ФЕДЕРАЛЬНОЕ АГЕНТСТВО СВЯЗИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«ПОВОЛЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ТЕЛЕКОММУНИКАЦИЙ И ИНФОРМАТИКИ»

Кафедра «Программное обеспечение и управление в технических
системах»

В.С. Сивков

АВТОМАТИЗИРОВАННЫЕ И ИНФОРМАЦИОННО-УПРАВЛЯЮЩИЕ СИСТЕМЫ

Методические указания по выполнению лабораторных работ

Самара
2018

УДК 681.5

Рекомендовано к изданию методическим советом ПГУТИ, протокол No , от 00.02.2018 г.

Сивков Вадим Сергеевич
Автоматизированные и информационно-управляющие системы: **м**етодические указания по выполнению лабораторных работ / В.С. Сивков — Самара: ПГУТИ, 2018 – 36 с.

Методические указания по выполнению лабораторных работ Автоматизированные и информационно-управляющие системы, разработаны в соответствии с ФГОС ВО по направлению подготовки 27.03.04 «Управление в технических системах», и предназначены для студентов 4 курса факультета ИСТ ПГУТИ.

Содержание

Введение

Настоящие методические указания для проведения лабораторных занятий предназначены для студентов дневной формы обучения, обучающихся по программе подготовки бакалавров по направлению 27.03.04 «Управление в технических системах».

Целью проведения лабораторных занятий является закрепление и расширение практических навыков по дисциплине, а также развитие способностей самостоятельной работы с технической и справочной документацией по программированию промышленных контроллеров.

Лабораторный практикум рассчитан на выполнение шести работ:

- «Программирование простых цепей управления на языке LD»;
- «Составление программы управления технологическим процессом на языке LD»;
- «Работа с таймерами и счетчиками на языке LD»;
- «Составление программы управления технологическим процессом (автоматический миксер) на языке LD»;
- «Основы работы в среде CoDeSys на языке ST»;
- «Составление программы управления лифтом на языке ST».

Первые четыре работы посвящены составлению программ для программируемых логических контроллеров (ПЛК) на языке программирования Ladder Diagram (LD). Для удобства выполнения работ вместо аппаратных контроллеров используется симулятор ПЛК — PSIM, который позволяет студентам не только создавать программы на языке релейной логики, но и оценить результат своей работы с помощью системы визуализации. Пятая и шестая работа выполняется в среде программирования CoDeSys. Данные работы посвящены составлению программ для ПЛК на языке Structured Text (ST).

Лабораторная работа №1 - Программирование простых цепей управления на языке LD.

Цель работы: Знакомство с приемами программирования ПЛК, создание простых цепей управления систем автоматизации.

Аппаратное и программное обеспечение: PC, ОС Linux/Windows, PSIM.

Для выполнения данной лабораторной работы необходим симулятор ПЛК PSIM.

Источники информации:
http;//www.google.ru
http://thelearningpit.com/plc/psim/doc/index.html

Контрольные вопросы:

1. Что такое промышленный контроллер.
2. Что такое ПЛК.
3. Особенности работы промышленных контроллеров.
4. Особенности языка программирования LD.
5. Структура программы на языке LD.
6. Основные элементы языка LD.
7. Реализация логических операций И, ИЛИ, НЕ на языке LD.

Содержание работы.

Задание 1. В эмуляторе ПЛК PSIM (режим I/O Simulator) составить типовую программу управления электродвигателем (Рисунок 1):

- однократное нажатие кнопки «Вкл» включает нагрузку (загорается сигнальная лампа включенной нагрузки),
- однократное нажатие кнопки «Выкл» выключает нагрузку (сигнальная лампа включенной нагрузки гаснет).

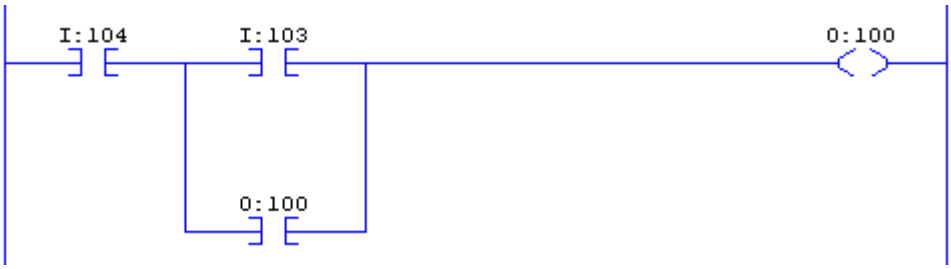

Рисунок 1

Добавить индикацию выключенного состояния (выход O:101). В режиме включенной нагрузки добавить индикацию нажатия кнопки на входе I:105 (должна загораться лампа на выходе O:102).

Зарисовать схему, продемонстрировать работу программы преподавателю.

*Задание 2***.** Составить программу управления одной нагрузкой (лампа выхода O:100) с двух выключателей (вход I:100 и I:101). Изменение положения любого выключателя должно менять состояние лампы на противоположное текущему (проходной выключатель).

Зарисовать схему, продемонстрировать работу программы преподавателю.

Задание 3. В соответствии с таблицей, составить программу управления нагрузками.

№ варианта	Задание
1	• когда I00 разомкнут — горят лампы O01 — O07, • когда I00 замкнут — горит лампа O00, • в любом положении I00, при нажатии I02 должны загораться лампы, подключенные к выходам с нечетными номерами.
2	• когда I01 разомкнут — горят лампы O02 — O07, • когда I01 замкнут — горит лампа O01, • в любом положении I01, при нажатии I04 должны загораться лампы, подключенные к выходам с четными номерами.
3	• когда I03 разомкнут — горят лампы O03 — O07, • после однократного нажатия I03 должны загораться лампы O00 - O02, • при замкнутом I00, после однократного нажатия I03 должны загораться лампы O05 - O07,
4	• когда I01 разомкнут — горят лампы O03 — O07, • когда I01 замкнут — горит лампа O03, • в любом положении I01, при нажатии I04 должны загораться лампы, подключенные к выходам с четными номерами.
5	• когда I04 не нажата — горят лампы O04 — O07, • после однократного нажатия I04 должны загораться лампы O00 - O04, • при замкнутом I01, после однократного нажатия I03 должны

	загораться лампы O06 - O07,
6	• когда I06 разомкнут — горят лампы O00 — O07, • когда I06 замкнут — горят лампы O02 - O05, • в любом положении I06, при нажатии I05 должны загораться лампы, подключенные к выходам с нечетными номерами.
7	• когда I01 разомкнут — горят лампы O02 — O07, • когда I01 замкнут — горит лампа O01, • в любом положении I01, при нажатии I04 должны загораться лампы, подключенные к выходам с четными номерами.
8	• когда I05 не нажата — горят лампы O02 — O07, • после однократного нажатия I05 должны загораться лампы O00 - O07, • при замкнутом I00, после однократного нажатия I05 должны загораться лампы O04 - O06,
9	• когда I07 разомкнут — горят лампы O00 — O07, • когда I07 замкнут — горит лампа O06, • в любом положении I07, при нажатии I05 должны загораться лампы, подключенные к выходам с нечетными номерами.
10	• когда I03 не нажата — горят лампы O00 — O07, • после однократного нажатия I03 должны загораться лампы O05 — O07, • при замкнутом I01, после однократного нажатия I03 должны загораться лампы O01 - O06,

Зарисовать схему, продемонстрировать работу программы преподавателю.

Содержание отчета.

1. Титульный лист с указанием номера и наименования работы, ф.и.о. студента, номера учебной группы.
2. Программа — схема на языке LD, в соответствии с заданием 1.
3. Программа — схема на языке LD, в соответствии с заданием 2.
4. Программа — схема на языке LD, в соответствии с заданием 3.

Лабораторная работа №2 - Составление программы управления технологическим процессом на языке LD

Цель работы: Знакомство с приемами программирования ПЛК, создание простых цепей управления систем автоматизации.

Аппаратное и программное обеспечение: PC, ОС Linux/Windows, PSIM.

Для выполнения данной лабораторной работы необходим симулятор ПЛК PSIM.

Источники информации:
http://www.google.ru
http://thelearningpit.com/plc/psim/doc/index.html

Контрольные вопросы:

1. Что такое промышленный контроллер.
2. Что такое ПЛК.
3. Особенности работы промышленных контроллеров.
4. Особенности языка программирования LD.
5. Структура программы на языке LD.
6. Основные элементы языка LD.
7. Реализация логических операций И, ИЛИ, НЕ на языке LD.
8. Реализация схемы «старт-стоп» на языке LD.

Содержание работы.

Задание 1. В эмуляторе ПЛК PSIM (режим Silo Simulator) составить программу автоматизации технологического процесса (Рисунок 2):

- однократное нажатие кнопки «START» (клавиша F2 в симуляторе — эмулирует нормально разомкнутую кнопку) включает двигатель конвеера (также должна загореться сигнальная лампа «RUN» на панели индикации),
- однократное нажатие кнопки «STOP» (клавиша F1 в симуляторе — эмулирует нормально замкнутую кнопку) выключает двигатель конвеера (также должна загореться сигнальная лампа «STANDBY» на панели индикации, а лампа «RUN» должна погаснуть),
- при попадании движущейся коробки в зону фото-датчика (Photo Switch) транспортерная лента останавливается, и происходит автоматическое наполнение коробки (необходимо использовать электромагнитный клапан Solenoid и индикатор уровня Level), после заполнения коробки должна загораться лампа «FULL» и гореть до тех пор, пока коробка не покинет зону действия фото-датчика.

Сохранить схему, продемонстрировать работу схемы преподавателю.

Задание 2. В соответствии с таблицей вариантов модернизировать программу.

№ варианта	Задание
1	• После наполнения контейнера останавливать работу и ждать ручного запуска
2	• Перед заполнением контейнера останавливать работу и ждать ручного запуска
3	• Добавить возможность отправлять по линии частично заполненный контейнер (ручной режим остановки/запуска)
4	• Запускать мотор с 5 секундной задержкой после нажатия кнопки запуска
5	• Заполнять контейнеры через один
6	• Заполнять каждый третий контейнер
7	• Заполнять каждый контейнер приблизительно на половину от уровня срабатывания датчика FULL
8	• Каждые 2 секунды останавливать работу двигателя на 1 секунду
9	• Выключать работу соленоида после наполнения 3х контейнеров
10	• Через 10 секунд работы переходить в режим «только транспортировка» (без заполнения контейнеров)

п

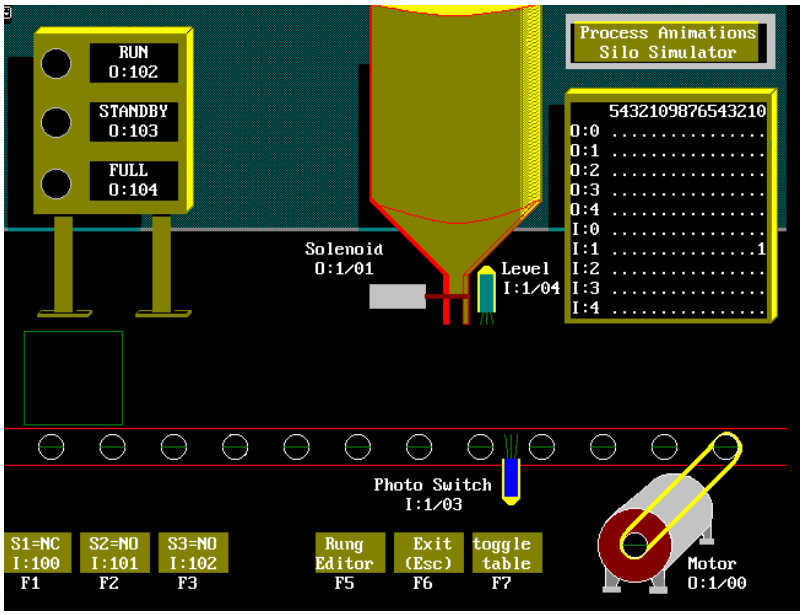

Рисунок 2

Содержание отчета.

1. Титульный лист с указанием номера и наименования работы, ф.и.о. студента, номера учебной группы.
2. Программа — схема на языке LD, в соответствии с заданием 1.
3. Программа — схема на языке LD, в соответствии с заданием 2.

Лабораторная работа №3 - Работа с таймерами и счетчиками на языке LD.

Цель работы: Знакомство с приемами программирования ПЛК, создание простых программ управления с использованием таймеров.

Аппаратное и программное обеспечение: PC, ОС Linux/Windows, PSIM.

Для выполнения данной лабораторной работы необходим симулятор ПЛК PSIM.

Источники информации:

http://www.google.ru
http://thelearningpit.com/plc/psim/doc/index.html

Контрольные вопросы:

1. Что такое промышленный контроллер.
2. Что такое ПЛК.
3. Особенности работы промышленных контроллеров.
4. Особенности языка программирования LD.
5. Реализация таймеров на языке LD.
6. Инструкции таймеров EN DN RST.
7. Что такое таблица таймеров ПЛК.
8. Реализация счетчиков на языке LD.
9. Что такое таблица счетчиков ПЛК.

Содержание работы.

Задание 1. В эмуляторе ПЛК PSIM (режим Traffic Light) составить программу управления светофором (Рисунок 3):

- необходимо реализовать два режима работы светофора: ночной режим (с обоих сторон светофор мигает желтым цветом), и автоматический режим,
- в автоматическом режиме последовательность переключения сигналов показана на рисунке 4,
- реализовать переключение между автоматическим и ночным режимами работы.

Зарисовать схему, продемонстрировать работу схемы преподавателю.

Задание 2. Изменить программу таким образом, чтобы можно было в автоматическом режиме фиксировать красный или зеленый сигнал светофора (стороны должны работать в противофазе), реализовать возврат в автоматический режим.

Зарисовать схему, продемонстрировать работу схемы преподавателю.

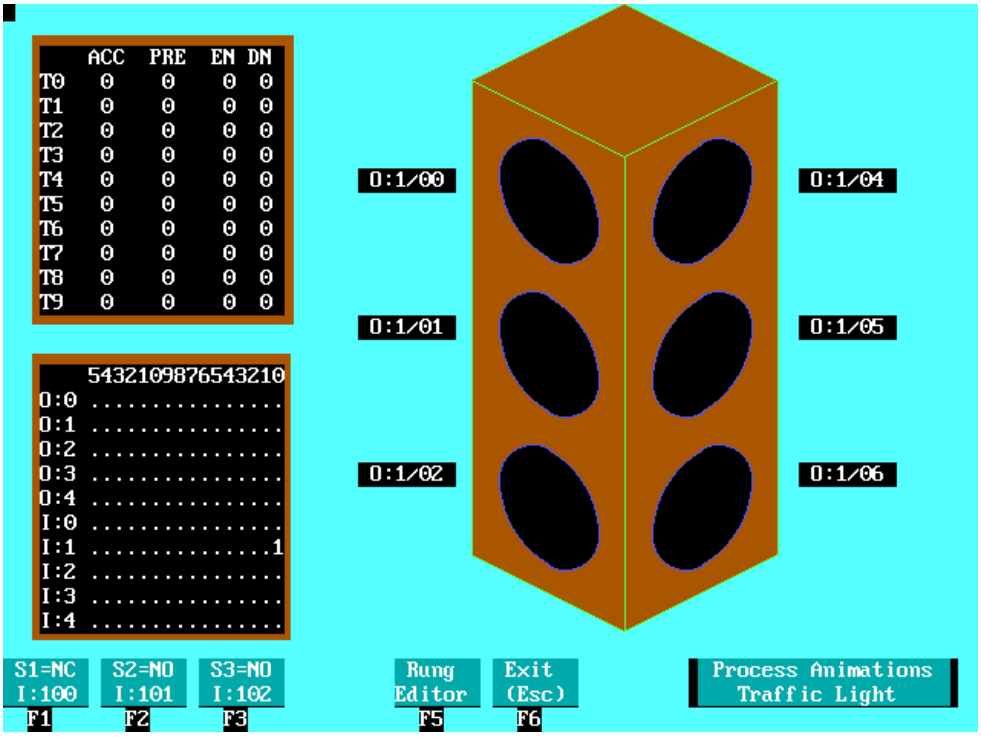

Рисунок 3

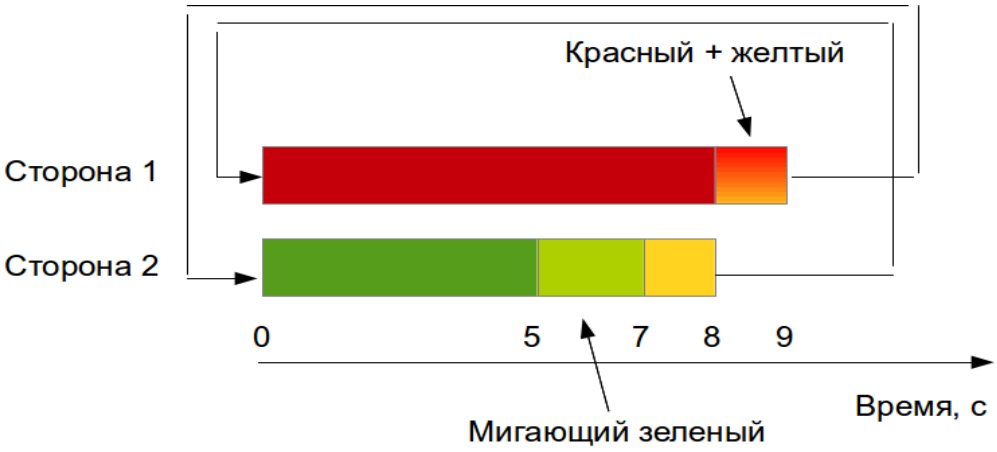

Рисунок 4

Задание 3. В соответствии с таблицей вариантов модернизировать программу.

№ варианта	Задание
1	• Красный горит 3 с, зеленый мигает 3 раза в течении 6 секунд.
2	• В автоматическом режиме желтый мигает 2 раза в течении 4 секунд
3	• После запуска программы включать все лампы светофора на 5 секунд, далее переходить в автоматический режим работы
4	• Запускать программу с ночного режима
5	• В автоматическом режиме желтый мигает 4 раза в течении 8 секунд
6	• После 30 секунд работы переходить в ночной режим
7	• При переключении в ночной режим работы возвращаться в автоматический режим через 10 с
8	• После запуска программы мигнуть 3 раза желтым, далее стандартный алгоритм работы
9	• Красный мигает с периодом 2 с
10	• После 10 переключений в автоматическом режиме переходить в ночной режим

Содержание отчета.

1. Титульный лист с указанием номера и наименования работы, ф.и.о. студента, номера учебной группы.
2. Программа — схема на языке LD, в соответствии с заданиями 1-3.

Лабораторная работа №4 - Составление программы управления технологическим процессом (автоматический миксер) на языке LD

Цель работы: Знакомство с приемами программирования ПЛК, создание программы управления автоматическим миксером на языке LD.

Аппаратное и программное обеспечение: PC, ОС Linux/Windows, PSIM.

Для выполнения данной лабораторной работы необходим симулятор ПЛК PSIM.

Источники информации:

http://www.google.ru
http://thelearningpit.com/plc/psim/doc/index.html

Контрольные вопросы:

1. Что такое промышленный контроллер.
2. Что такое ПЛК.
3. Особенности работы промышленных контроллеров.
4. Особенности языка программирования LD.
5. Реализация таймеров на языке LD.
6. Инструкции таймеров EN DN RST.
7. Что такое таблица таймеров ПЛК.
8. Реализация счетчиков на языке LD.
9. Что такое таблица счетчиков ПЛК.

Содержание работы.

Задание 1. В эмуляторе ПЛК PSIM (режим Batch Mixer) составить программу управления технологической установкой смешивания двух компонентов (Рисунок 5):

- при нажатии кнопки S2 должны запускаться помпы P1 и P2. Пропорции и общая степень заполнения контролируется с помощью импульсных датчиков потока FL1 и FL2, установленных на трубопроводах. Уровень жидкости в резервуаре также контролируется двумя датчиками уровня;
- при уровне срабатывания датчика Hi-Level, остановить помпы P1 и P2 и включить нагреватель O:1/04 и миксер O:1/00;
- после 4х секунд работы нагревателя и двигателя, включить помпу P3, выключить нагреватель и двигатель, и продолжать опустошать резервуар, до момента, когда перестает срабатывать датчик Lo-Level. Далее необходимо выключить все исполнительные устройства и дожидаться нажатия кнопки S2;

– кнопка S1 должна выключать все исполнительные устройства, кроме помпы P3, и незамедлительно полностью опустошать резервуар;

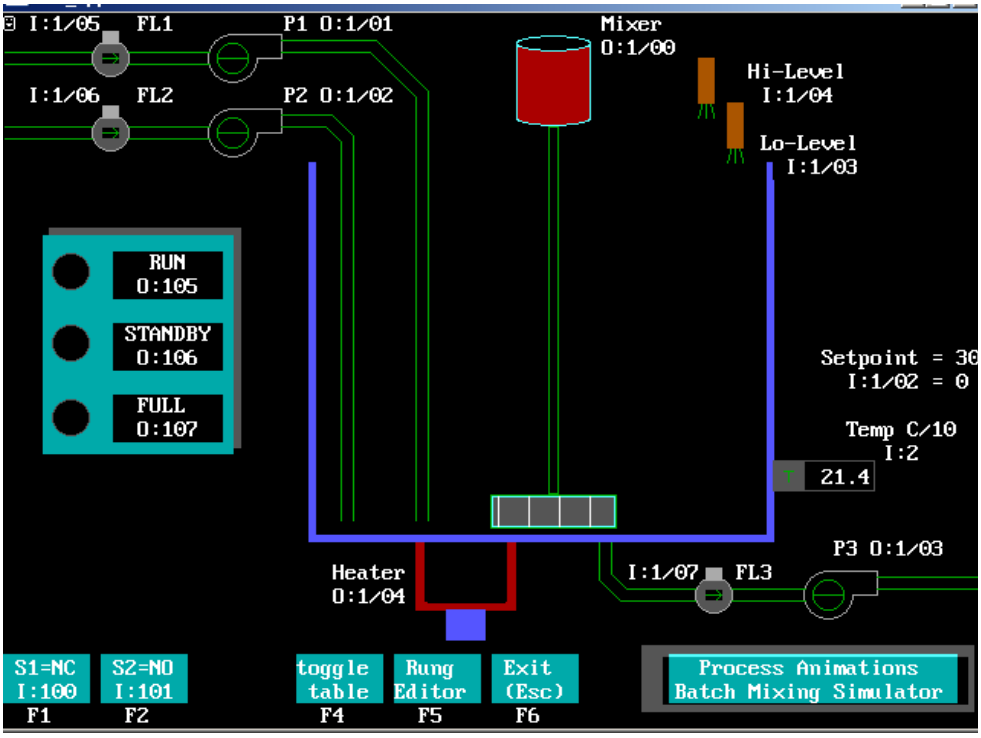

Рисунок 5

Зарисовать схему, продемонстрировать работу схемы преподавателю.

Задание 2. В соответствии с таблицей вариантов модернизировать программу.

№ варианта	Задание
1	• Установить следующие пропорции ингредиентов в миксере — из первой трубы 10%, из второй трубы 90%. Нагревать смесь до 23 градусов.
2	• Установить следующие пропорции ингредиентов в миксере — из первой трубы 20%, из второй трубы 80%. Нагревать смесь до 24 градусов.
3	• Установить следующие пропорции ингредиентов в миксере — из первой трубы 30%, из второй трубы 70%. Нагревать смесь до

№ варианта	Задание
	25 градусов.
4	• Установить следующие пропорции ингредиентов в миксере — из первой трубы 40%, из второй трубы 60%. Нагревать смесь до 26 градусов.
5	• Установить следующие пропорции ингредиентов в миксере — из первой трубы 50%, из второй трубы 50%. Нагревать смесь до 27 градусов.
6	• Установить следующие пропорции ингредиентов в миксере — из первой трубы 60%, из второй трубы 40%. Нагревать смесь до 28 градусов.
7	• Установить следующие пропорции ингредиентов в миксере — из первой трубы 70%, из второй трубы 30%. Нагревать смесь до 29 градусов.
8	• Установить следующие пропорции ингредиентов в миксере — из первой трубы 80%, из второй трубы 20%. Нагревать смесь до 30 градусов.
9	• Установить следующие пропорции ингредиентов в миксере — из первой трубы 90%, из второй трубы 10%. Нагревать смесь до 31 градусов.
10	• Установить следующие пропорции ингредиентов в миксере — из первой трубы 10%, из второй трубы 90%. Нагревать смесь до 32 градусов.

Зарисовать схему, продемонстрировать работу схемы преподавателю.

Содержание отчета.

1. Титульный лист с указанием номера и наименования работы, ф.и.о. студента, номера учебной группы.
2. Программа — схема на языке LD, в соответствии с заданием 1 и заданием 2.

Лабораторная работа №5 - Основы работы в среде CoDeSys на языке ST.

Цель работы: Знакомство с приемами программирования ПЛК на языке ST, использование среды разработки CoDeSys.

Аппаратное и программное обеспечение: PC, ОС Linux/Windows, CoDeSys.

Для выполнения данной лабораторной работы необходима среда разработки CoDeSys.

Источники информации:
http;//www.google.ru
http://www.owen.ru/catalog/codesys_v2/opisanie

Контрольные вопросы:

1. Что такое промышленный контроллер.
2. Что такое ПЛК.
3. Особенности работы промышленных контроллеров.
4. Особенности языка программирования ST.
5. Инициализация переменных на языке ST.
6. Условные операторы языка ST.
7. Операторы организации циклов на языке ST.

Содержание работы.

Задание 1. В CoDeSys составить программу управления светофором на языке ST. Создать объект визуализации работы программы (рисунок 6). Техническое задание для системы управления светофором:

- необходимо реализовать три режима работы светофора: ночной режим (с обоих сторон светофор мигает желтым цветом), автоматический режим, режим ручного управления,
- в автоматическом режиме последовательность переключения сигналов показана на рисунке 7,
- реализовать переключение между режимами работы.
- В ручном режиме переключение сигналов красный-зеленый выполняется оператором (последовательность переключения сигналов аналогична автоматическому)

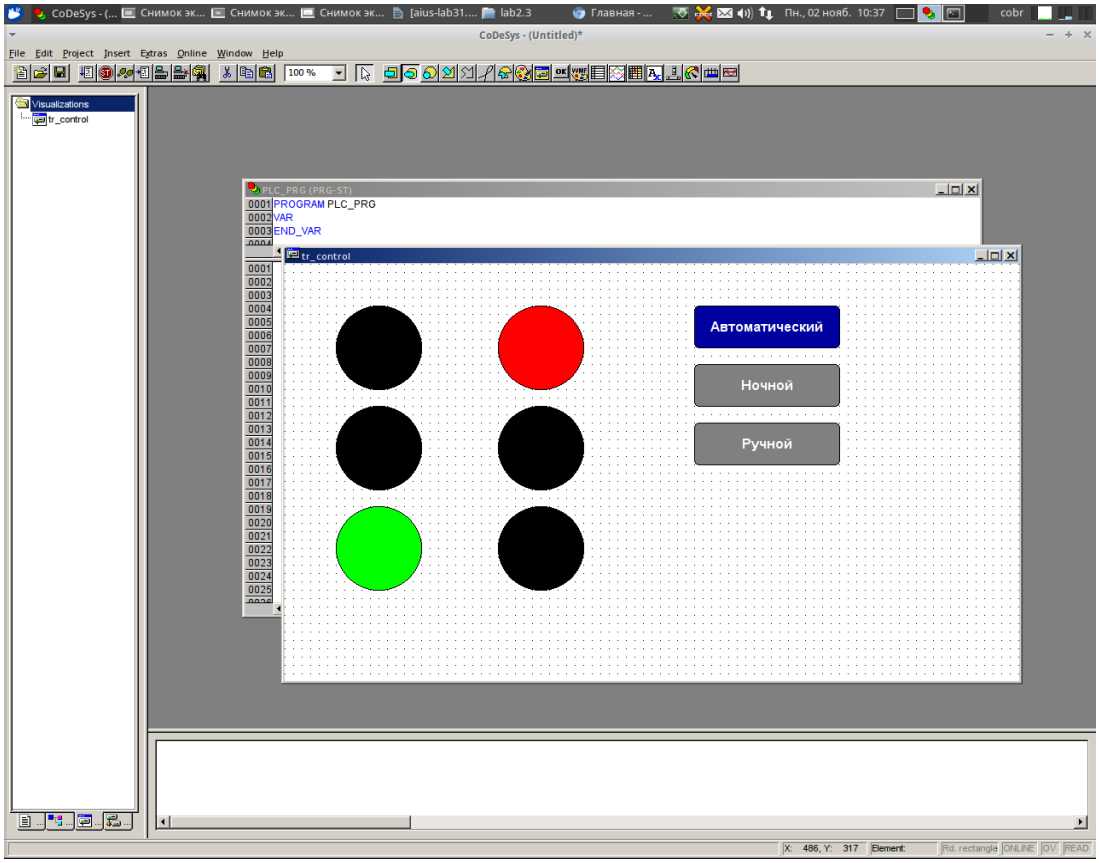

Рисунок 6

Зарисовать схему, продемонстрировать работу схемы преподавателю.

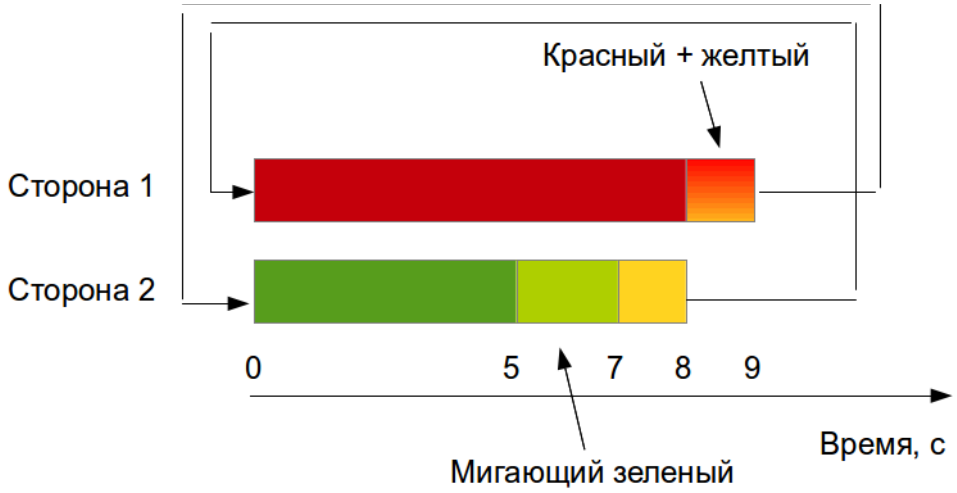

Рисунок 7

Содержание отчета.

1. Титульный лист с указанием номера и наименования работы, ф.и.о. студента, номера учебной группы.
2. Программа — схема на языке ST, в соответствии с заданием 1.

Лабораторная работа №6 - Составление программы управления лифтом на языке ST.

Цель работы: Знакомство с приемами программирования ПЛК на языке ST, использование среды разработки CoDeSys.

Аппаратное и программное обеспечение: PC, ОС Linux/Windows, CoDeSys.

Для выполнения данной лабораторной работы необходима среда разработки CoDeSys.

Источники информации:
http;//www.google.ru
http://www.owen.ru/catalog/codesys_v2/opisanie

Контрольные вопросы:
1. Что такое промышленный контроллер.
2. Что такое ПЛК.
3. Особенности работы промышленных контроллеров.
4. Особенности языка программирования ST.
5. Инициализация переменных на языке ST.
6. Условные операторы языка ST.
7. Операторы организации циклов на языке ST.

Содержание работы.

Задание 1. В CoDeSys составить программу управления лифтом на языке ST. Создать объект визуализации работы программы (рисунок 8). Упрощенное техническое задание для системы управления лифтом:

- количество этажей - 9,
- начальное положение лифта — 1ый этаж, время прохождения одного этажа — 1 секунда, задержка при открытии/закрытии дверей — 5 секунд,
- реализовать кнопку аварийной остановки лифта.
- При движении лифта вниз реализовать систему «сбора» пассажиров

Рисунок 8

Сохранить программу, продемонстрировать работу программы преподавателю.

Содержание отчета.

1. Титульный лист с указанием номера и наименования работы, ф.и.о. студента, номера учебной группы.
2. Программа — схема на языке ST, в соответствии с заданием 1.

Список литературы

Основная:

1. Минаев И.Г. Свободно программируемые устройства в автоматизированных системах управления / И.Г. Минаев, В.В. Самойленко, Д.Г. Ушкур, И.В. Федоренко - Ставрополь: АГРУС. 2016. - 168 с

2. Минаев И. Г. Программируемые логические контроллеры в автоматизированных системах управления / И. Г. Минаев, В. М. Шарапов, В. В. Самойленко, Д. Г. Ушкур. 2-е изд., перераб. и доп. - Ставрополь: АГРУС, 2010. - 128 с.

Дополнительная:

1. Минаев И. Г. Программируемые логические контроллеры. Практическое руководство для начинающего инженера. / И. Г. Минаев, В. В. Самойленко - Ставрополь: АГРУС, 2009. - 100 с.

Приложение А — Основные сведения о ПЛК.

Программируемый логический контроллер (ПЛК) - электронная составляющая промышленного контроллера, специализированного (компьютеризированного) устройства, используемого для автоматизации технологических процессов.

В качестве основного режима работы ПЛК выступает его длительное автономное использование, зачастую в неблагоприятных условиях окружающей среды, без серьёзного обслуживания и практически без вмешательства человека.

ПЛК являются устройствами реального времени, и имеют ряд особенностей, отличающих их от прочих электронных приборов, применяемых в промышленности:

- в отличие от микроконтроллера (одно-кристального компьютера) — микросхемы, предназначенной для управления электронными устройствами — областью применения ПЛК обычно являются автоматизированные процессы промышленного производства в контексте производственного предприятия;
- в отличие от компьютеров, ориентированных на принятие решений и управление оператором, ПЛК ориентированы на работу с машинами через развитый ввод сигналов датчиков и вывод сигналов на исполнительные механизмы;
- в отличие от встраиваемых систем ПЛК изготавливаются как самостоятельные изделия, отдельные от управляемого при его помощи оборудования.

Основными компонентами промышленного контроллера являются:

- источник питания (с защитой от помех и резервированием);
- центральный вычислительный модуль (собственно ПЛК);
- модули цифрового ввода-вывода;
- модули аналого-цифрового и цифро-аналогово преобразования (АЦП и ЦАП)

Дополнительными компонентами промышленного контроллера могут быть:

- устройство программирования (программатор);
- модули сетевых интерфейсов;
- устройства для ввода данных оператором.

Основные достоинства промышленного контроллера достигаются за счет возможности перепрограммирования ПЛК — универсальность и гибкость применения, возможность моделирования и визуальной отладки, возможность «визуального» программирования.

ПЛК работает под управлением специализированной операционной системы (ОС ПЛК), в задачи которой входит:

- загрузка и выполнение пользовательских программ;
- связь между устройствами (модули ввода/вывода, другие ПЛК);
- сбор и хранение сервисной информации (количество и тип модулей ввода вывода, информация о статусе модулей и системы).

Алгоритм работы ПЛК состоит из процедур инициализации, сканирования модулей ввода, выполнение программы пользователя и изменения состояний модулей вывода. Ввод, программа и вывод являются раздельными операциями. Изменение состояния входов устройства будет учтено только в следующую итерацию ввода. Изменение состояния выходов не происходит во время операции ввода и выполнения пользовательской программы.

Время выполнения пользовательской программы не должно превышать определенного значения (масштаб реального времени системы определяется по времени переходного процесса в системе). Типовое время выполнения программы — несколько миллисекунд.

Приложение Б — Язык программирования Ladder Diagram (LD).

Язык Ladder Diagram, или язык релейно-контактной логики, предназначен для программирования ПЛК, его реализации регулируются стандартом МЭК 61131-3. Основные отличия LD от других языков программирования ПЛК: ориентация на инженеров работающих с релейными схемами, наглядный интерфейс логики работы контроллера, представленный в виде электрических цепей.

Основными элементами языка являются контакты, которые можно образно уподобить паре контактов реле или кнопки. Пара контактов отождествляется с логической переменной, а состояние этой пары — со значением переменной.

Элементы языка. Различаются нормально замкнутые и нормально разомкнутые контактные элементы, которые можно сопоставить с нормально замкнутыми и нормально разомкнутыми кнопками в электрических цепях:

─┤ ├─ нормально разомкнутый контакт разомкнут при значении ложь (False), назначенной ему переменной и замыкается при значении истина (True);

─┤ / ├─ нормально замкнутый контакт, напротив, замкнут, если переменная имеет значение False, и разомкнут, если переменная имеет значение True;

─( )─ итог логической цепочки копируется в целевую переменную, которая называется катушка (англ. coil). Это слово имеет обобщенный образ исполнительного устройства, поэтому в русскоязычной документации обычно говорят о выходе цепочки, хотя можно встретить и частные значения термина, например катушка реле.

Далее приведены реализации основных логических операций на LD.

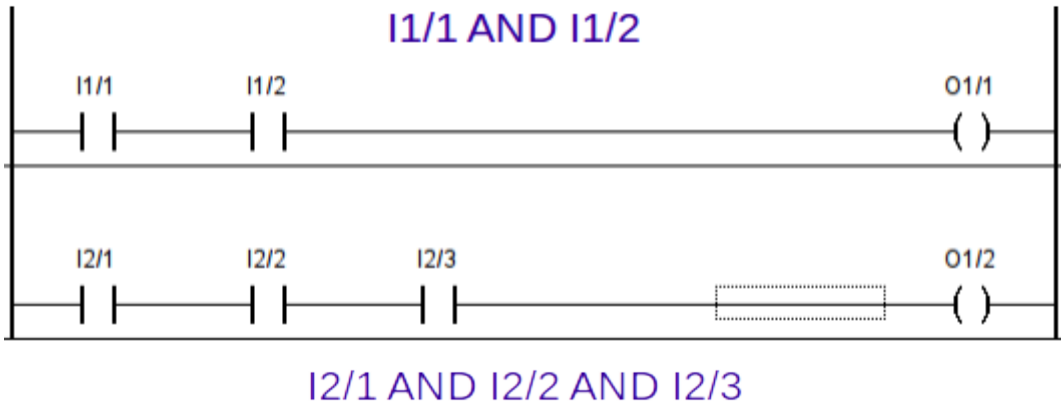

Рисунок 9: Логическое И

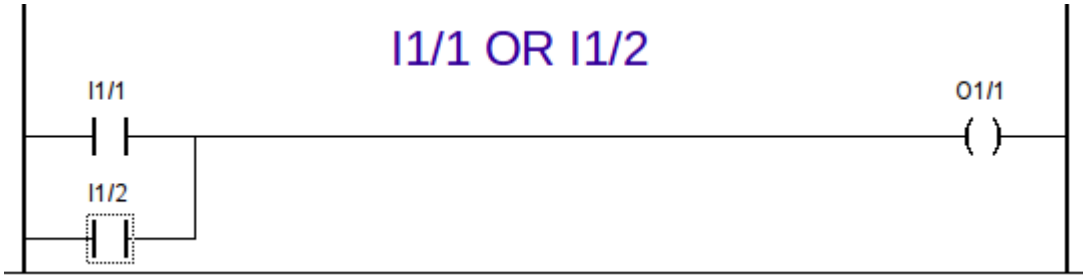

Рисунок 10: Логическое ИЛИ

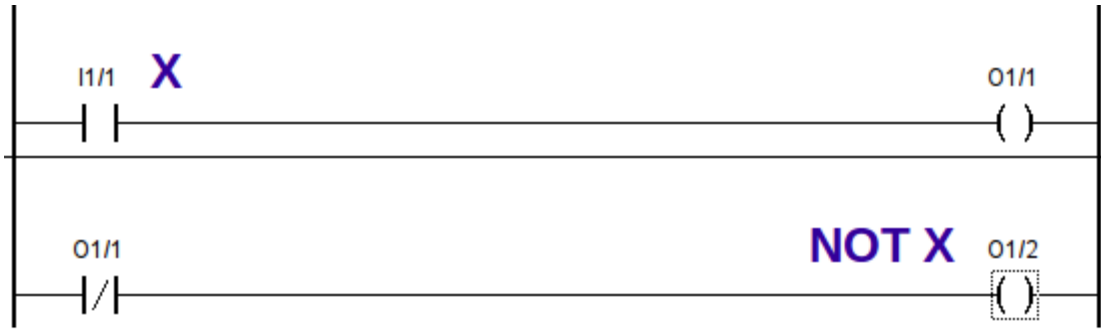

Рисунок 11: Логическое НЕ

Рисунок 12: Схема Старт-Стоп

Правила оформления программ на языке LD запрещают повторное использование одной и той же выходной инструкции, запрещено последовательное соединение выходных инструкций. Выходные инструкции могут соединяться параллельно.

Таймеры. Для отсчета временных интервалов используются таймеры. Простой таймер обычно имеет один вход (AC, для запуска) и три выходных инструкции — работы (EN), сигнала срабатывания (DN) и сигнала сброса (RES). ОС ПЛК хранит информацию о таймерах в

специальной таблице таймеров. Рассмотрим некоторые примеры работы с таймерами:

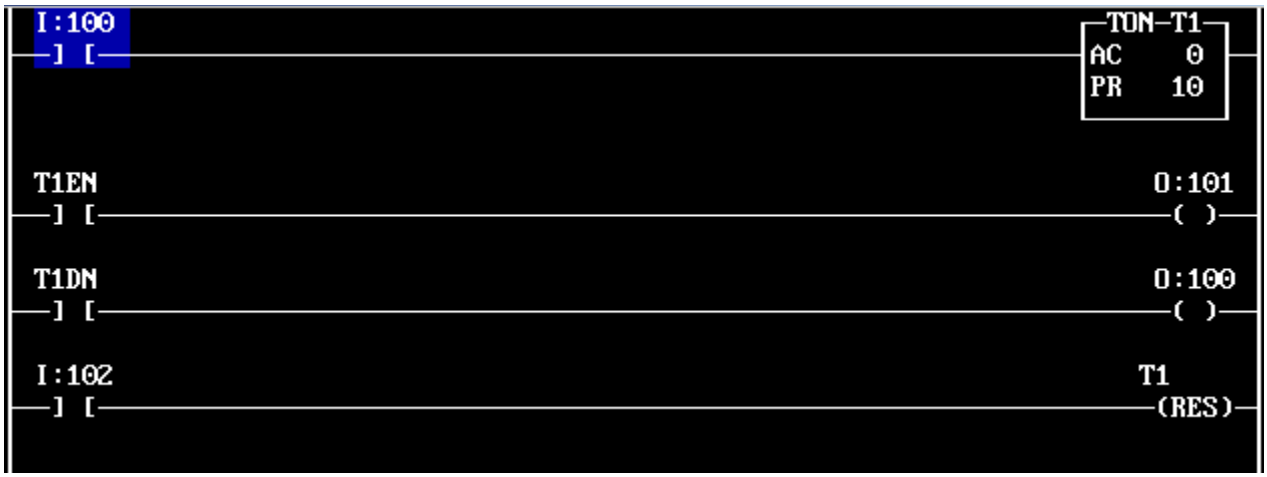

Рисунок 13: Таймер TON в PSIM

На рисунке 13 показан пример работы с TON — таймером с задержкой включения. При активации (замыкании) контакта I:100 таймер начинает отсчитывать время, одновременно активируется контакт T1EN. Когда таймер досчитает до величины Preset (PR 10)? активируется контакт T1DN. В данной реализации для перезапуска таймера необходимо активировать выходную инструкцию (RES). В данном случае таймер будет работать только при замкнутом контакте I:100.

Рассмотрим работу аналогичного функционального блока таймера в реализации CoDeSys.

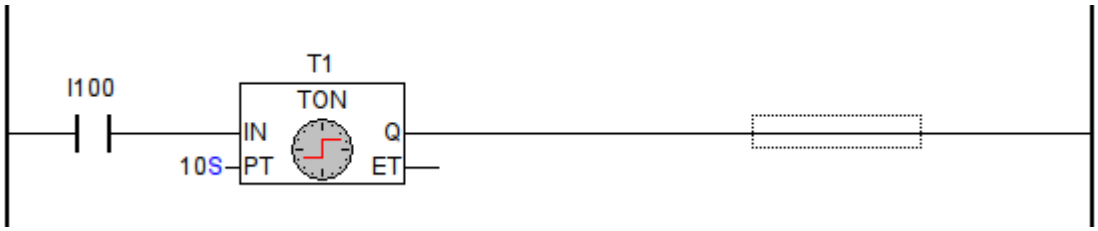

Рисунок 14: Таймер TON в CoDeSys

Входы IN и PT имеют типы BOOL и TIME соответственно. Выходы Q и ET аналогично типов BOOL и TIME. Пока IN равен FALSE, выход Q = FALSE, выход ET = 0. Как только IN становится TRUE, начинается отсчет времени (в миллисекундах) на выходе ET до значения, равного PT. Далее счетчик не увеличивается. Q равен TRUE, когда IN равен TRUE и ET равен PT, иначе FALSE. Таким образом, выход Q устанавливается с задержкой PT от фронта входа IN (см. рисунок 15).

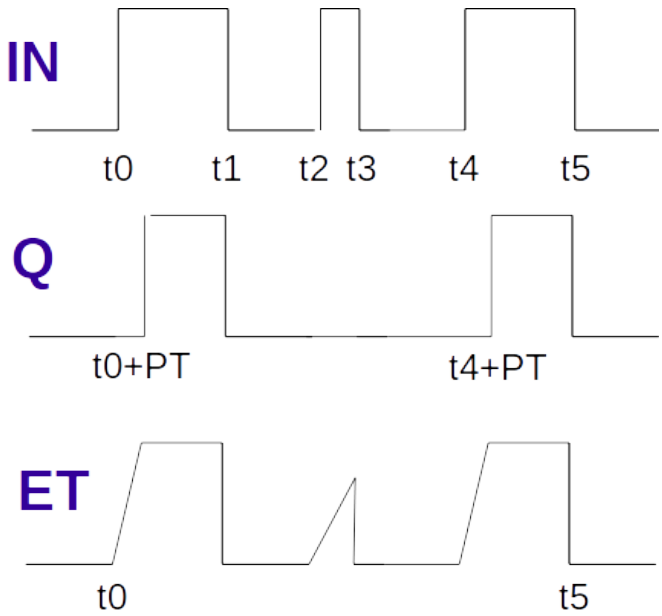

Рисунок 15: Временая диаграмма выводов таймера TON в CoDeSys

Приложение В — Эмулятор ПЛК PSIM.

Программа эмулятор ПЛК PSIM — это бесплатная система для обучения основам работы с ПЛК Allen Bradley. Основой программы служит редактор программ на языке LD, позволяющий пользователям создавать и редактировать программы с использованием инструкций Allen Bradley PLC-2. Следующий важный компонент — эмулятор работы ПЛК — обеспечивает «сканирование» виртуальных входов ПЛК и изменение состояния виртуальных выходов ПЛК, также как это происходит на реальном оборудовании. Третья часть программы — набор симуляторов для моделирования работы различных устройств под управлением ПЛК.

Установка программы. Для установки программы достаточно распаковать архив с программой в корневой каталог системного раздела накопителя (желательно использовать именно корневой каталог, т. к. программа работает через оболочку DosBox). Для начала работы достаточно запустить исполняемый файл из каталога программы. Так же в каталоге программы имеется папка «doc», в которой находится документация и упражнения (на английском языке).

Симуляторы. В PSIM реализовано 4 типа визуальных симуляторов, для моделирования работы устройств, под управлением ПЛК.

I/O Simulator — самый простой симулятор, позволяет получить первоначальные навыки разработки программ управления. Симулятор содержит 4 переключателя с фиксацией, 2 кнопки без фиксации с нормально разомкнутыми контактами, 2 кнопки без фиксации с нормально замкнутыми контактами, 8 контрольных лампочек. Так же показана таблица состояний регистров ПЛК, таблицы таймеров и счетчиков.

Silo Simulator — симулятор линии автоматического наполнения контейнеров. В состав симулятора входят: контрольная панель оператора с лампами и переключателями; конвейер, приводимый в движение электромотором, емкость с электромагнитной задвижкой, датчик положения контейнера, датчик превышения уровня наполнения контейнера. Так же показана таблица состояний регистров ПЛК, таблицы таймеров и счетчиков.

Traffic Light — симулятор системы управления светофором. В состав симулятора входят: двусторонний светофор, элементы управления (переключатели. Так же показана таблица состояний регистров ПЛК, таблицы таймеров и счетчиков.

Batch Mixer — симулятор автоматизированной системы смешивания компонентов. В состав симулятора входят: 2 помпы для наполнения

емкости, 2 расходомера; двигатель миксера; датчики высокого и низкого уровня; газовая горелка-нагреватель; датчик температуры; помпа для откачки с расходомером. Так же показана таблица состояний регистров ПЛК, таблицы таймеров и счетчиков.

Работа в редакторе LD. После запуска программы необходимо нажать Enter для отображения основного меню программы. В редактор LD можно попасть через любой из симуляторов (I/O Simulator, Silo Simulator, Traffic Light, Batch Mixer). Нажимаем, например, цифру 1 и попадаем в I/O Simulator, далее нажимаем F5 и попадаем в редактор программ на языке LD. Работая в редакторе, необходимо прежде всего, внимательно смотреть на нижнюю строку экрана — там располагаются все функции, доступные в текущий момент работы редактора.

Процесс создания программы на языке LD состоит в добавлении / удалении / редактировании цепей управления («ступенек лестницы», rungs). Самая простая цепь обычно состоит из как минимум одной входной инструкции и одной выходной инструкции. Для завершения работы с редактором достаточно нажать F10, при этом программа перейдет в режим симуляции и автоматически сохранит текущие наработки (до конца текущей сессии). При необходимости, в редакторе можно сохранить наработки в отдельный файл (через меню F7 — Program Utility).

В качестве примера рассмотрим процесс создания цепи управления, реализующей логическое ИЛИ (Рисунок 16). В редакторе LD добавляем «ступеньку» (F1), далее нажимаем F3 для реализации параллельной установки инструкций (Branch Start), вводим адрес первой инструкции (например I:1/01), снова нажимаем F3 для активации ветвления, вводим адрес второй инструкции (например I:1/02),, после чего закрываем параллельную ветвь клавишей F4 (Branch Close). Далее устанавливаем выходную инструкцию (клавиша F8, адрес O:1/00).

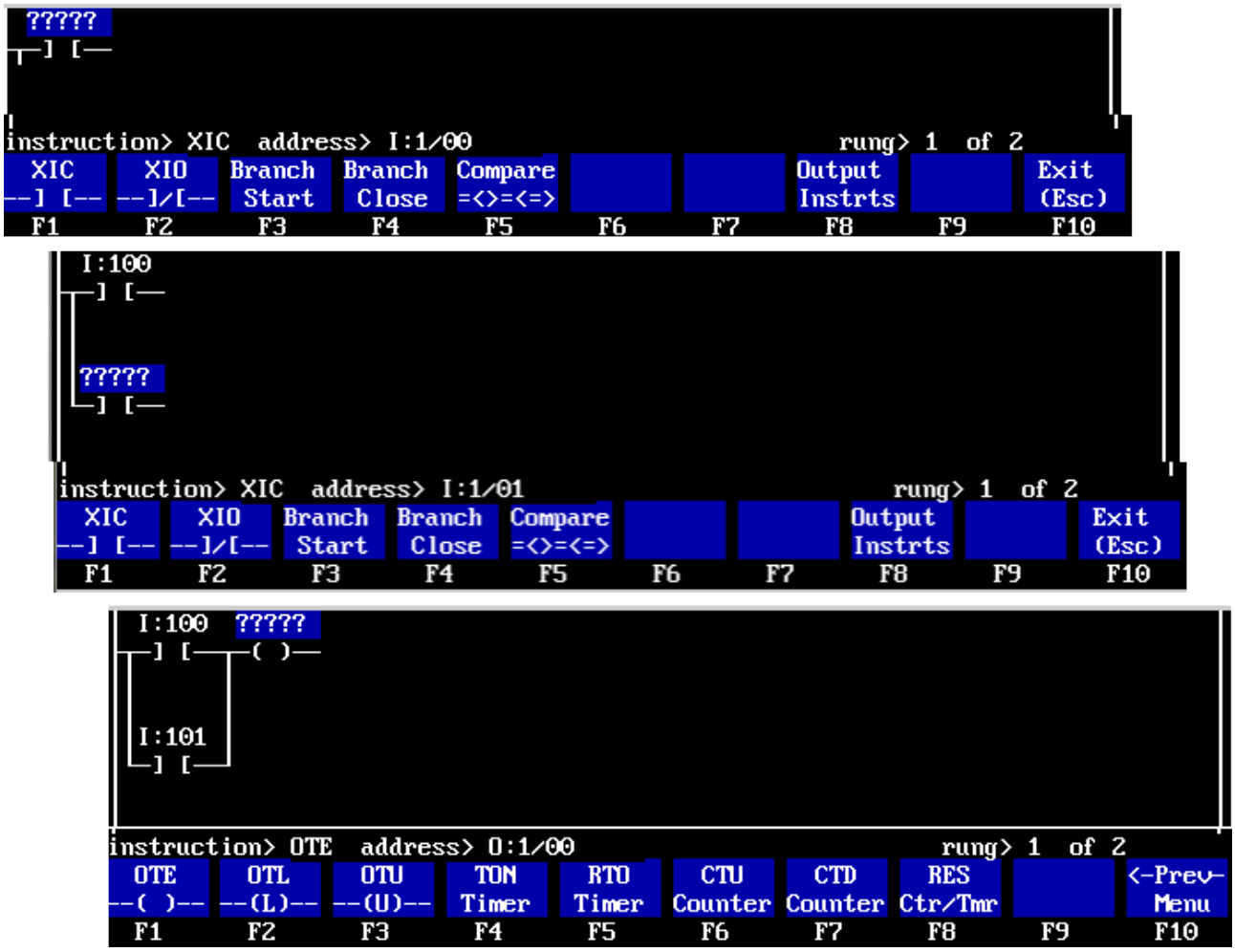

Рисунок 16

Далее рассмотрим работу с таймерами на примере таймера TON. Реализуем программу, которая активирует выходную инструкцию после 10 секундной задержки, также реализуем сброс таймера.

Приложение Г — Язык программирования Structured Text (ST).

ST это язык программирования стандарта IEC61131-3, по структуре и синтаксису ближе всего к языку программирования Паскаль. Удобен для написания больших программ и работы с аналоговыми сигналами и числами с плавающей точкой. ST представляет собой набор инструкций высокого уровня, которые могут использоваться в условных операторах ("IF...THEN...ELSE") и в циклах (WHILE...DO). Далее будем рассматривать реализацию ST в среде CoDeSys.

Выражения. Выражение – это конструкция, возвращающая определенное значение после его вычисления. Выражение состоит из операторов и операндов. Операндом может быть константа, переменная, функциональный блок или другое выражение. Вычисление выражений выполняется согласно правилам приоритета. Оператор с самым высоким приоритетом выполняется первым, оператор с более низким приоритетом – вторым и т.д., пока не будут выполнены все операторы. Операторы с одинаковым приоритетом выполняются слева направо.

Операция	Обозначение	Приоритет
Выражение в скобках	(Выражение)	Самый высокий.
Вызов функции	Имя функции (список параметров)	
Возведение в степень	EXPT	
Замена знаков	-	
Числовое дополнение	NOT	
Умножение	*	
Деление	/	
Абсолютная величина	MOD	
Сложение	+	
Вычитание	-	
Сравнение	< , > ,<=, >=	
Неравенство	< >	
Равенство	=	
Логическое И	AND	
Логическое исключающее	XOR	

ИЛИ		
Логическое ИЛИ	OR	Самый низкий

Перед оператором присваивания находится операнд (переменная или адрес), которому присваивается значение выражения, стоящего после оператора присваивания.
Пример: Var2: = Var3 * 10;
После выполнения этой операции Var2 принимает значение в десять раз большее, чем Var3.

Функциональный блок вызывается с помощью имени экземпляра функционального блока и списка входных параметров с присваиванием данных в круглых скобках. В следующем примере вызывается таймер с параметрами IN и PT. Значение выходной переменной Q присваивается переменной A. К выходной переменной можно обратиться с помощью имени экземпляра функционального блока, точки, следующей за ним и имени выходной переменной:
CMD_TMR (IN: = %IX5, PT: = 300);
A: =CMD_TMR.Q

Инструкция IF. Используя инструкцию IF, можно проверить условие, и в зависимости от этого условия выполнить какие-либо действия.
Синтаксис:
IF <Boolean_expression1> THEN
<IF_instructions>
{ELSIF <Boolean_expression2> THEN
<ELSIF_instructions1>
.
.
.ELSIF <Boolean_expression n> THEN
<ELSIF_instructions n-1>
ELSE
<ELSE_instructions>}
END_IF;
Часть конструкции фигурных скобках не обязательна. Если <Boolean_expression1> возвращает истину, тогда <IF_Instructions> выполняется. В противном случае будут выполняться остальные логические выражения одно за другим, пока одно из них не возвратит истину. Тогда выполняются инструкции, стоящие после этого логического выражения до следующего ELSIF или ELSE. Если все логические выражения ложны, то выполняются инструкции, стоящие после ELSE.

Инструкция CASE. С помощью инструкции CASE можно нескольким различным значениям целочисленной переменной сопоставить различные инструкции.
Синтаксис:
CASE <Var1> OF
<Value1>:
<Instruction 1>
<Value2>: <Instruction 2>
<Value3, Value4, Value5>:
<Instruction 3>
<Value6 .. Value10>:
<Instruction 4>
...
<Value n>:
<Instruction n>
ELSE
<ELSE instruction>
END_CASE;
Инструкция CASE выполняется согласно следующим правилам:

- если переменная <Var1> имеет значение <Value i>, то выполняется инструкция <Instruction i>;
- если <Var1> не принимает ни одного из указанных значений, то выполняется <ELSE Instruction>;
- чтобы одна и та же инструкция выполнялась при различных значениях переменной <Var1>, необходимо перечислить эти значения через запятую;
- чтобы одна и та же инструкция выполнялась для целого диапазона значений, необходимо указать начальное и конечное значения, разделенные двумя точками.

Цикл FOR. С помощью FOR можно программировать повторяющиеся процессы.
Синтаксис:
INT_Var :INT;
FOR <INT_Var> := <INIT_VALUE> TO <END_VALUE> {BY <Step size>} DO
<Instructions>
END_FOR
Часть конструкции, заключенная в фигурные скобки, не обязательна. <Instructions> выполняются, пока счетчик <INT_Var> не больше <END_VALUE>. Это условие проверяется перед выполнением

<Instructions>, поэтому раздел <Instructions> не выполняется, если <INIT_VALUE> больше <END_VALUE>. Всякий раз, когда выполняются <Instructions>, значение <INIT_VALUE> , увеличивается на <Step_size>. <Step_ size> может принимать любое целое значение. По умолчанию шаг устанавливается равным 1.

Цикл WHILE. Цикл WHILE может использоваться, как и цикл FOR, с тем лишь различием, что условие выхода определяется логическим выражением. Это означает, цикл выполняется, пока верно заданное условие.

Синтаксис:

WHILE <Boolean expression>
<Instructions>
END_WHILE

Раздел <Instructions> выполняется циклически до тех пор, пока <Boolean_expression> дает TRUE. Если <Boolean_expression> равно FALSE уже при первой итерации, то раздел <Instructions> не будет выполнен ни разу. Если <Boolean_expression> никогда не примет значение FALSE, то раздел <Instructions> будет выполняться бесконечно. Так же необходимо следить, чтобы цикл не станет бесконечным. Для этого в теле цикла значение входящей в условие переменной обязательно должно изменяться. Например, путем инкремента или декремента счетчика.

Цикл REPEAT. Цикл REPEAT отличается от цикла WHILE тем, что первая проверка условия выхода из цикла осуществляется, когда цикл уже выполнился 1 раз. Это означает, что независимо от условия выхода цикл выполняется хотя бы один раз.

Синтаксис:

REPEAT
<Instructions>
UNTIL <Boolean expression>
END_REPEAT

Раздел <Instructions> выполняется циклически до тех пор, пока <Boolean_expression> дает TRUE. Если <Boolean_expression> равно FALSE уже при первой итерации, то раздел <Instructions> не будет выполнен один раз. Если <Boolean_expression> никогда не примет значение FALSE, то раздел <Instructions> будет выполняться бесконечно, т. е. программист должен быть уверен, что цикл не станет бесконечным. Для этого в теле цикла значение входящей в условие переменной обязательно должно изменяться. Например, путем инкремента или декремента счетчика.

Инструкция EXIT. Если EXIT встречается в циклах FOR, WHILE, REPEAT, то цикл заканчивает свою работу независимо от значения условия выхода.