МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное
образовательное учреждение высшего образования
«Тульский государственный университет»

Политехнический институт
Кафедра «Промышленная автоматика и робототехника»

Утверждено на заседании кафедры
«Промышленная автоматика
и робототехника»
«17» января 2023 г., протокол № 2

И. О. заведующего кафедрой

_____О.А. Ерзин

Контрольная работа заочника

«Программирование автоматизированного оборудования»

основной профессиональной образовательной программы
высшего образования – программы бакалавриата

по направлению подготовки
15.03.04 Автоматизация технологических процессов и производств

с направленностью (профилем)
Автоматизация технологических процессов и производств

Формы обучения: очная, заочная

Идентификационный номер образовательной программы: 150304-01-23

Тула 2023 год

ЛИСТ СОГЛАСОВАНИЯ
рабочей программы дисциплины (модуля)

Разработчик:

Ерзин О.А., доцент, канд. техн. наук _____ _____
(ФИО, должность, ученая степень, ученое звание) (подпись)

Задание №1 ознакомиться с программой CoDeSys v2.3, научиться создавать проекты.

Выполнение

1. Создаем новый проект.
2. Настройка целевой платформы (Target Settings).

На страничке диалогового окна ’’Configuration’’ установите CoDeSys SP for Windows NT Realtime и подтвердите ввод – ОК (рисунок 1).

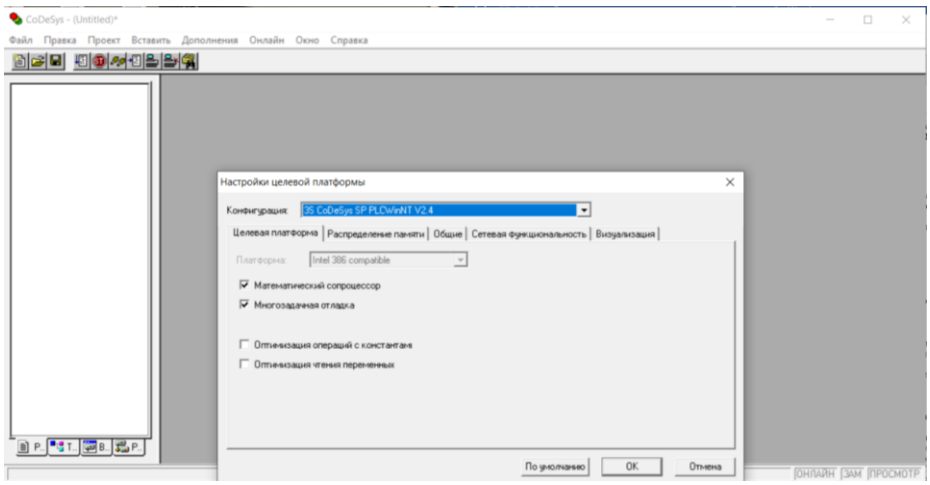

Рисунок 1 – Окно настройки целевой платформы

3. Главная программа PLC_PRG POU

Выберете язык реализации (language of the POU) FBD и сохраните предложенные по умолчанию тип компонента – программа (Type of the POU Program) и имя – Name PLC_PRG (рисунок 2).

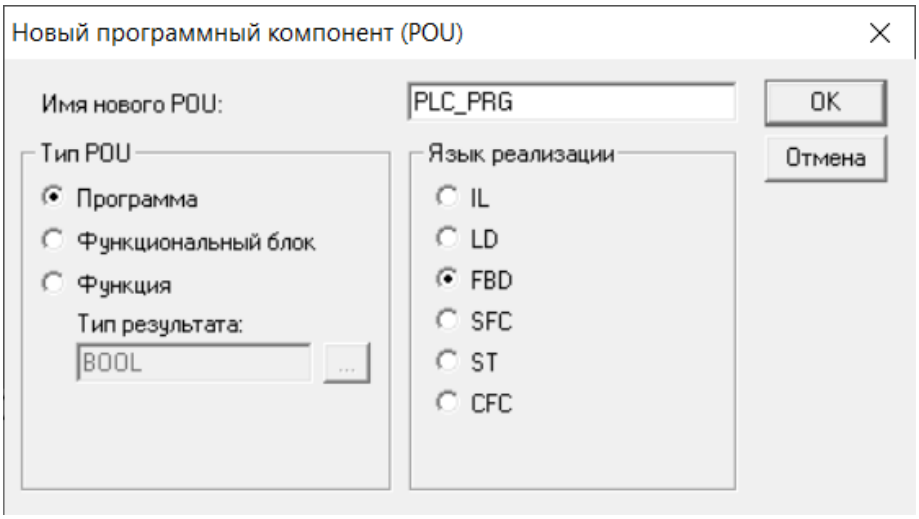

Рисунок 2 – Окно выбора языка реализации

4. Объявляем Переключатель подтверждения. В первой цепи графического FBD редактора выделите строку вопросов ??? и введите

наименование нашей первой переменной. Пусть это будет Observer (наблюдатель). Теперь нажмите на клавиатуре стрелку вправо. В появившемся диалоге определения переменной сохраните наименование (Name Observer) и логический тип (Type BOOL) (рисунок 3,4). Измените класс переменной (Class) на глобальный (VAR_GLOBAL). Подтвердите определение – ОК. Теперь определение переменной Observer должно появиться в окне глобальных переменных проекта (Global_Variables):

```
VAR_GLOBAL
        Observer:BOOL;
END_VAR
```

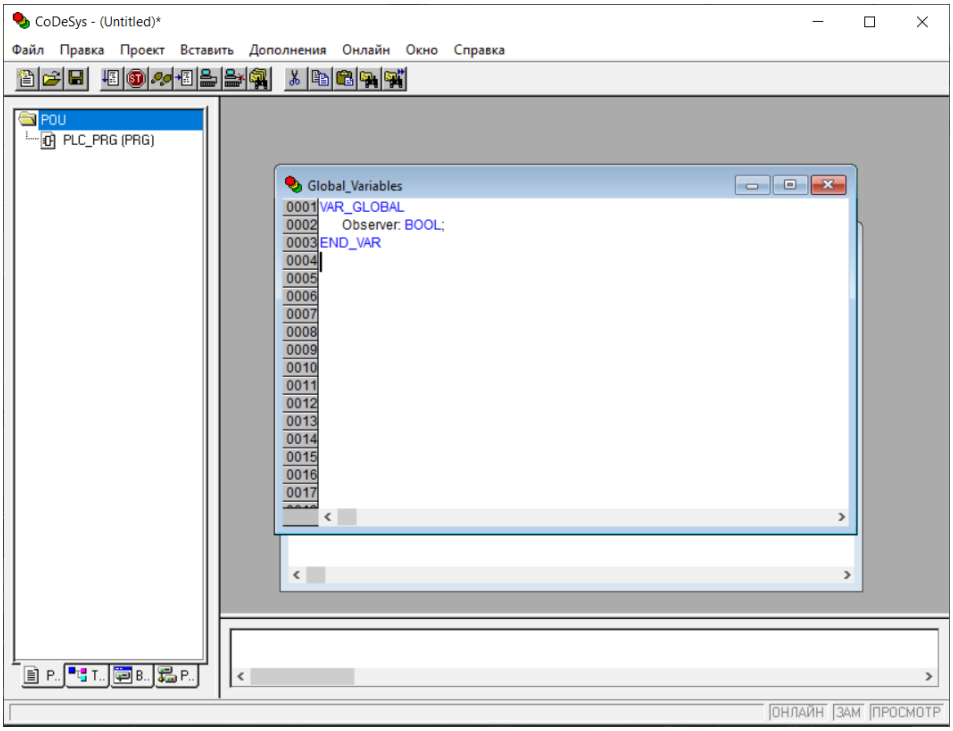

Рисунок 3 – Окно объявления переключателя подтверждения

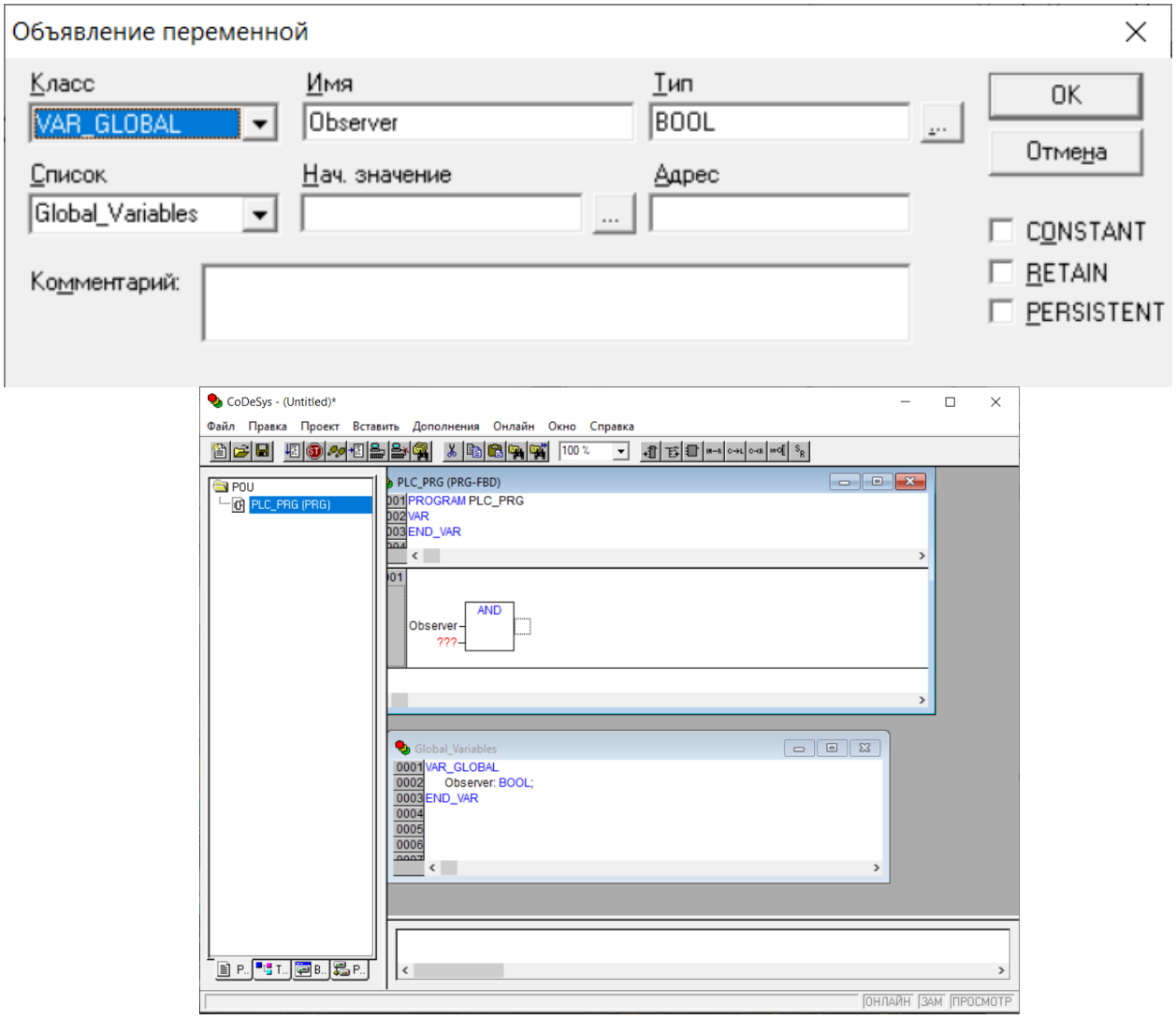

Рисунок 4 – Окно объявления переключателя подтверждения

5. Воспользуемся ассистентом ввода: нажмите клавишу F2. В диалоговом окне выбираем категорию: стандартные функциональные блоки. Из триггеров стандартной библиотеки выбираем R_TRIG. Он формирует логическую единицу по переднему фронту на входе (рисунок 5).

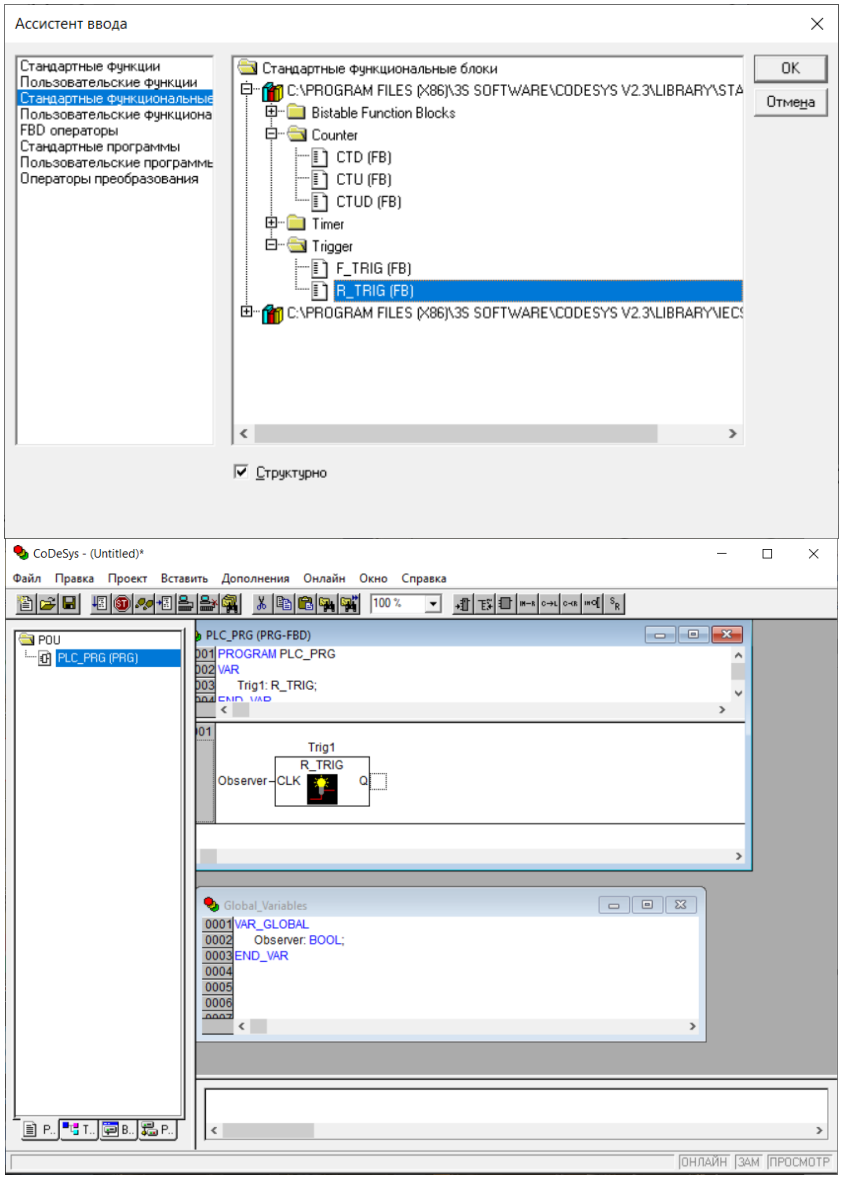

Рисунок 5 – Окно выбора триггера

6. Выделяем вход функционального блока триггера и вставляем элемент AND и переименовываем его в OR (рисунок 6).

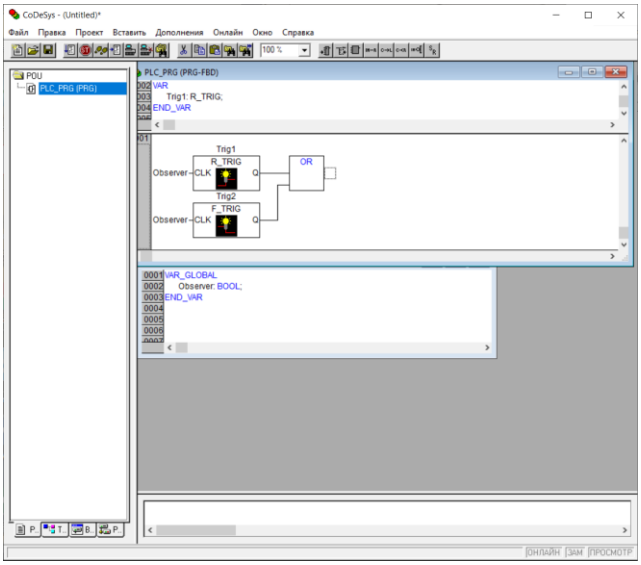

Рисунок 6 – Окно CoDeSys

7. Выделяем выход Q таймера Timer1 и в контекстном меню даём команду Assign (присвоить). Заменяем вопросы на имя переменной Warning. В диалоге определения задаем класс Class VAR_GLOBAL и тип BOOL (рисунок 7).

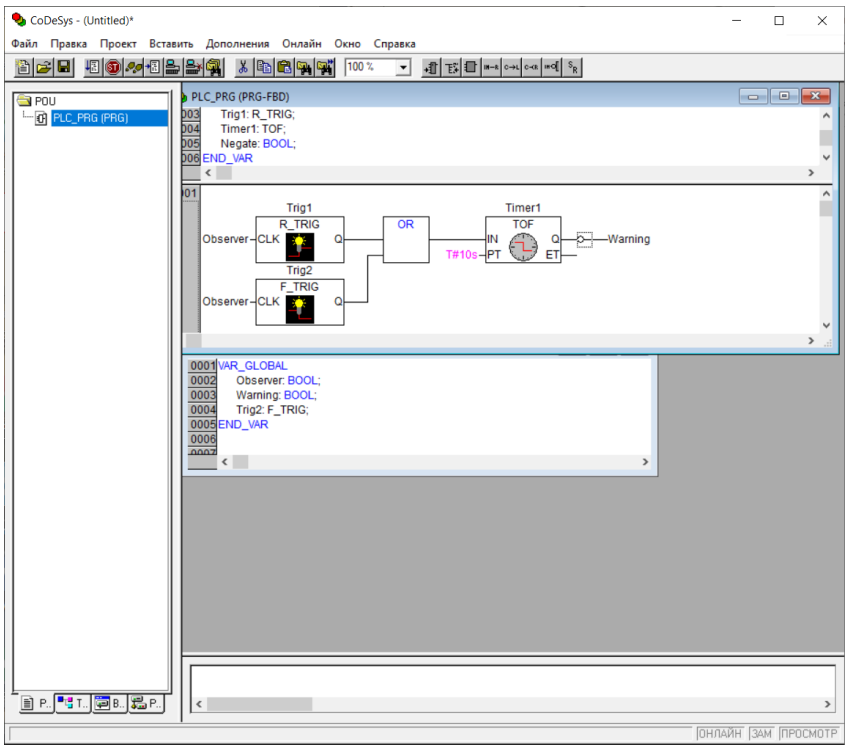

Рисунок 7 – Окно CoDeSys

8. Формируем Стоп Сигнал по второму интервалу времени.

Создаем новую цепь командой меню Insert-Network (after). Вставляем из стандартной библиотеки в новую цепь элемент Box типа TON под именем Timer2 (рисунок 8).

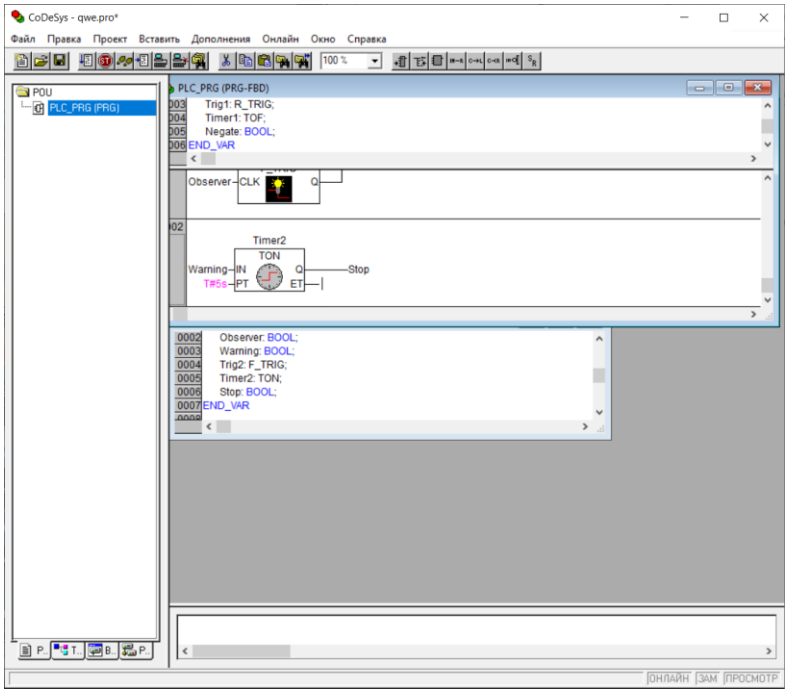

Рисунок 8 – Создание новой цепи

9. Вставляем POU управления механизмом. В левой части окна CoDeSys расположен организатор объектов POUs. Вставляем в контекстное меню новый программы компонент с именем Machine. По умолчанию создается пустая диаграмма, содержащая начальный шаг Init и соответствующий переход Trans0 заканчивающийся возвратом к Init (рисунок 9).

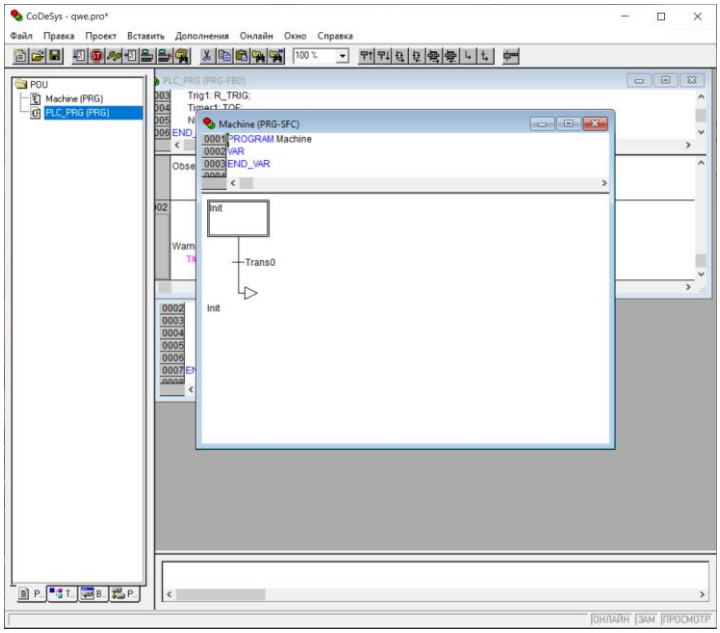

Рисунок 9 – Создание нового программного компонента с именем Machine

10. Программируем последующие шаги. Щелкаем дважды на шаге Go_Right. Программа начнет определение действия шага и попросит выбрать

язык его реализации. В этом шаге рабочий орган нашего механизма должен перемещаться по оси X вправо (рисунок 10).

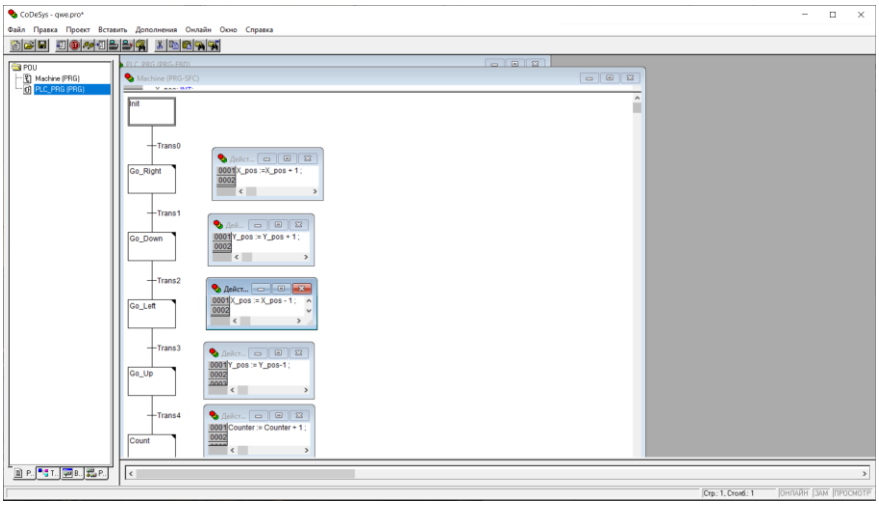

Рисунок 11 – Программирование шагов программы

11. Определяем переходы. Переход должен содержать условие, разрешающее переключение на следующий шаг. Переход после шага Init назовём Start. Следующий переход должен содержать условие X_Pos=100, так при значении позиции X включается следующая фаза движения (рисунок 12), остальные переходы аналогично.

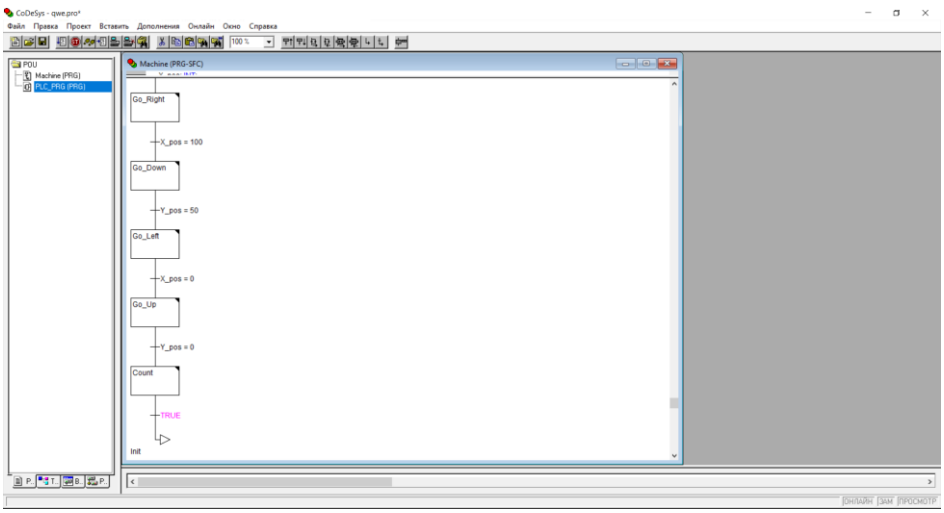

Рисунок 12 – Определение переходов

12. Останов механизма. Добавляем третью цепь. Вместо вопросов вставляем переменную Stop, а затем вставляем оператора Return (рисунок 13).

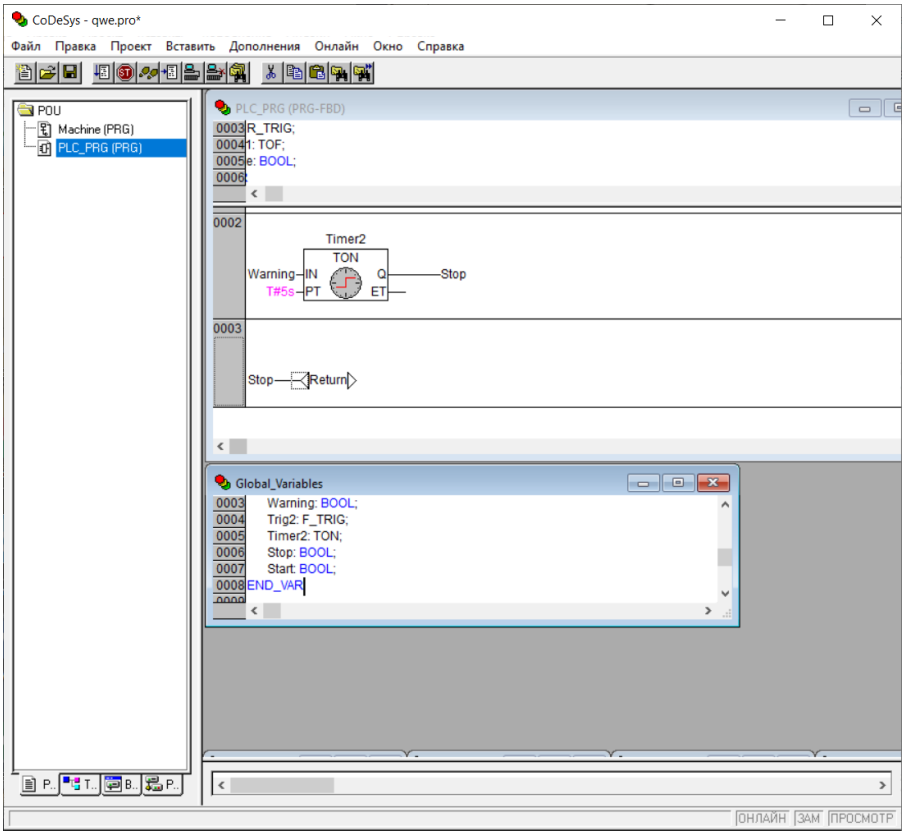

Рисунок 13 – Останов механизма

13. Вызов POU управления механизмов. Добавляем еще одну цепь, вставляем элемент Box. Нажимаем F2 и в ассистенте ввода задаём POU управления механизмом в категории пользовательских программ.

14. Компиляция проекта. Откомпилируем проект целиком командой меню Progect-Rebuild или клавишей F11.

15. Создаем визуализацию (рисунок 14).

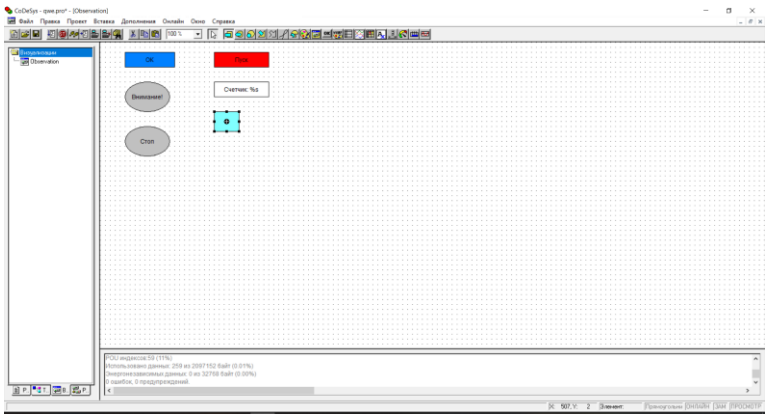

Рисунок 14 – Создание визуализации

Далее представлен результат работы (рисунок 15).

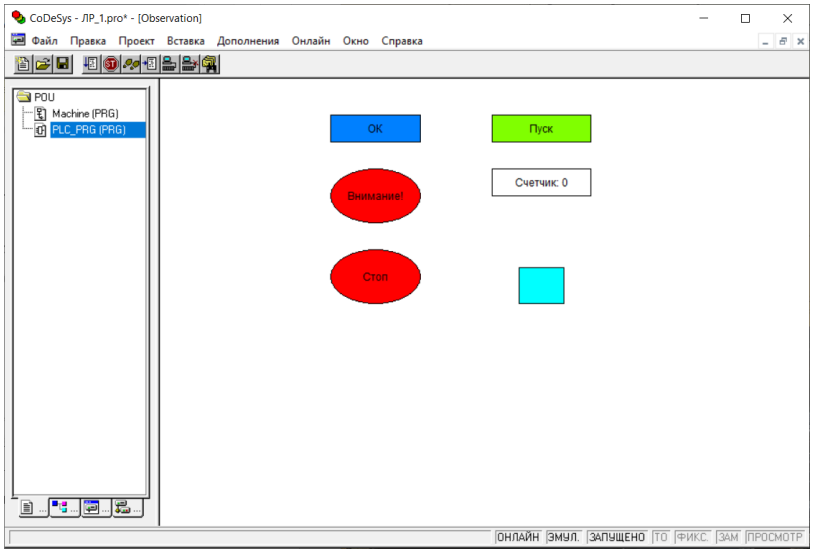

Рисунок 15 – Результат работы

Содержание отчета.

1. Представить описание выполненной работы
2. Сделать выводы

Задание №2 ознакомиться с программой CoDeSys v2.3, научиться создавать проекты.

Блок управления светофором

1. Создайте POU. В окне диалога определим первый POU. По умолчанию он получает наименование PLC_PRG. Не изменяйте его. В качестве языка программирования данного POU выбираем язык CFC (рисунок 1).

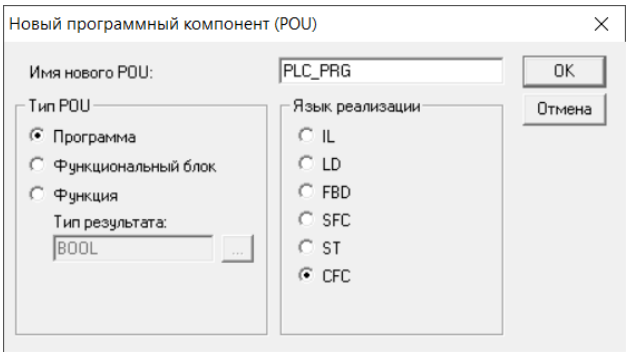

Рисунок 1 – Окно создания нового POU

2. Создаем еще три объекта. Создаём программу на языке SFC с именем SEQUENCE, функциональный блок на языке FBD с именем TRAFFICSIGNAL и еще один аналогичный блок – WAIT, который мы будем описывать на языке IL (рисунок 2,3,4).

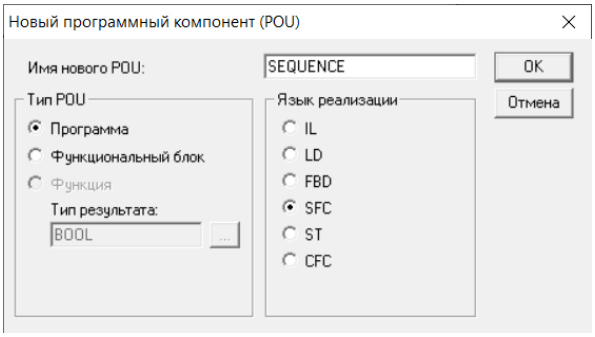

Рисунок 2 – Окно создания 1-ого объекта

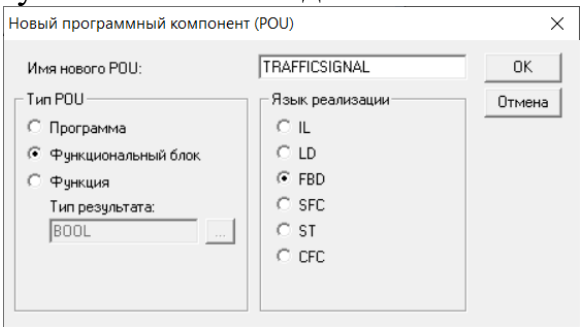

Рисунок 3 – Окно создания 2-ого объекта

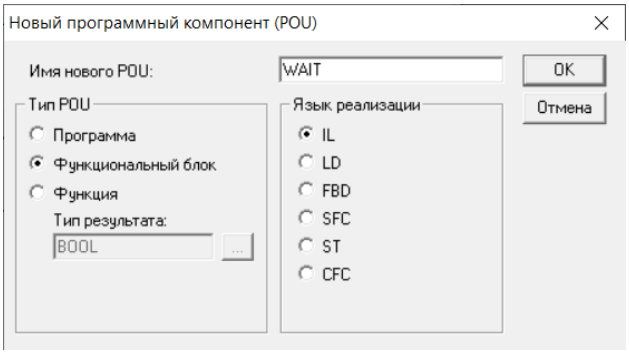

Рисунок 4 – Окно создания 3-ого объекта

3. Возвращаемся к POU TRAFFICSIGNAL. В редакторе объявлений определяем входную переменную по имени STATUS типа INT. STATUS будет иметь четыре возможных состояния, определяющие соответствующее стадии – зеленая, желтая, желто-красная и красная (рисунок 5).

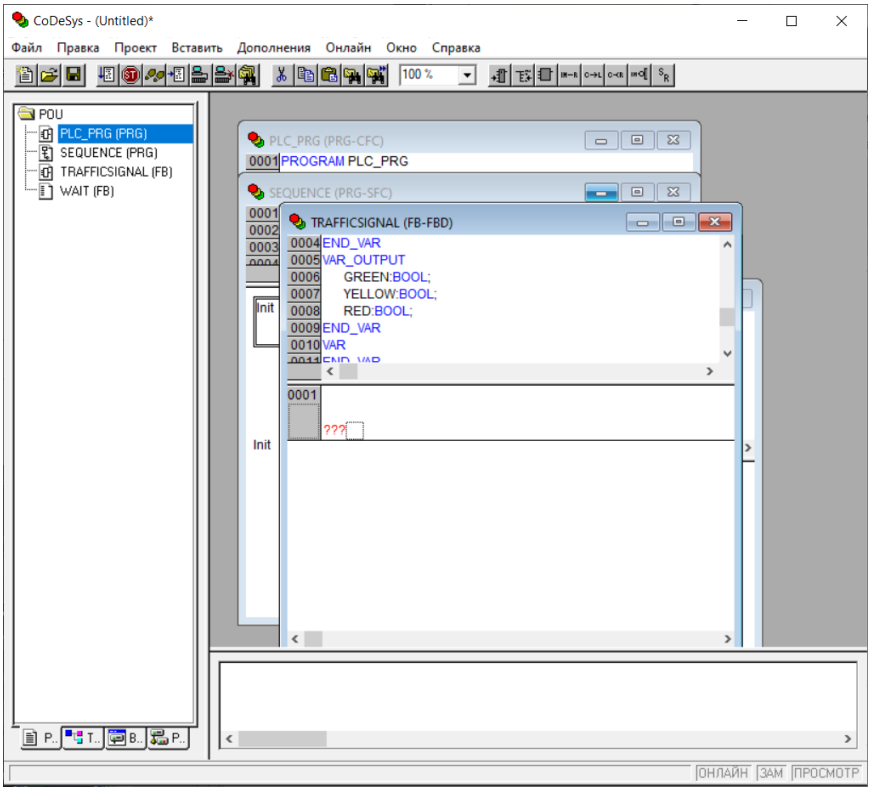

Рисунок 5 – Окно редактора объявлений

4. Программируем TRAFFICSIGNAL. В первой цепи будет вставлен прямоугольник с оператором AND и двумя входами. Щелкните мышкой на тексте AND и замените его на EQ. Три знака вопроса около верхнего из двух входов замените на имя переменной STATUS. Для нижнего входа вместо трех знаков вопроса нужно поставить 1. Щелкните теперь на место позади прямоугольника EQ. Теперь выбран выход EQ. Выполните команду меню

(Вставка – Присваивание). Измените три вопроса *???* на GREEN. Таким образом, GREEN будет включен, когда STATUS равен 1.

Для других цветов нам понадобятся еще две цепи. Создаете их командой Вставка – Цепь (после). Законченный POU должен выглядеть следующим образом (рисунок 6).

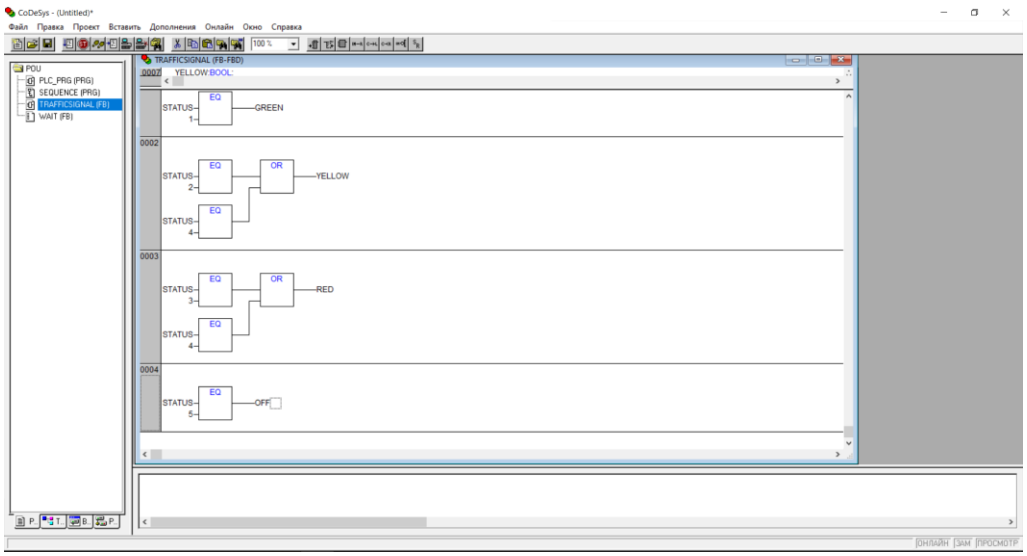

Рисунок 6 – Окно программирования TRAFFICSIGNAL

5. Объявления WAIT. POU будет работать таймером. POU должен иметь входную переменную TIME типа TIME и генерировать на выходе двоичную переменную, которую назовём ОК.

Предварительно устанавливаем эту переменную в FALSE в конце строки объявления. Далее нужен генератор времени POU TP. Он имеет два входа (IN и PT) и два входа (Q и ET). Чтобы использовать TP в POU WAIT, мы должны создать его локальный экземпляр. Для этого объявляем локальную переменную ZAB типа TP (рисунок 7).

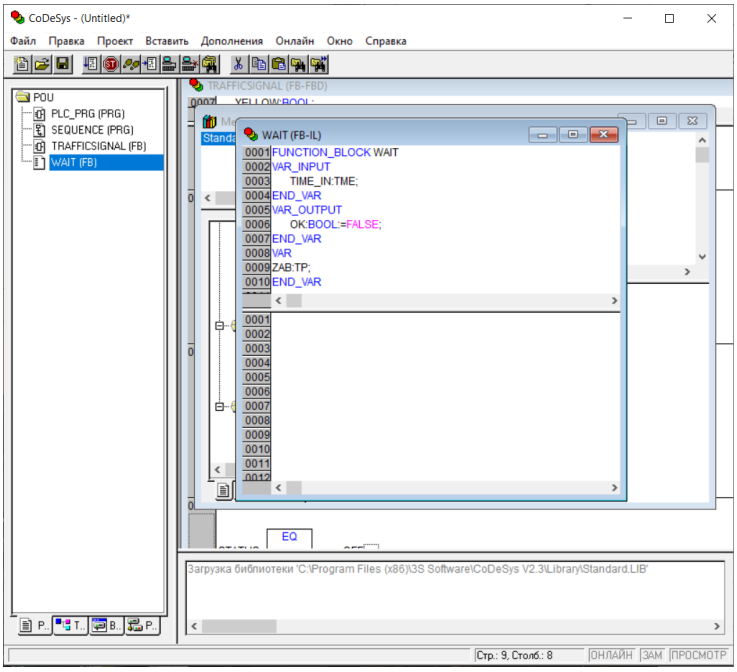

Рисунок 7 – Окно объявления WAIT

6. Программируем WAIT. Текст программы для создания желаемого таймера приведён на рисунке 8.

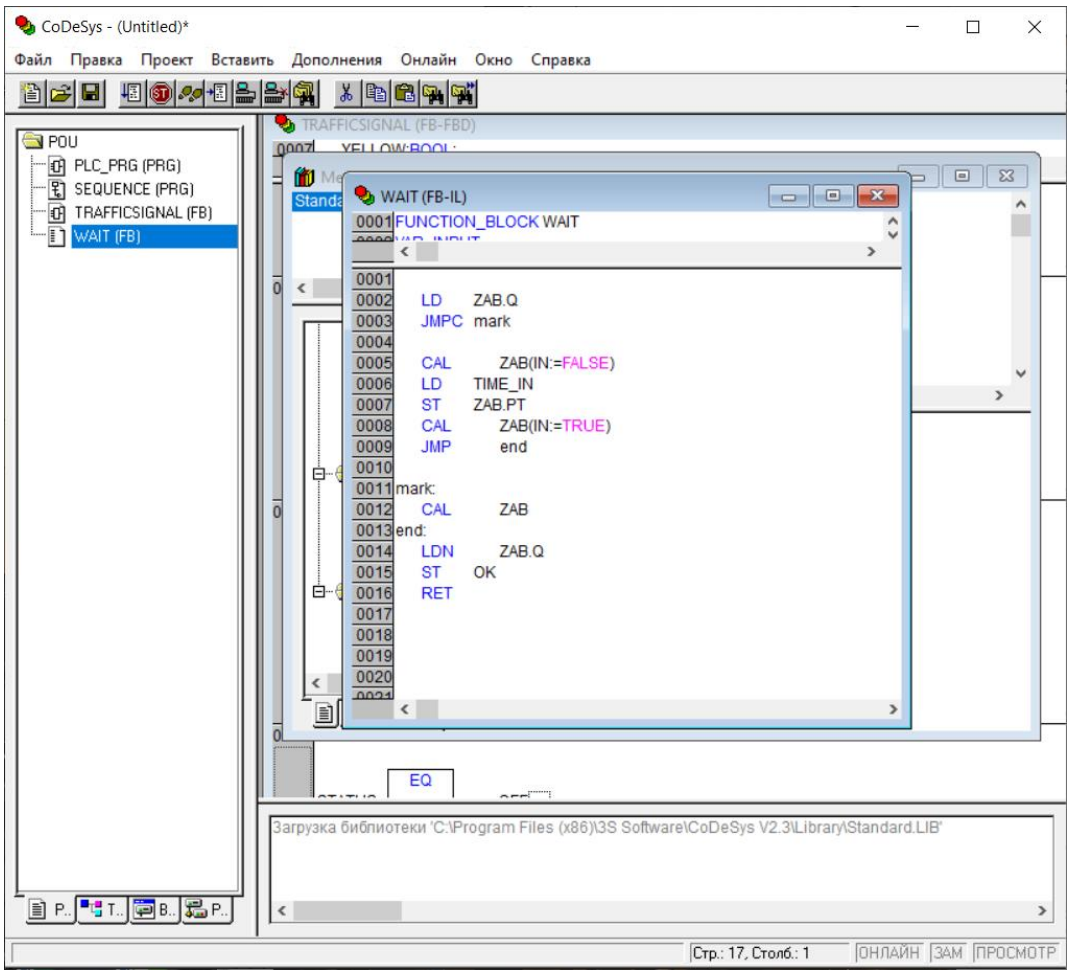

Рисунок 8 – Окно программирования WAIT

7. Объединение двух блоков WAIT и TRAFFICSGNAL в главной программе PLC_PRG. Сначала объявляем необходимые переменные. Это

входная переменная START типа BOOL, две выходных переменные TRAFFICSIGNAL1 и TRAFFICSIGNAL2 типа INT и одна типа WAIL (рисунок 9).

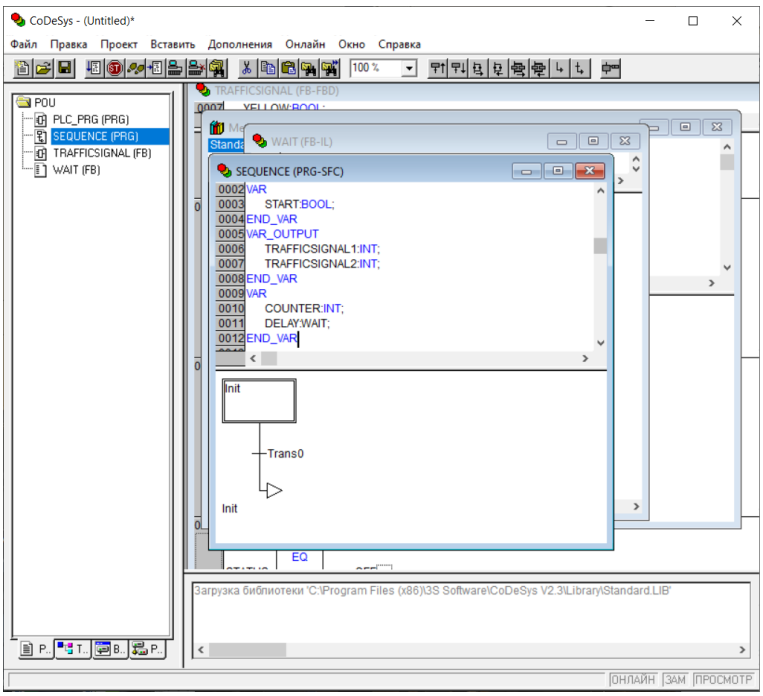

Рисунок 9 – Окно объединения двух блоков WAIT и TRAFFICSGNAL

8. Создаем SFC диаграмму. Прежде чем программировать конкретные этапы и переходы, выстроим структуру графа. Сначала нам понадобятся этапы для каждой стадии TRAFFICSIGNAL. Вставьте их, отмечая Trans0 и выбирая команды Вставка – Шаг-переход (снизу). Повторяем процедуру 3 раза. Для редактирования названия перехода или этапа нужно просто щелкнуть мышкой на нужном тексте. Назовите первый после Init START, а все прочие переходы DELAY.OK (рисунок 10).

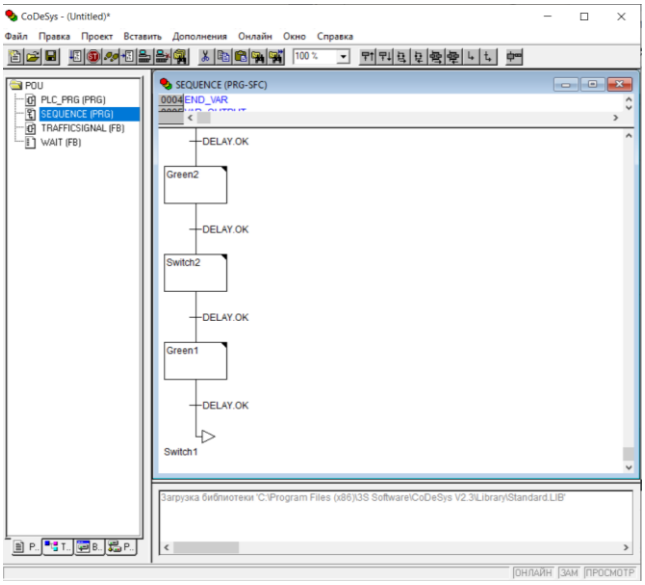

Рисунок 10 – Окно SFC диаграммы

9. Программирование этапов и переходов. Во время действия этапа Init проверяем, активен сигнал включения START или нет. Если сигнал активен, то светофор выключается. Этого можно достичь, если записать в переменные TRAFFICSIGNAL1 и TRAFFICSIGNAL2 число 5 (рисунок 11). Далее аналогично записываем в переменные (рисунок 12-14).

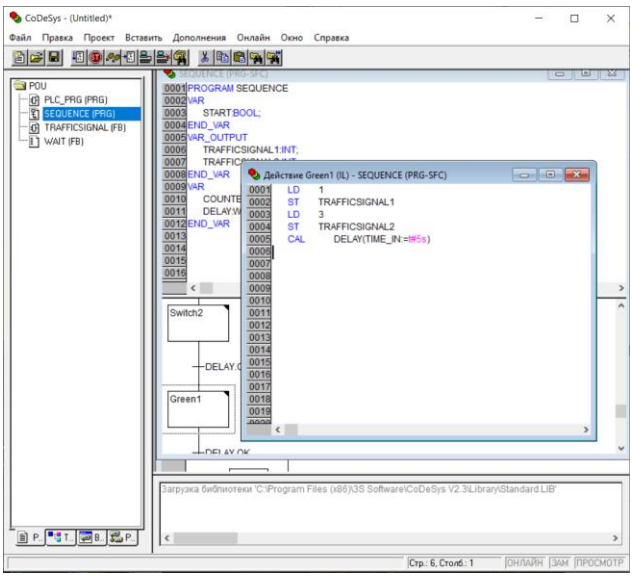

Рисунок 11 – Окно программирования 1-ого этапа

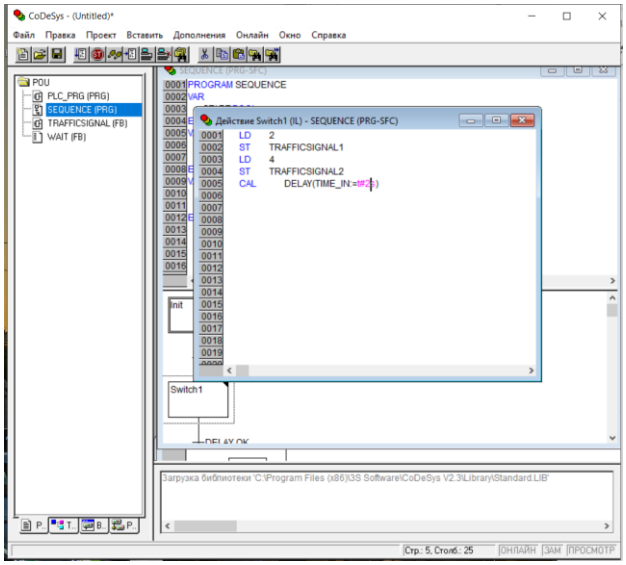

Рисунок 12 – Окно программирования 2-ого этапа

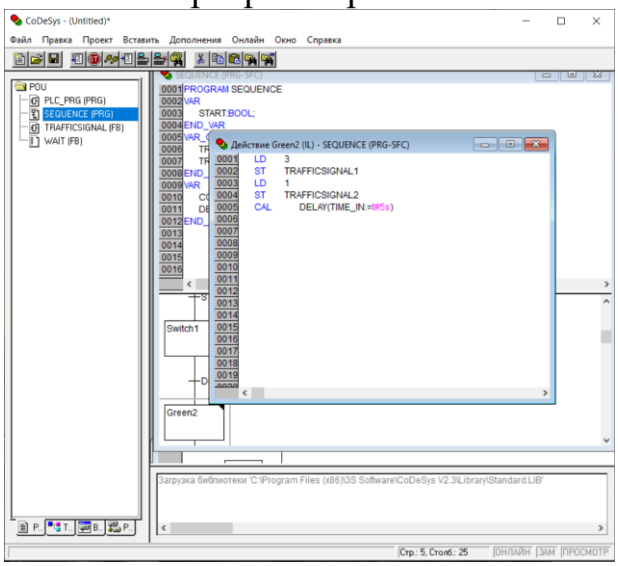

Рисунок 13 – Окно программирования 3-ого этапа

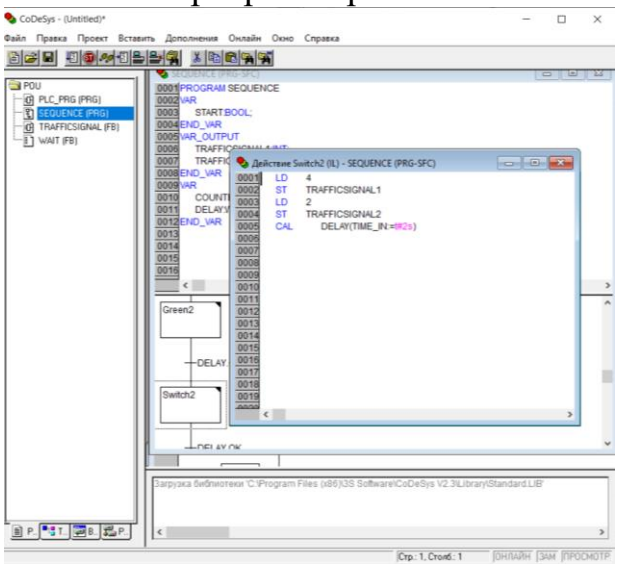

Рисунок 14 – Окно программирования 4-ого этапа

10. Теперь немного усложним программу. Разумно будет выключать светофоры на ночь. Для этого мы создадим в программе счетчик, который

после некоторого числа циклов произведет отключение устройства. Для начала нам нужна новая переменная COUNTER ТИПА INT.

Теперь выбираем переход после Switch1 и вставьте ещё один этап и переход. Выберете результирующий переход и вставьте альтернативную ветвь вправо. После левого перехода вставьте удаленный переход на Init. Теперь новые части должны выглядеть как фрагмент, представленный на рисунке 15.

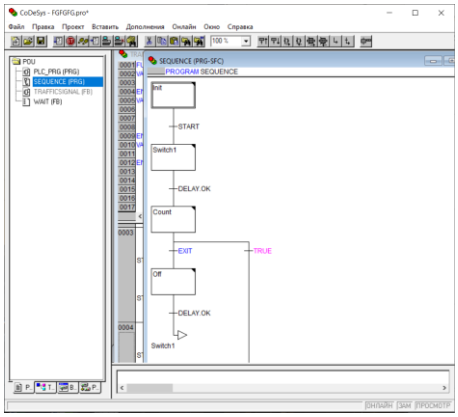

Рисунок 15 – Окно созданного фрагмента программы

11. Результат. Необходимо распределить входные и выходные переменные в блоке PLC_PRG. Объявим соответствующие Boolean переменные для всех шести выходов и одного входа, затем создадим программу и сопоставим переменные соответствующим IEC адресам. Для представления шести ламп светофоров нужно 6 переменных типа Boolean (рисунок 16).

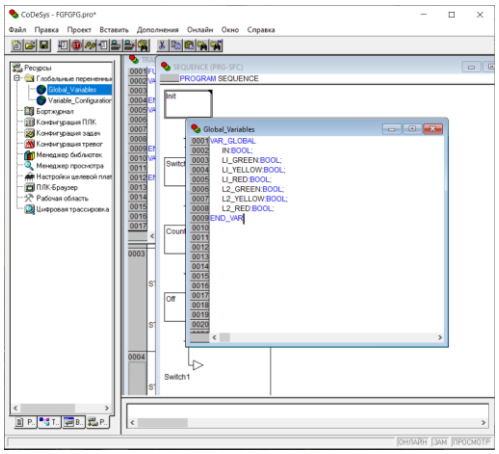

Рисунок 16 – Окно программирования светофора

12. Переходим в окно редактора, мы выбрали редактор Непрерывных Функциональных Схем, и, следовательно, нам доступна соответствующая панель инструментов. Наша программа должна принять вид, представленный на рисунке 17.

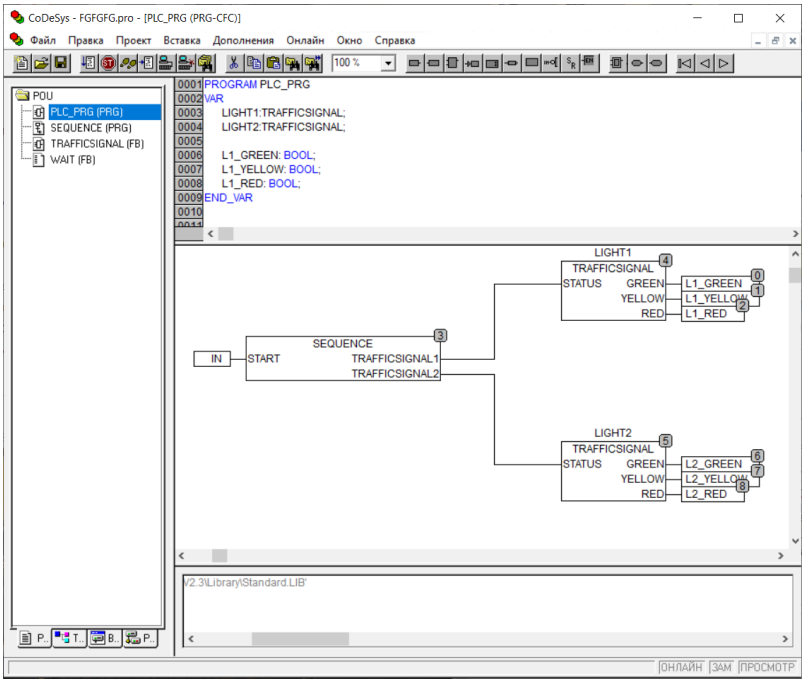

Рисунок 17 – Окно редактора

13. Создание новой визуализации. Для этого выбираем вкладку Визуализация в организаторе объектов. В конечном итоге получаем следующую конструкцию (рисунок 18).

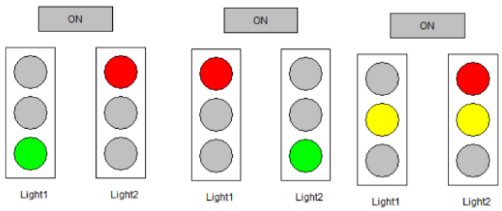

Рисунок 18 – Результаты программы

Содержание отчета.

1. Представить описание выполненной работы
2. Оформить пояснительную записку с описанием выполнения заданий и скриншотами экрана программы.