ST и среда разработки CODESYS, которая позволяла как создавать виртуальный объект управления, так и программировать алгоритм управления, предоставляя возможности для редактуры кода, отладки программы и отслеживания всех необходимых программисту значений при её проведении.

1.3 Интерфейс IDE стандарта IEC 61131-3 CODESYS

Интерфейс среды разработки CODESYS выглядит следующим образом: (рисунок 1)

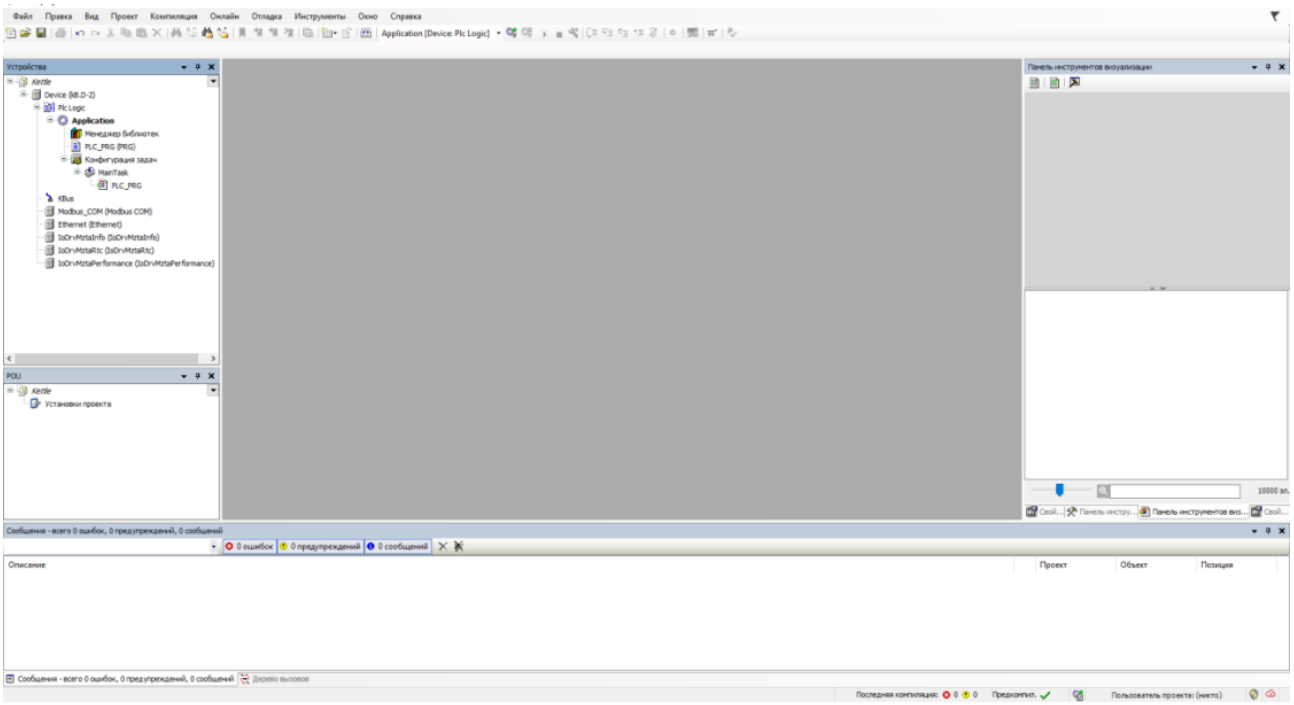

Рисунок 1 – Интерфейс CODESYS

Сверху находятся: панель управления, которая содержит в себе основные разделы для настройки среды CODESYS и работы над проектом; панель инструментов - содержит состоит из наиболее часто используемых команд из панели управления.

Слева расположены: дерево проекта - отображает компоненты проекта (программы, функциональные блоки, экраны визуализации и т. д.) поле POU - содержит глобальные объекты проекта типа POU. В центре окна находится текущие поле редактирования –языка программирования или визуализации.
Справа размещена панель инструментов разработчика, содержащая все элементы, доступные для выбранного языка.
Снизу находятся: панель сообщений - поле предназначено для вывода информации о состоянии компиляции, возникновении ошибок и предупреждениях;

Поле редактирования кода программы в CODESYS состоит из двух панелей – объявление переменных и написание кода. Область объявления переменных выделена словами VAR и END_VAR. Между этими словами происходит объявление локальных переменных.

Предполагается возможность создания либо подключения уже существующей визуализации ОУ.

Во вкладке отладка находятся кнопки, предоставляющие возможность управлять ходом работы АУ, в частности «Старт», «Стоп», «Один цикл» и т.д.

Значения переменных могут отображаться как в тексте исполняемой программы, так и в отдельном окне.

1.4 Требования к реализации интерфейса пользователя, выявленные в результате анализа

В результате анализа выявлено, что web-интерфейс программиста виртуального ПЛК на языке poST должен соответствовать особенностям процесс-ориентированного программирования и конкретно языка poST. Основой же послужит упрощенная и переработанная версия интерфейса CODESYS.

Итоговый список требований, возникших в результате анализа, получился следующим:

1. Должно присутствовать текстовое поле для редактирования кода на языке poST.

2. Должны присутствовать текстовые поля для вывода отладочной информации: значений переменных, состояний процессов и таймеров.

3. Должно быть реализовано поле для изменения во время работы программы значения переменных группы «Input», которые являются отражением датчиков ОУ.

4. Должно быть реализовано текстовое поле для обратной связи программы с пользователем, в нём будут выводиться ошибки, предупреждения, исполненные программой команды пользователя.

5. Должны быть реализованы командная строка и дублирующая её строка инструментов:

a. для работы с кодом как с файлом:
    i. Должна быть возможность сохранить код алгоритма на компьютере пользователя;
    ii. Должна быть возможность загрузить в поле редактирования код АУ с компьютера пользователя;
    iii. Должна быть возможность очистить все поля ввода.
b. для управления работой программы на языке poST:
    i. Должна быть возможность запустить работу алгоритма управления;
    ii. Должна быть возможность остановить работу алгоритма управления;
    iii. Должна быть возможность поставить на паузу и возобновить работу алгоритма управления;
    iv. Должна быть возможность активировать по одному исполняемому циклу алгоритма управления.
c. для перехода к интерфейсу выбора и отладки объекта управления.
    i. Должна быть возможность подключиться к определённому ОУ (учитывая, что программа

предполагается для обучения языку poST, объект управления создаётся заранее преподавателем)

6. Должна быть реализована возможность влиять на скорость работы АУ.

ГЛАВА 2. Разработка интерфейса программиста АУ

2.1 Проектирование интерфейса пользователя

Исходя из требований к интерфейсу программиста алгоритма управления, учитывая особенности языка poST и ориентированность интерфейса лишь на один язык был спроектирован интерфейс пользователя, а собственно, созданы пользовательские сценарии работы с интерфейсом программы.

1. Сбросить все изменения в полях ввода: Нажатие кнопки «Очистить» –> Очистка всех текстовых полей и приведение к изначальному виду
2. Загрузить с компьютера пользователя файл алгоритмом управления: Нажатие кнопки «Загрузить» –> Выбрать файл в открывшемся «Проводнике» –> Код алгоритма появится в текстовом поле для кода алгоритма.
3. Сохранить файл на компьютер: Нажатие кнопки «Сохранить» –> Сохранение в папку «Загрузки»
4. Работа алгоритма: (рисунок 2)
5. Переход к объекту управления: Нажатие кнопки «Открыть вкладку выбора объекта» –> Открытие в браузере новой вкладки с интерфейсом для работы с объектом управления.
6. Ввод кода алгоритма в текстовое поле ввода: Вписать либо вставить скопированный код в поле «Поле ввода программы»
7. Изменить скорость выполнения АУ: Вписать в поле «множитель времени» положительное число D –> нажать на кнопку «Множитель времени» –> выполнение АУ ускорится в D раз

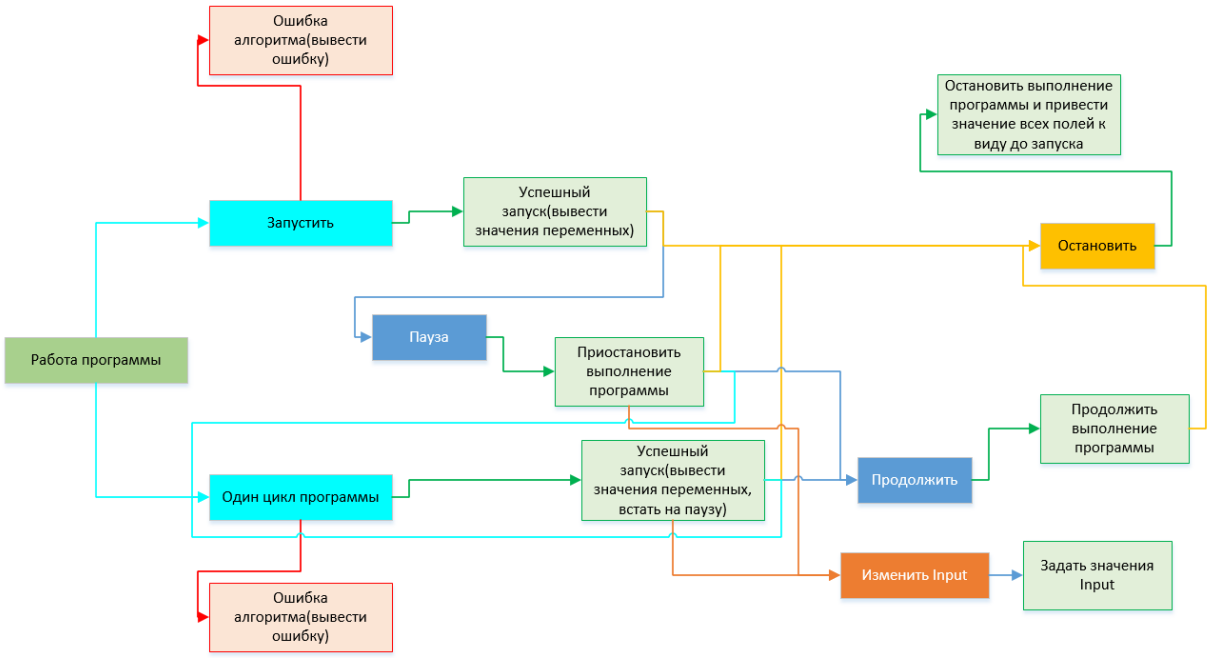

Рисунок 2 – Пользовательский сценарий работа АУ

Ориентированность программы на один язык позволила избавиться от множества пользовательских сценариев, реализованных в CODESYS, по причине их ненужности из-за того, что они были либо связаны с изменением языка программирования, либо с другими языками программирования, либо вообще с функциями, не предполагаемыми к исполнению в программе.

2.2 Прототипирование интерфейса программиста

Исходя из пользовательских сценариев и требований к интерфейсу программисту АУ был создан прототип, а собственно внешний вид интерфейса с активными относительно друг друга элементами интерфейса (Рисунок 3):

1. Панель управления + панель инструментов, с кнопками-иконками, дублирующими кнопки панели управлений:
    a. Раздел Файл, с кнопками:
        i. Очистить – очищает поле ввода
        ii. Загрузить – загружает с компьютера пользователя существующий файл с алгоритмом управления на языке poST
        iii. Сохранить – сохраняет на компьютере пользователя алгоритм управления.

b. Раздел Работа программы с кнопками:
    i. Запустить – запускает работу алгоритма
    ii. Пауза/Продолжить – ставит на паузу и продолжает работу алгоритма
    iii. Остановить – останавливает работу алгоритма полностью
    iv. Один цикл программы – при переходе к режиму отладки исполняет при нажатии один полный цикл алгоритма
    v. Задать Input – отправляет на сервер значения переменных Input группы
c. Раздел Объект управления с кнопкой
    i. Открыть вкладку выбора – позволяет перейти открыть отдельную вкладку для подключения нужного объекта управления нужным способом
d. На панели инструментов поле ввода с кнопкой Множитель времени – позволит ускорить либо замедлить работу poST программы на сервере кратно числу в поле ввода

2. Поле ввода программы – текстовое поле, куда вбивается либо загружается код программы на языке poST. Рядом с глобальными переменными и процессами во время работы программы отображаются их актуальные значения. (Рисунок 4)
3. Поле изменения значений переменных группы «Input» – Во время работы программы поциклово, либо при постановке её на паузу есть возможность изменить входящие параметры алгоритма управления
4. Поле «Ошибки и уведомления» – выводит ошибки и произведённые программистом действия
5. Поля отображения важных значений – отображают состояния процессов, значения параметров в алгоритме управления, таймеры процессов.

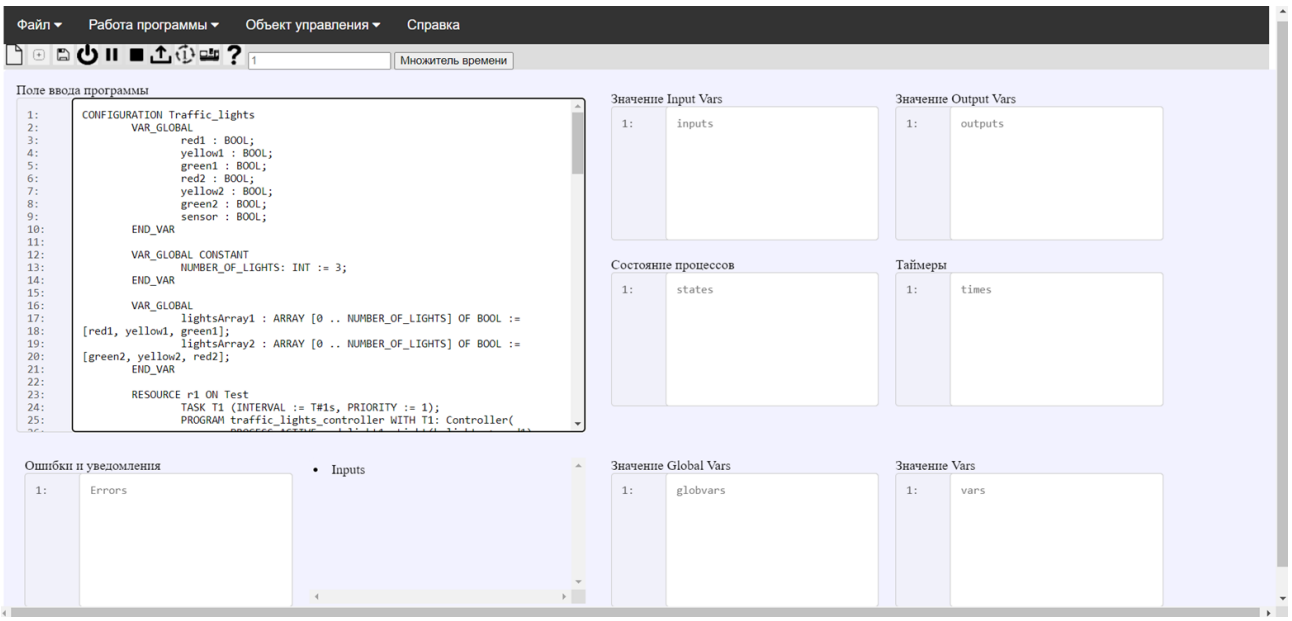

Рисунок 3 – Внешний вид интерфейса пользователя

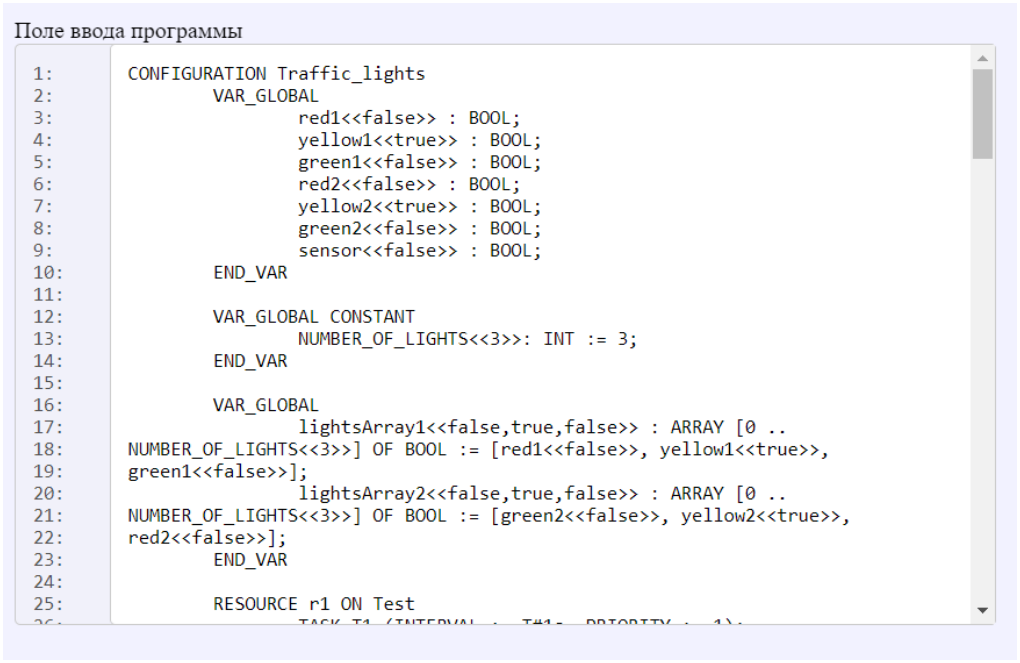

Рисунок 4 – Поле редактирования кода во время работы программы

ГЛАВА 3. Имплементация интерфейса в веб-приложение и тестирование его работы на модельной задаче

3.1 Реализация веб-интерфейса

3.1.1 Выбор технологий

Используются язык разметки HTML, стили CSS и язык JavaScript [7] для реализации логики на стороне клиента.

Для обмена информацией с сервером используется библиотека jquery, это позволяет передавать и принимать информацию в формате JSON, не перезагружая при этом страницу пользователя. Что уменьшает, как количество пересылаемых между сервером и клиентом данных, так и количество вычислений необходимых серверу произвести для каждого пользователя.

Docker выбран для контейнеризации интерфейса, соединённого с виртуальным ПЛК Масеевского А.М. и их зависимостей, что значительно упрощает процесс развертывания и управления инфраструктурой.

3.1.2 Пользовательский интерфейс

Сверху интерфейса реализована панель управления, с выдвигающимися вниз по наведению на них меню под названиями «Файл», «Работа программы», «Объект управления» и «Справка». Под панелью управления находится панель инструментов с кнопками-иконками, дублирующими панель управления. Первые три кнопки работают на клиенте без обращений к серверу. Это: «Очистить» имеет type = reset, потому очищает всю форму до изначального состояния; «Загрузить» вызывает всплывающее окно, позволяющее выбрать файл с локального компьютера пользователя; «Сохранить в файл» сохраняет написанное в поле «Поле ввода программы» в папку «Загрузки» пользователя в формате « .post». Следующие 5 кнопок посылают POST запросы на сервер: кнопка «Запустить» отправляет содержимое поля «Поле ввода программы», своё значение, а также значение в поле «Множитель времени» на сервер, при ответе от сервера об успехе блокирует себя и кнопку для поциклового выполнения алгоритма для исключения лишних нажатий пользователем. Копирует содержимое поля «Поле ввода программы», чтобы при окончании работы

программы вернуть исходный код в поле. Запускает таймер, каждую секунду посылающий GET запросы о всех видах переменных, которые отображаются в полях справа. Рядом с глобальными переменными и процессами во время работы программы в поле «Поле ввода программы» отображаются их актуальные значения в формате: Имя_переменной/Имя_процесса <<значение переменной/состояние процесса>>.

Для локальных переменных, находящихся процессов обозначение текущих значений в формате: pressed<<control2 , false>><<control1 , false>>

Имя_переменной<<имя_процесса,значениепеременной>><<имя_процесса,значениепеременной >> (Приложение В)

Кнопка «Пауза» отправляет своё значение, при успешном ответе от сервера меняет свой внешний вид становясь кнопкой «Продолжить», разблокирует кнопки «Один цикл программы» и «Задать Input», а также приостанавливает таймер посылки GET запросов. Второе нажатие делает всё наоборот.

Кнопка «Задать Input» отправляет на сервер значения переменных из поля ввода «Inputs» для изменения значений переменных, поступающих на вход АУ.

Кнопка «Один цикл программы» отправляет (в случае если программа до этого не была запущена) значение из поля «Поле ввода программы», и своё значение для того чтобы на сервере прошёл лишь один цикл, после чего при успешном ответе посылает GET запросы о значениях всех переменных и обновляет их в полях ввода/вывода, блокирует кнопку «Запустить».

Кнопка «Остановить» посылает POST запрос о прекращении работы АУ. И при успешном ответе от сервера очищает поля вывода значений переменных, приводит кнопки к изначальному виду и возвращает исходный код в поле «Поле ввода программы»

Кнопка «Открыть вкладку выбора объекта» открывает новую вкладку интерфейса выбора/создания объекта управления и передаёт ей значение сессии данного пользователя на сервере.

Кнопка справка открывает текстовое окно с инструкцией пользователя данного приложения.

Кнопка множитель времени отсылает значение из поля «Множитель времени» на сервер для корректировки скорости выполнения АУ.

Слева под панелью инструментов находится поле «Поле ввода программы» именно здесь редактируется программистом АУ.

Под ним слева неизменяемое пользователем поле вывода «Ошибки и уведомления» для вывода сообщений сервера.

Под ним справа форма полей ввода для входных переменных АУ.

В правой половине экрана находятся поля вывода значений переменных, как всех, так и по группам, значений состояний процессов, а также таймеров.

Все поля слева имеют счётчик строк для того, чтобы проще ориентироваться.

Все поля можно прокрутить за ползунок вниз или вверх в случае если вся информация из-за своего количества не влезла на имеющуюся площадь поля.

Код интерфейса (Приложение А).

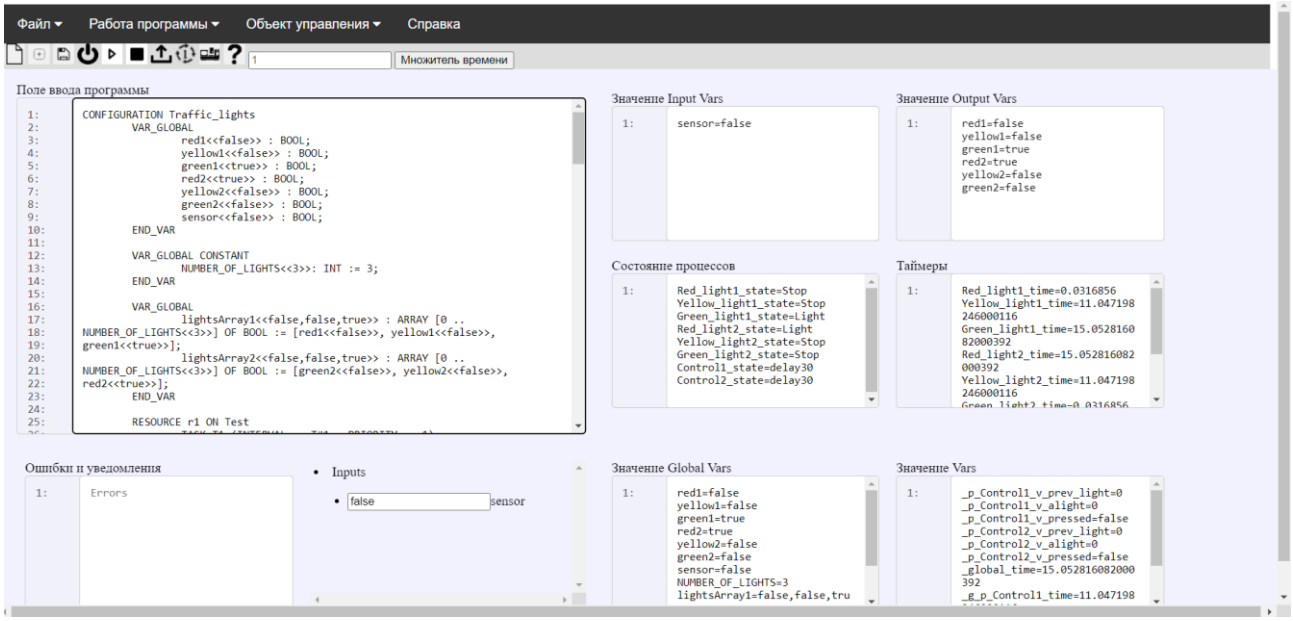

Рисунок 5 – Пользовательский интерфейс веб-приложения

3.2 Тестирование работы пользовательского интерфейса на модельной задаче

Для практического тестирования пользовательского интерфейса в составе веб-приложения была выбрана задача вывода значений и влияния на работу

алгоритма двух синхронизированных трёхцветных светофоров, одновременно выдающих противоположные сигналы, с одним сенсором, который при активации автоматически включает зелёный цвет на одном из них и красный – на другом. Система имеет 6 выходов и 1 вход.

Сформулированы следующие требования, цвета противоположны для другого светофора:

1. Текущие значения переменных, привязанные к сигналам должны выводиться в свои поля вывода согласно их объявлению в АУ.
2. Состояния процессов должны в полном объёме выводиться в поле предназначенное для них.
3. Значения таймеров должны отображаться в полном объёме в поле для значений таймеров.
4. Отправка активации сенсора должна привести к предполагаемому изменению в работе АУ.
5. Отправка множителя времени должна изменить скорость работы алгоритма управления (работу таймеров).
6. Нажатие активации работы алгоритма управления должно запустить программу, в случае ошибки в запуске программы в поле «Ошибки и уведомления» должна быть выведена причина неисполнения АУ.
7. При нажатии на кнопку паузы должны быть выведены последние актуальные значения всех переменных.
8. При нажатии на кнопку остановки должны быть очищены все поля вывода, а в поле ввода должен появиться исходный код алгоритма управления.
9. При нажатии на кнопку активации одного цикла программы должны быть выведены последние актуальные значения всех переменных и поставлена программа на паузу.
10. Редактирование кода в процессе работы запущенного АУ не должно влиять на работу программы на сервере.

11. В поле редактуры кода программы должны быть выведены значения переменных и состояния процессов. При чём если переменная является внутренней переменной какого-либо процесса, то должно быть указано значением при каком процессе является отображаемое значение.

Для тестирования работы интерфейса использовался виртуальный ПЛК написанный Масеевским А.М. с интерфейсом, описанным ранее в этой главе, развёрнутом в Docker-контейнере, в который был установлен образ Ubuntu, Python 3.10 и все необходимые модули для работы веб-сервера, виртуальная машина Java и скопированы все основные реализованные средства, необходимые для работы симулятора.

3.2.1 Реализация алгоритма на языке poST

Алгоритм имеет следующую структуру:

8 процессов, реализующие 2 шаблона:

- Light – процесс, отвечающий за зажигание сигнала. Имеет один выходной сигнал.
- Control – процесс, контролирующий 3 процесса типа Light. На вход получает массив сигналов, в которых хранятся выходные сигналы процессов Light. Имеет 2 входные переменные – массив, про который сказано ранее, и сигнал сенсора, однако массив сам по себе не является входной переменной программы, а потому не должен отображаться в интерфейсе пользователя как поле для ввода.

Синхронизация процесса control2 была произведена просто изменением порядка параметров шаблона, если control1 принимает процессы в порядке: red_light1, yellow_light1, green_light1, то второй в порядке: green_light2, yellow_light2, red_light2. Оба процесса принимают на вход одно значение сенсора.

Полный исходный код алгоритма находится в приложении Б.

Преобразованный интерфейсом код алгоритма находится в приложении В.

3.2.2 Анализ полученных результатов

Разработанный алгоритм на языке poST был передан при помощи пользовательского интерфейса в веб-приложение симулятора ПЛК и запущен там же. Корректность вывода значений переменных и их распределения по полям вывода проверялась путём сравнения значений переменных, выдаваемых в полях интерфейса и существовавших на сервере. Вывод значений переменных в зоне редактирования кода программы соответствует заявленному формату. Работа кнопок интерфейса связанных с работой алгоритма управления соответствует требованиям. Сформулированные требования выполняются.

Кроме того, с помощью этого алгоритма была проверена исправность работы веб-приложения и всех заявленных функций, все встреченные в процессе тестирования недочёты были исправлены.

ЗАКЛЮЧЕНИЕ

В результате работы были проанализированы особенности курса «Процесс-ориентированное программирование», подход к организации практических занятий при обучении программированию на курсе. Также был рассмотрен интерфейс IDE CODESYS. В результате анализа были сформулированных требования к веб-интерфейсу программиста АУ, с соблюдением которых по итогам данной работы реализован веб-интерфейс программиста АУ для облачного практикума по языку poST, который можно использовать для написания, запуска, проверки корректности работы и тестирования алгоритмов управления для программируемых логических контроллеров, а также для подключения к виртуальным ОУ. Созданный инструмент позволяет проводить практические занятия по курсу «Процесс-ориентированное программирование» на языке poST не смотря на невозможность их проведения старыми способами.

В дальнейшем планируется расширить возможности пользователя и добавить возможность ставить точку останова в любом месте АУ, а также добавить синтаксическую подсветку текста АУ.

Выпускная квалификационная работа выполнена мной самостоятельно и с соблюдением правил профессиональной этики. Все использованные в работе материалы и заимствованные принципиальные положения (концепции) из опубликованной научной литературы и других источников имеют ссылки на них. Я несу ответственность за приведенные данные и сделанные выводы.

Я ознакомлен с программой государственной итоговой аттестации, согласно которой обнаружение плагиата, фальсификации данных и ложного цитирования является основанием для не допуска к защите выпускной квалификационной работы и выставления оценки «неудовлетворительно».

_____ _____

ФИО студента *Подпись студента*

« ____ »_____20 __г.

СПИСОК ЛИТЕРАТУРЫ

1. IEC 61131-3 Programmable Controllers–Part 3: Programming Languages // International Electrotechnical Commission. 2013. – 464 с.
2. Зюбин В. Е. Использование виртуальных объектов для обучения программированию информационно-управляющих систем // Информационные технологии, 2009, № 6, С. 79-82.
3. Родченко А. С. Исследование подходов к разработке виртуальных лабораторных стендов в среде CODESYS // МНСК-2019. – 2019. С. 74-74.
4. Bashev V., Anureev I., Zyubin V. The poST language: process-oriented extension for IEC 61131-3 structured text // 2020 International Russian Automation Conference (RusAutoCon). – IEEE, 2020. – С. 994-999.
5. Mader A. A classification of PLC models and applications // Discrete Event Systems: Analysis and Control. – 2000. – С. 239-246.
6. Zyubin V. E. Hyper-automaton: A model of control algorithms // 2007 Siberian Conference on Control and Communications. – IEEE, 2007. – С. 51-57.
7. Robbins J. N. Learning web design: A beginner's guide to HTML, CSS, JavaScript, and web graphics. – Sebastopol, California: O'Reilly Media, 2018. P. 700. ISBN: 9781491960202.
8. Wagner F. Modeling software with finite state machines: a practical approach. – Boca Raton, Florida: Auerbach Publications, 2006. P. 390. ISBN: 9780849380860.
9. Курс «Программирование ПЛК на основе современного отечественного оборудования FASTWEL I/O» // СПбГЭТУ “ЛЭТИ” [сайт]. URL: https://etu.ru/ru/povyshenie-kvalifikacii/programmy/avtomatizaciya-i-upravlenie/programmirovanie-plk-na-osnove-sovremennogo-otechestvennogo-oborudovaniya-fastwel-io (дата обращения: 16.04.2023).
10. PLC TECHNICIAN TRAINING Online Education Program // PLC Technician [сайт]. URL: https://www.plctechnician.com/ (дата обращения: 16.04.2023)

11. Rust & Docker in production @ Coursera // Medium [сайт]. URL: https://medium.com/coursera-engineering/rust-docker-in-production-coursera-f7841d88e6ed (дата обращения: 16.04.2023).
12. Рабочая программа дисциплины «Процесс-ориентированное программирование» // ФИТ НГУ [старый сайт]. URL: https://fit-old.nsu.ru/data_/docs/mag/OOP/4_RPD/TRPS/_TRPS_DV4.4_rpd.pdf (дата обращения: 21.04.2024)

ПРИЛОЖЕНИЕ А

Исходный код интерфейса

```
<!DOCTYPE html>
<html>
<head>
<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">
 <script src="https://ajax.googleapis.com/ajax/libs/jquery/1.7.1/jquery.min.js"></script>

 <script src="../static/jquery-linenumbers.js"></script>
  <script>$('document').ready(function(){
          $('textarea').linenumbers();
          $(window).resize(function() {
             $('textarea').linenumbers();
          });
          document.getElementById('poST_code').addEventListener('keydown', function(e) {
             if (e.key == 'Tab') {
                e.preventDefault();
                var start = this.selectionStart;
                var end = this.selectionEnd;
                this.value = this.value.substring(0, start) + "\t" + this.value.substring(end);
                this.selectionStart = this.selectionEnd = start + 1;
             }
          });
       })</script>
                <script async src="https://www.googletagmanager.com/gtag/js?id=G-3B067ZK7SP"></script>
       <script>
          window.dataLayer = window.dataLayer || [];
          function gtag(){dataLayer.push(arguments);}
          gtag('js', new Date());

          gtag('config', 'G-3B067ZK7SP');
```

```
        </script>
<style>
.navbar {
    overflow: hidden;
    background-color: #333;
    font-family: Arial, Helvetica, sans-serif;
}

.instrbar {
overflow: hidden;
    background-color: #ddd;
            height = 25;
}

.image-button {
  width: 25px;
  height: 25px;
  border: none;
  background-size: cover;
}

.navbar a {
    float: left;
    font-size: 16px;
    color: white;
    text-align: center;
    padding: 14px 16px;
    text-decoration: none;
}

.dropdown {
    float: left;
    overflow: hidden;
}
```

```
.dropdown .dropbtn {
    font-size: 16px;
    border: none;
    outline: none;
    color: white;
    padding: 14px 16px;
    background-color: inherit;
    font-family: inherit;
    margin: 0;
}

.navbar a:hover, .dropdown:hover .dropbtn {
    background-color: red;
}

.dropdown-content {
    display: none;
    position: absolute;
    background-color: #f9f9f9;
    min-width: 160px;
    box-shadow: 0px 8px 16px 0px rgba(0,0,0,0.2);
    z-index: 1;
}

.dropdown-content button, .dropdown-content a {
    float: none;
    color: black;
    padding: 12px 16px;
        width:100%;
    text-decoration: none;
    display: block;
    text-align: left;
        cursor: pointer;
}
```

```
.dropdown-content button:hover, .dropdown-content a:hover {
   background-color: #ddd;
}

.btn {
   float: none;
   background-color: white;
   border: none;
   color: black;
   padding: 12px 16px;
        width: 100%;
   font-size: 16px;
   cursor: pointer;
}

/* Darker background on mouse-over */
.btn:hover {
   background-color: #ddd;
}

.dropdown:hover .dropdown-content {
   display: block;
}



.input-file {
        position: relative;
        display: inline-block;
}
.input-file span {
        position: relative;
        display: inline-block;
        cursor: pointer;
        outline: none;
        text-decoration: none;
```

```
        font-size: 14px;
        vertical-align: middle;
        color: rgb(255 255 255);
        text-align: center;
        border-radius: 4px;
        background-color: #419152;
        line-height: 22px;
        height: 40px;
        padding: 10px 20px;
        box-sizing: border-box;
        border: none;
        margin: 0;
        transition: background-color 0.2s;
}
.input-file input[type=file] {
        position: absolute;
        z-index: -1;
        opacity: 0;
        display: block;
        width: 0;
        height: 0;
}

/* Focus */
.input-file input[type=file]:focus + span {
        box-shadow: 0 0 0 0.2rem rgba(0,123,255,.25);
}

/* Hover/active */
.input-file:hover span {
        background-color: #59be6e;
}
.input-file:active span {
        background-color: #2E703A;
}
```

```
/* Disabled */
.input-file input[type=file]:disabled + span {
        background-color: #eee;
}

.column {
    float: left;
    width: 45%;
    padding: 10px;
}

/* Clear floats after the columns */
.row:after {
height:50%;
width: 100%;
    content: "";
    display: table;
    clear: both;
}



textarea {

    width: 100%;
        height:100%;
    padding: 12px;
    border: 1px solid #ccc;
    border-radius: 4px;
    box-sizing: border-box;
    margin-top: 0px;

    resize: none;
}
```

```
.container {
    width: 100%;
    border-radius: 5px;
    background-color: #f2f2ff;
    padding: 5px;
}
</style>
</head>
<body>
<form method="POST" enctype="multipart/form-data" id="poST_code_form"
name="poST_code_form">
<div class="navbar">
  <div class="dropdown">
    <button type="button" name ="but" class="dropbtn">Файл
      <i class="fa fa-caret-down"></i>
    </button>
    <div class="dropdown-content">
      <button id = "clear" name ="but" class="btn" type = "reset" form = "poST_code_form">
Очистить</button>
            <label class="input-file">
                  <input type="file" id="Download">
                  <span>Загрузить</span>
           </label>
            <a id="tosave">Сохранить в файл</a>

    </div>
  </div>
  <div class="dropdown">
          <button type="button" name ="but" class="dropbtn">Работа программы
      <i class="fa fa-caret-down"></i>
    </button>
    <div class="dropdown-content">
```

```
        <button type="button" class="btn" id ="Run" name ="action" value="runProgram" >
Запустить</button>
        <button type="button" class="btn" style= "background-color: #ddd" name ="action"
id="pauseSim" value="pauseProgram" disabled=""><span id="pause" >Пауза</span></button>
        <button type="button" class="btn" style= "background-color: #ddd" name ="action"
id="stopSim" value="stopProgram" disabled="">Остановить</button>
        <button type="button" class="btn" style= "background-color: #ddd" name ="action"
id="loadInp" value="loadInputs_no_new_page" disabled="">Задать Input</button>
        <button type="button" class="btn" style= "background-color: #ddd" name ="action"
id="oneCycle" value="oneCycle" >Один цикл программы</button>
    </div>
  </div>
  <div class="dropdown">
        <button type="button" name ="but" class="dropbtn" name ="Object">Объект управления
     <i class="fa fa-caret-down"></i>
    </button>
    <div class="dropdown-content">
        <button type="button" id = "Object" name ="but" class="btn"> Открыть вкладку
выбора</button>

    </div>
  </div>
  <a id = "About" href="#news">Справка</a>
</div>

<div class="instrbar">

     <button type="button" title="Очистить" name ="but" class="image-button"
style="background-image: url('../static/clean.png')" onclick =
document.getElementById('clear').click() form = "poST_code_form" ></button>
          <button type="button" title="Загрузить код из компьютера" name ="but" class="image-
button" style="background-image: url('../static/open.png')" id="download" onclick =
document.getElementById('Download').click()></button>
```

<button type="button" title="Сохранить на компьютере" name ="but" class="image-button" style="background-image: url('../static/save.png')" id ="save" onclick = document.getElementById('tosave').click()></button>

<button type="button" title="Запуск программы" name ="but" class="image-button" style="background-image: url('../static/run.png')" id ="run" onclick = document.getElementById('Run').click()></button>

<button type="button" title="Пауза/Продолжить" name ="but" class="image-button" style= " background-image: url('../static/pause.png')" id="pausesim" disabled="" onclick = document.getElementById('pauseSim').click()></button>

<button type="button" title="Остановить" name ="but" class="image-button" style= " background-image: url('../static/stop.png')" id="stopsim" disabled="" onclick = document.getElementById('stopSim').click()></button>

<button type="button" title="Задать Input" name ="but" class="image-button" style= " background-image: url('../static/loadinp.png')" id="loadinp" disabled="" onclick = document.getElementById('loadInp').click()></button>

<button type="button" title="Один цикл программы" name ="but" class="image-button" style= " background-image: url('../static/onecycle.png')" id="onecycle" onclick = document.getElementById('oneCycle').click()></button>

<button type="button" title="Открыть вкладку выбора объекта" name ="but" class="image-button"style="background-image: url('../static/object.png')" onclick = document.getElementById('Object').click()> </button>

<button type="button" title="Справка" name ="but" class="image-button" style="background-image: url('../static/about.png')" id ="about" onclick = document.getElementById('About').click()></button>

<input class="ss" name ="mnoj" id = "mnojitel" style="width: 100px height:25px" type="text" placeholder="1"></input>

<button type="button" name ="action" id = "mnojVrem" style="width: 50 height: 25px " value="mnojVrem" >Множитель времени</button>

```
    <div class="row">
  <div class="column">
   <label for="poST_code" style = "font-family: Arial">Поле ввода программы</label>
      <div class="col-6"style="height:400px">

            <textarea type="text" id="poST_code" name="poST_code" placeholder="poST_code"
style="height:400px" autofocus></textarea>

          </div>

          </div>
          <div class="column">
      <div class = "row">
                  <div class="column" style = "width: 47%">
                  <label for="Input" style = "font-family: Arial">Значение Input Vars</label>
                 <div class="col-6" style="height:160px">

              <textarea id="Input" name="Input" style="height:160px" placeholder="inputs"
readonly></textarea>
                  </div>
           </div>


                  <div class="column" style = "width: 47%">
                  <label for="Output" style = "font-family: Arial">Значение Output Vars</label>
        <div class="col-6" style="height:160px">

              <textarea id="Output" name="Output" style="height:160px" placeholder="outputs"
readonly></textarea>
                  </div>
           </div>

          </div>
          <div class = "row">
```

```
                <div class="column" style = "width: 47%">
                <label for="States" style = "font-family: Arial">Состояние процессов</label>
        <div class="col-6" style="height:160px">

            <textarea id="States" name="States" style="height:160px" placeholder="states"
readonly></textarea>
                </div>
          </div>

                <div class="column" style = "width: 47%">
                <label for="Times" style = "font-family: Arial">Таймеры</label>
       <div class="col-6" style="height:160px">

            <textarea id="Times" name="Times" style="height:160px" placeholder="times"
readonly></textarea>
                </div>
          </div>

         </div>
        </div>
        </div>
        <div class="row">
        <div class="column">
          <div class="column" style = "width: 47%">


                <label for="Errors" style = "font-family: Arial">Ошибки и уведомления</label>
       <div class="col-6" style="height:160px">

            <textarea id="Errors" name="Errors" style="height:160px" placeholder="Errors"
readonly></textarea>
                </div>
                </div>
                <div class="column" style = "width: 47%">
```

```
                <div class="container" style="height: 160px; overflow: scroll; padding-bottom:
10px;">
                    <li>
                      <a style = "font-family: Arial">Inputs</a>
                      <ul class="scroll" id="assetNameMenu">
                      </ul>
                    </li>
                  </div>
                </div>
                    </div>

<div class="column">

            <div class="column" style = "width: 47%">
                    <label for="Globvars" style = "font-family: Arial">Значение Global Vars</label>
                      <div class="col-6" style="height:160px">

                  <textarea id="Globvars" name="Globvars" style="height:160px"
placeholder="globvars" readonly></textarea>
                      </div>
                </div>
                    <form id="out">
                    <div class="column" style = "width: 47%">
                    <label for="Vars" style = "font-family: Arial">Значение Vars</label>
          <div class="col-6" style="height:160px">

                  <textarea id="Vars" name="Vars" style="height:160px" placeholder="vars"
readonly></textarea>
                      </div>
                </div>
                    </form>
                    </div>
                </div>
```

```
</div>
</form>
<script>
$(".ss").keypress(function(event){
  event = event || window.event;
  if (event.charCode && event.charCode!=0 && event.charCode!=46 && (event.charCode < 48 ||
event.charCode > 57) )
    return false;
});
</script>
<script>

$("#Download").change(function(){
        var reader = new FileReader();
        reader.onload = function(e){
                $("#poST_code").val(e.target.result);
        };
        reader.readAsText($("#Download")[0].files[0], "UTF-8");
});
</script>
<script>
document.getElementById('tosave').onclick = function() {
    let text = document.getElementById("poST_code").value;
    let myData = 'data:application/txt;charset=utf-8,' + encodeURIComponent(text);
    this.href = myData;
    this.download = 'code.post';
}
</script>

<script>
document.getElementById('Object').onclick = function() {
    window.open(window.location.href + 'plant/?userpath={{path}}')
}
</script>
```

```
<script>
document.getElementById('mnojVrem').onclick = function() {
   var data = new FormData()
  data.append('action', 'mnojVrem')
  data.append('mnoj', document.getElementById('mnojitel').value)
  data.append('poST_code', document.getElementById('poST_code').value)
  console.log(Array.from(data.entries()))
  $.ajax({
            type: 'POST',
                        contentType: false,
                        processData: false,
            url: window.location.href,
            data: data,
            success: function(data){}});
}
</script>


<script>
var intervalId;
document.getElementById('Run').onclick = function() {
programText=document.getElementById('poST_code').value;
var data = new FormData()
  data.append('action', 'runProgram')
  data.append('mnoj', document.getElementById('mnojitel').value)
  data.append('poST_code', document.getElementById('poST_code').value)
  console.log(Array.from(data.entries()))
  $.ajax({
            type: 'POST',
                        contentType: false,
                        processData: false,
            url: window.location.href,
            data: data,
            success: function(data){
```

```
document.getElementById('Run').setAttribute('disabled',true);
document.getElementById('Run').style.background = "#ddd";
document.getElementById('oneCycle').setAttribute('disabled',true);
document.getElementById('oneCycle').style.background = "#ddd";
document.getElementById('onecycle').setAttribute('disabled',true);
document.getElementById('onecycle').style.backgroundColor = "#ddd";
document.getElementById('run').setAttribute('disabled',true);
document.getElementById('run').style.backgroundColor = "#ddd";
  document.getElementById('pauseSim').removeAttribute("disabled");
  document.getElementById('pauseSim').style.background="white";
  document.getElementById('pausesim').removeAttribute("disabled");
  document.getElementById('pausesim').style.backgroundColor= "white";
  document.getElementById('stopSim').removeAttribute("disabled");
  document.getElementById('stopSim').style.background= "white";
  document.getElementById('stopsim').removeAttribute("disabled");
  document.getElementById('stopsim').style.backgroundColor= "white";
  updateInputs();
}});
intervalId = setInterval(() => {updateVars();updateProg();}, 1000);
  }
</script>


 <script>
            function updateInputs() {
               $.getJSON("{{path}}"+"inputs", function(data) {
                  if(data ==null)
                                                    {
                     setTimeout(updateInputs, 1000);
                  }
```

```
                    var assetList = $('#assetNameMenu');
                    for(var id in data)
                    {

                        var li = $('<li/>')
                            .addClass('ui-menu-item')
                            .attr('role', 'menuitem')
                            .appendTo(assetList);

                        var input = $('<input/>')
                            .addClass('ui-all')
                            .attr('type', 'text')
                            .attr('value', data[id])
                            .attr('name', 'inputcheck')
                            .appendTo(li);

                        var aaa = $('<a/>')
                            .addClass('ui-all')
                            .text(id)
                            .appendTo(li);

                    };
                });
            }
        </script>

<script>
document.getElementById('pauseSim').onclick = function() {
var data = new FormData()
  data.append('action', 'pauseProgram')
  data.append('poST_code', programText)
  console.log(Array.from(data.entries()))
  $.ajax({
        type: 'POST',
```

```
                    contentType: false,
                    processData: false,
            url: window.location.href,
            data: data,
            success: function(data){
   if(document.getElementById('pause').innerHTML === 'Пауза') {
          document.getElementById('pause').innerHTML = 'Продолжить';
          clearInterval(intervalId);
          updateVars();
          document.getElementById('oneCycle').removeAttribute("disabled");
    document.getElementById('oneCycle').style.background= "white";
           document.getElementById('onecycle').removeAttribute("disabled");
  document.getElementById('onecycle').style.backgroundColor= "white";
          document.getElementById('pausesim').style.backgroundImage = "url('../static/pause2.png')";
          document.getElementById('loadInp').removeAttribute("disabled");
    document.getElementById('loadInp').style.background= "white";
           document.getElementById('loadinp').removeAttribute("disabled");
  document.getElementById('loadinp').style.backgroundColor= "white";
          updateProg();
    }
          else {
          document.getElementById('oneCycle').setAttribute('disabled',true);
          document.getElementById('oneCycle').style.background = "#ddd";
          document.getElementById('onecycle').setAttribute('disabled',true);
          document.getElementById('onecycle').style.backgroundColor = "#ddd";
          document.getElementById('loadInp').setAttribute('disabled',true);
    document.getElementById('loadInp').style.background = "#ddd";
          document.getElementById('loadinp').setAttribute('disabled',true);
          document.getElementById('loadinp').style.backgroundColor = "#ddd";
          document.getElementById('pause').innerHTML = 'Пауза';
          document.getElementById('pausesim').style.backgroundImage = "url('../static/pause.png')";
          intervalId = setInterval(() => {
          updateVars(); updateProg();}, 1000);
    }
}});
```

```
}
</script>

<script>
document.getElementById('loadInp').onclick = function() {
var data = new FormData()
  data.append('action', 'loadInputs_no_new_page')
  data.append('poST_code', programText)
  $("[name ='inputcheck']").each(function() {
    data.append($(this).attr("name"), $(this).attr("value"))
});
console.log(Array.from(data.entries()))
$.ajax({
            type: 'POST',
                        contentType: false,
                        processData: false,
            url: window.location.href,
            data: data,
            success: function(data){
                        document.getElementById('Errors').value = "Input установлен";
   }
});



}
</script>

<script>
document.getElementById('oneCycle').onclick = function() {
if (document.getElementById('Run').style.background === "white")
{
programText=document.getElementById('poST_code').value;
}
var data = new FormData()
  data.append('action', 'oneCycle')
```

```
  data.append('poST_code', programText)
  console.log(Array.from(data.entries()))
  $.ajax({
            type: 'POST',
                        contentType: false,
                        processData: false,
            url: window.location.href,
            data: data,
            success: function(data){
    document.getElementById('Run').setAttribute('disabled',true);
document.getElementById('Run').style.background = "#ddd";
  document.getElementById('run').setAttribute('disabled',true);
document.getElementById('run').style.backgroundColor = "#ddd";
    if(document.getElementById('pause').innerHTML === 'Пауза')
{document.getElementById('pause').innerHTML = 'Продолжить';
document.getElementById('pausesim').style.backgroundImage = "url('../static/pause2.png')";}
    if(document.getElementById('pauseSim').style.background === "#ddd"){
          document.getElementById('pauseSim').removeAttribute("disabled");
   document.getElementById('pauseSim').style.background="white";
   document.getElementById('pausesim').removeAttribute("disabled");
   document.getElementById('pausesim').style.backgroundColor= "white";
   }
   if(document.getElementById('stopSim').style.background === "#ddd"){
   document.getElementById('stopSim').removeAttribute("disabled");
   document.getElementById('stopSim').style.background= "white";
   document.getElementById('stopsim').removeAttribute("disabled");
   document.getElementById('stopsim').style.backgroundColor= "white";
   }
   if(document.getElementById('loadInp').style.background === "#ddd"){
   document.getElementById('loadInp').removeAttribute("disabled");
   document.getElementById('loadInp').style.background= "white";
   document.getElementById('loadinp').removeAttribute("disabled");
   document.getElementById('loadinp').style.backgroundColor= "white";
   }
   clearInterval(intervalId);
```

```
  updateVars();
  updateProg();

});
}
</script>

<script>
document.getElementById('stopSim').onclick = function() {
var data = new FormData()
 data.append('action', 'stopProgram')
 data.append('poST_code', programText)
 console.log(Array.from(data.entries()))
 $.ajax({
            type: 'POST',
                        contentType: false,
                        processData: false,
            url: window.location.href,
            data: data,
            success: function(data){
   document.getElementById('Run').removeAttribute("disabled");
  document.getElementById('Run').style.background= "white";
 document.getElementById('run').removeAttribute("disabled");
  document.getElementById('run').style.backgroundColor= "white";
   if(document.getElementById('pause').innerHTML === 'Продолжить')
{document.getElementById('pause').innerHTML = 'Пауза';
document.getElementById('pausesim').style.backgroundImage = "url('../static/pause.png')";}
          document.getElementById('pauseSim').setAttribute('disabled',true);
          document.getElementById('pauseSim').style.background = "#ddd";
          document.getElementById('pausesim').setAttribute('disabled',true);
          document.getElementById('pausesim').style.backgroundColor = "#ddd";
          document.getElementById('loadInp').setAttribute('disabled',true);
   document.getElementById('loadInp').style.background = "#ddd";
          document.getElementById('loadinp').setAttribute('disabled',true);
  document.getElementById('loadinp').style.backgroundColor = "#ddd"
```

```
        document.getElementById('oneCycle').removeAttribute("disabled");
    document.getElementById('oneCycle').style.background = "white";
        document.getElementById('onecycle').removeAttribute("disabled");
   document.getElementById('onecycle').style.backgroundColor= "white";
   document.getElementById('stopSim').setAttribute('disabled',true);
   document.getElementById('stopSim').style.background = "#ddd";
   document.getElementById('stopsim').setAttribute('disabled',true);
   document.getElementById('stopsim').style.backgroundColor = "#ddd";
   clearInterval(intervalId);
   document.getElementById('poST_code').value = programText;
   $('.ui-all').each(function() {
    $(this).remove();
});
$('.ui-menu-item').each(function() {
    $(this).remove();
});

document.getElementById('Output').value ="";
document.getElementById('Input').value ="";
document.getElementById('Times').value ="";
document.getElementById('States').value ="";
document.getElementById('Vars').value ="";
document.getElementById('Globvars').value ="";
}});
}
</script>

<script>
var programText;
var modProgramText;
function updateProg() {
modProgramText = programText;
updateProgStates()
updateProgGlobvars()
updateProgVars()
```

```
}

function updateProgStates() {
$.getJSON("{{path}}"+"states", function(data) {


                for(var id in data){
                                                let modId = id.toLowerCase();
                                                modId = modId.substring(0,modId.length-6)
                                                let indexSbstr =
modProgramText.indexOf(modId);
                                                while (indexSbstr != -1)
                                                {
                                                modProgramText =
modProgramText.substring(0, indexSbstr + modId.length) + "<<" + data[id] + ">>" +
modProgramText.substring(indexSbstr + modId.length, modProgramText.length);
                                                 indexSbstr =
modProgramText.indexOf(modId, indexSbstr + modId.length);
                                                 }
                                                }
                                                document.getElementById('poST_code').value
= modProgramText;
      });
}


function updateProgGlobvars() {
$.getJSON("{{path}}"+"globvars", function(data) {


            for(var id in data){
                                                let indexSbstr = modProgramText.indexOf(id)
                                                while (indexSbstr != -1)
```

```
                                                {
                                                modProgramText =
modProgramText.substring(0, indexSbstr + id.length) + "<<" + data[id] + ">>" +
modProgramText.substring(indexSbstr + id.length, modProgramText.length);
                                                 indexSbstr = modProgramText.indexOf(id,
indexSbstr + id.length);
                                                 }
                                                }
                                                document.getElementById('poST_code').value
= modProgramText;
        });
}

function updateProgVars() {
$.getJSON("{{path}}"+"vars", function(data) {


                        for(var id in data){
                                                let modIdLow = id.toLowerCase();
                                                let indexV = modIdLow.indexOf("_v_");
                                                let indexP = modIdLow.indexOf("_p_");
                                                if(indexV != -1){
                                                let modId = modIdLow.substring(indexV+3,
modIdLow.length)
                                                let indexSbstr =
modProgramText.indexOf(modId);
                                                while (indexSbstr != -1)
                                                {
                                                modProgramText =
modProgramText.substring(0, indexSbstr + modId.length) + "<<" +
modIdLow.substring(indexP+3, indexV) +" , "+ data[id] + ">>" +
modProgramText.substring(indexSbstr + modId.length, modProgramText.length);
                                                 indexSbstr =
modProgramText.indexOf(modId, indexSbstr + modId.length);
                                                 }
```

```
                                                    }
                                                    }
                                                    document.getElementById('poST_code').value
= modProgramText;
        });
}
</script>


<script>


    function updateVars() {

        $.getJSON("{{path}}"+"outputs", function(data) {


            document.getElementById('Output').value ="";
                    for(var id in data){
                                                        document.getElementById('Output').value =
document.getElementById('Output').value + id + "=" + data[id] + "\n";
                                                    }
        });
                $.getJSON("{{path}}"+"states", function(data) {


            document.getElementById('States').value ="";
                    for(var id in data){
                                                        document.getElementById('States').value =
document.getElementById('States').value + id + "=" + data[id] + "\n";
                                                    }
        });
                $.getJSON("{{path}}"+"inputs", function(data) {
```

```
        document.getElementById('Input').value ="";
                for(var id in data){
                                                document.getElementById('Input').value =
document.getElementById('Input').value + id + "=" + data[id] + "\n";
                                                }
    });
            $.getJSON("{{path}}"+"times", function(data) {


        document.getElementById('Times').value ="";
                for(var id in data){
                                                document.getElementById('Times').value =
document.getElementById('Times').value + id + "=" + data[id] + "\n";
                                                }
    });
            $.getJSON("{{path}}"+"globvars", function(data) {


        document.getElementById('Globvars').value ="";
                for(var id in data){
                                                document.getElementById('Globvars').value =
document.getElementById('Globvars').value + id + "=" + data[id] + "\n";
                                                }
    });
            $.getJSON("{{path}}"+"vars", function(data) {


        document.getElementById('Vars').value ="";
                for(var id in data){
                                                document.getElementById('Vars').value =
document.getElementById('Vars').value + id + "=" + data[id] + "\n";
                                                }
    });
  }
</script>
```

```
</body>
</html>
```

ПРИЛОЖЕНИЕ Б

Исходный код модельной задачи на языке poST

```
CONFIGURATION Traffic_lights
VAR_GLOBAL
	red1 : BOOL;
	yellow1 : BOOL;
	green1 : BOOL;
	red2 : BOOL;
	yellow2 : BOOL;
	green2 : BOOL;
	sensor : BOOL;
END_VAR

VAR_GLOBAL CONSTANT
	NUMBER_OF_LIGHTS: INT := 3;
END_VAR

VAR_GLOBAL
	lightsArray1 : ARRAY [0 .. NUMBER_OF_LIGHTS] OF BOOL := [red1, yellow1,
green1];
	lightsArray2 : ARRAY [0 .. NUMBER_OF_LIGHTS] OF BOOL := [green2,
yellow2, red2];
END_VAR

RESOURCE r1 ON Test
	TASK T1 (INTERVAL := T#1s, PRIORITY := 1);
	PROGRAM traffic_lights_controller WITH T1: Controller(
		PROCESS ACTIVE red_light1: Light(b_light => red1),
		PROCESS yellow_light1: Light(b_light => yellow1),
		PROCESS green_light1: Light(b_light => green1),
		PROCESS red_light2 : Light(b_light => red2),
		PROCESS yellow_light2 : Light(b_light => yellow2),
		PROCESS ACTIVE green_light2 : Light(b_light => green2),
```

```
                PROCESS ACTIVE control1: Control(control_sensor := sensor, pRed :=
red_light1, pYellow := yellow_light1, pGreen := green_light1, rLightsArray := lightsArray1),
                PROCESS ACTIVE control2: Control(control_sensor := sensor, pRed :=
green_light2, pYellow := yellow_light2, pGreen := red_light2, rLightsArray := lightsArray2)
        );
    END_RESOURCE
    END_CONFIGURATION

    PROGRAM Controller
    PROCESS Light
        VAR_OUTPUT
            b_light : BOOL;
        END_VAR

        STATE Light
            b_light := TRUE;
        END_STATE
    END_PROCESS

    PROCESS Control
        VAR_INPUT
            control_sensor : BOOL;
            rLightsArray : ARRAY [*] OF BOOL;
        END_VAR

        VAR_PROCESS
            pRed : Light;
            pYellow : Light;
            pGreen : Light;
        END_VAR

        VAR
            prev_light : INT;
            alight : INT;
            pressed : BOOL;
```

```
END_VAR

STATE Work
  IF pressed THEN
        prev_light := 0;
        pressed := FALSE;
    ELSIF ((PROCESS pRed IN STATE INACTIVE) AND (PROCESS pGreen IN STATE ACTIVE)) OR ((PROCESS pGreen IN STATE INACTIVE) AND (PROCESS pRed IN STATE ACTIVE)) THEN
        FOR alight := 0 TO NUMBER_OF_LIGHTS DO
            IF rLightsArray[alight] THEN
                prev_light := alight;
            END_IF
            rLightsArray[alight] := FALSE;
        END_FOR
        STOP PROCESS pRed;
        START PROCESS pYellow;
        STOP PROCESS pGreen;
        SET STATE delay10;
    ELSIF prev_light = 0 THEN
        FOR alight := 0 TO NUMBER_OF_LIGHTS DO
            rLightsArray[alight] := FALSE;
        END_FOR
        STOP PROCESS pRed;
        STOP PROCESS pYellow;
        START PROCESS pGreen;
        SET STATE delay30;
    ELSIF prev_light = 2 THEN
        FOR alight := 0 TO NUMBER_OF_LIGHTS DO
            rLightsArray[alight] := FALSE;
        END_FOR
        START PROCESS pRed;
        STOP PROCESS pYellow;
        STOP PROCESS pGreen;
        SET STATE delay30;
```

```
            END_IF
        END_STATE

        STATE delay10
            TIMEOUT T#10s THEN
                SET STATE Work;
                IF control_sensor THEN
                    pressed := TRUE;
                END_IF
            END_TIMEOUT
        END_STATE

        STATE delay30
            IF control_sensor AND PROCESS pRed IN STATE ACTIVE THEN
               pressed := TRUE;
                SET STATE Work;
            END_IF
            TIMEOUT T#30s THEN
                pressed := FALSE;
                SET STATE Work;
            END_TIMEOUT
        END_STATE
END_PROCESS
END_PROGRAM
```

ПРИЛОЖЕНИЕ В

Преобразованный код модельной задачи на языке poST

```
CONFIGURATION Traffic_lights
	VAR_GLOBAL
		red1<<false>> : BOOL;
		yellow1<<false>> : BOOL;
		green1<<true>> : BOOL;
		red2<<true>> : BOOL;
		yellow2<<false>> : BOOL;
		green2<<false>> : BOOL;
		sensor<<false>> : BOOL;
	END_VAR

	VAR_GLOBAL CONSTANT
		NUMBER_OF_LIGHTS<<3>>: INT := 3;
	END_VAR

	VAR_GLOBAL
		lightsArray1<<false,false,true>> : ARRAY [0 .. NUMBER_OF_LIGHTS<<3>>]
OF BOOL := [red1<<false>>, yellow1<<false>>, green1<<true>>];
		lightsArray2<<false,false,true>> : ARRAY [0 .. NUMBER_OF_LIGHTS<<3>>]
OF BOOL := [green2<<false>>, yellow2<<false>>, red2<<true>>];
	END_VAR

	RESOURCE r1 ON Test
		TASK T1 (INTERVAL := T#1s, PRIORITY := 1);
		PROGRAM traffic_lights_controller WITH T1: Controller(
			PROCESS ACTIVE red_light1<<Stop>>: Light(b_light => red1<<false>>),
			PROCESS yellow_light1<<Stop>>: Light(b_light => yellow1<<false>>),
			PROCESS green_light1<<Light>>: Light(b_light => green1<<true>>),
			PROCESS red_light2<<Light>> : Light(b_light => red2<<true>>),
			PROCESS yellow_light2<<Stop>> : Light(b_light => yellow2<<false>>),
			PROCESS ACTIVE green_light2<<Stop>> : Light(b_light =>
green2<<false>>),
```

```
                    PROCESS ACTIVE control1<<delay30>>:
Control(control_sensor<<false>> := sensor<<false>>, pRed := red_light1<<Stop>>, pYellow :=
yellow_light1<<Stop>>, pGreen := green_light1<<Light>>, rLightsArray :=
lightsArray1<<false,false,true>>),
                    PROCESS ACTIVE control2<<delay30>>:
Control(control_sensor<<false>> := sensor<<false>>, pRed := green_light2<<Stop>>, pYellow :=
yellow_light2<<Stop>>, pGreen := red_light2<<Light>>, rLightsArray :=
lightsArray2<<false,false,true>>)
            );
    END_RESOURCE
END_CONFIGURATION

PROGRAM Controller
    PROCESS Light
        VAR_OUTPUT
            b_light : BOOL;
        END_VAR

        STATE Light
            b_light := TRUE;
        END_STATE
    END_PROCESS

    PROCESS Control
        VAR_INPUT
            control_sensor<<false>> : BOOL;
            rLightsArray : ARRAY [*] OF BOOL;
        END_VAR

        VAR_PROCESS
            pRed : Light;
            pYellow : Light;
            pGreen : Light;
        END_VAR
```

```
VAR
    prev_light<<control2 , 0>><<control1 , 0>> : INT;
    alight<<control2 , 0>><<control1 , 0>> : INT;
    pressed<<control2 , false>><<control1 , false>> : BOOL;
END_VAR

STATE Work
  IF pressed<<control2 , false>><<control1 , false>> THEN
      prev_light<<control2 , 0>><<control1 , 0>> := 0;
      pressed<<control2 , false>><<control1 , false>> := FALSE;
    ELSIF ((PROCESS pRed IN STATE INACTIVE) AND (PROCESS pGreen
IN STATE ACTIVE)) OR ((PROCESS pGreen IN STATE INACTIVE) AND (PROCESS pRed IN
STATE ACTIVE)) THEN
      FOR alight<<control2 , 0>><<control1 , 0>> := 0 TO
NUMBER_OF_LIGHTS<<3>> DO
        IF rLightsArray[alight<<control2 , 0>><<control1 , 0>>]
THEN
          prev_light<<control2 , 0>><<control1 , 0>> :=
alight<<control2 , 0>><<control1 , 0>>;
        END_IF
        rLightsArray[alight<<control2 , 0>><<control1 , 0>>] :=
FALSE;
      END_FOR
      STOP PROCESS pRed;
      START PROCESS pYellow;
      STOP PROCESS pGreen;
      SET STATE delay10;
    ELSIF prev_light<<control2 , 0>><<control1 , 0>> = 0 THEN
      FOR alight<<control2 , 0>><<control1 , 0>> := 0 TO
NUMBER_OF_LIGHTS<<3>> DO
        rLightsArray[alight<<control2 , 0>><<control1 , 0>>] :=
FALSE;
      END_FOR
      STOP PROCESS pRed;
      STOP PROCESS pYellow;
```

```
                    START PROCESS pGreen;
                    SET STATE delay30;
                ELSIF prev_light<<control2 , 0>><<control1 , 0>> = 2 THEN
                    FOR alight<<control2 , 0>><<control1 , 0>> := 0 TO
NUMBER_OF_LIGHTS<<3>> DO
                        rLightsArray[alight<<control2 , 0>><<control1 , 0>>] :=
FALSE;
                    END_FOR
                    START PROCESS pRed;
                    STOP PROCESS pYellow;
                    STOP PROCESS pGreen;
                    SET STATE delay30;
                END_IF
            END_STATE
```