Министерство образования и науки Российской Федерации
Государственное бюджетное образовательное учреждение высшего профессионального образования
«Томский государственный университет систем управления и радиоэлектроники»

УТВЕРЖДАЮ

Зав.кафедрой ЭС

_____ Н.Е.Родионов
"____" _____ 2012 г.

Вводится в действие с"____" _____ 20 г.

МЕТОДИЧЕСКИЕ УКАЗАНИЯ ПО ПРОВЕДЕНИЮ ЛАБОРАТОРНЫХ РАБОТ

по дисциплине

Автоматизированные системы контроля и управления производственными процессами

Составлена кафедрой Электронных систем

Для студентов, обучающихся
по специальности 210302.65 «Радиотехника»

Форма обучения очная

Составитель доцент кафедры
Электронных систем, к.ф-м.н. Антипин М.Е.

"_10_" _июля_ 2012 г

Томск 2012 г.

Введение

При создании систем автоматизации технологических процессов значительную роль играет разработка средств сбора и визуализации данных о состоянии оборудования. Этот уровень реализуется с помощью программных SCADA-пакетов (Supervisory Control and Data Acquisition). Ряд лабораторных работ проводится для получения навыков настройки верхнего уровня автоматизации на базе программного пакета SCADA Infinity, разработанного компанией «ЭлеСи».

Ключевую роль в управлении технологическими процессами играют программируемые логические контроллеры (ПЛК, PLC). 2 лабораторные работы проводятся для получения навыков программирования ПЛК ЭЛСИ-ТМ (разработка Компании «ЭлеСи») в среде OpenPCS (разработка фирмы «InfoTeam») на языках стандарта IEC 61131-3. Знания в значительной степени могут быть использованы при программировании ПЛК других производителей, поддерживающих стандартные языки программирования.

Общие требования

Лабораторные работы выполняются студентами очной формы обучения индивидуально под контролем со стороны преподавателя. Все консультации осуществляются преподавателем. Число студентов, одновременно присутствующих на занятии не должно превышать 12 человек. Если в списочном составе группы студентов больше 12, то группа должна быть разделена на подгруппы численностью от 6 до 12 человек в каждой.

Для выполнения лабораторных работ целесообразно в учебном расписании выделять 4 академических часа подряд, без больших перерывов. Расписание также должно предусматривать раздельное проведение занятий у подгрупп, если группа была разделена.

Перед началом занятий студенты должны изучить инструкцию по охране труда. Преподаватель должен убедиться в знании инструкции, задавая студенту вопросы по ее содержанию, после чего сделать соответствующую запись в журнале охраны труда.

Во время проведения лабораторных занятий в аудитории (лаборатории) студентам запрещается передавать друг другу файлы и другие материалы, являющиеся результатом выполнения заданий.

Студент имеет право:

- Выходить из аудитории (лаборатории) не спрашивая разрешения у преподавателя.
- Самостоятельно распределять аудиторное время, определяя необходимость перерыва или непрерывной работы.

- Просить консультации у преподавателя, если он в текущий момент не распределяет задания, не принимает выполненные работы и не консультирует другого студента.

Преподаватель, давая консультацию студенту, указывает раздел технической документации или методической литературы, в которой имеется ответ на вопрос студента. Если необходимые сведения в документации и литературе отсутствуют, то преподаватель должен дать устные пояснения или продемонстрировать практические действия, приводящие к требуемому результату, с последующей отменой для повторения студентом.

Самостоятельная работа студентов над лабораторными заданиями осуществляется в той же аудитории (лаборатории), где проводятся лабораторные занятия. Преподаватель должен согласовать со студентами расписание самостоятельной работы - не менее 2 астрономических часов в неделю. В указанное время по учебному расписанию студентов и в аудитории (лаборатории) не должны проводиться другие занятия. Преподаватель должен обеспечить доступ студентов в аудиторию (лабораторию) в указанные часы. Необходимость самостоятельной работы определяет студент.

Консультации, выдача лабораторных заданий и прием результатов выполнения осуществляется только во время аудиторных занятий. Задания выполняются последовательно. Правильное выполнение некоторых заданий возможно только, если студент корректно выполнил предыдущие задания. Поэтому приступать к следующему заданию студент может, только сдав преподавателю результат выполнения предыдущего.

Техническое обеспечение практических работ

Для выполнения лабораторных работ студенту предоставляется индивидуальное рабочее место, в состав которого входят:

- стул;
- стол;
- персональный компьютер с операционной системой Windows XP;
- программное обеспечение SCADA Infinity;
- программное обеспечение OpenPCS;
- учебно-лабораторный стенд, имеющий в своем составе ПЛК ЭЛСИ-ТМ;
- модель объекта управления.

Размещение и освещенность рабочих мест в учебной аудитории (лаборатории) должно удовлетворять действующим требованиям СанПиН.

Прием результатов выполнения лабораторных работ

Результаты выполнения лабораторных работ демонстрируются преподавателю. Во время приема выполненной работы преподаватель вправе:

- Требовать у студента демонстрации выполнения функций сконфигурированного программного обеспечения, предусмотренных заданием.
- Самостоятельно производить манипуляции с программным обеспечением, не изменяя его конфигурацию.
- Требовать у студента пояснений, относящихся к способам реализации функций и назначению используемых управляющих элементов.
- Требовать у студента пояснений, относящихся к исходному коду и способам реализации программы.

Задание считается выполненным и принимается преподавателем только в том случае, если реализован весь функционал, предусмотренный заданием. Если какие то функции, предусмотренные заданием, не работают, или работают неверно, то результат выполнения подлежит доработке. Студент должен работать над кодом программы максимально самостоятельно, использовать отладочные средства, предоставляемые средой OpenPCS и программным пакетом SCADA Infinity.

Результаты выполнения заданий сохраняются преподавателем в электронном виде и хранятся в течение двух лет.

До конца семестра студент должен сдать результаты выполнения всех лабораторных работ, предусмотренным настоящими указаниями. В противном случае студенты к сдаче экзамена (зачета) не допускаются.

Задания для лабораторных работ

1. InfinityServer: Управляющий и конфигуратор сервера. Нормативное время выполнения– 2 часа:
   1.1. Управление сервером ввода-вывода Infinity Server. Создание, сохранение и загрузка конфигурации InfinityServer. Задание в приложении А.
2. InifinityHMI: Знакомство и основы. Нормативное время выполнения – 4 часа:
   2.1. Infinity HMI. Создание объекта «Прямоугольник» с динамикой цвета и размера в соответствии с изменениями заданного сигнала Infinity Server. Нормативное время выполнения – 2 часа. Задание в приложении Б.

2.2. Infinity HMI. Изучение встроенного динамического объекта «Кнопка», работа с битовыми сигналами Infinity Server. Нормативное время выполнения – 2 часа. Задание в приложении В.

3. InifinityHMI: Формулы и локальные переменные Нормативное время выполнения – 4 часа:

   3.1. Создание графической мнемосхемы учебного стенда «Булевы функции». Нормативное время выполнения – 2 часа. Задание в приложении Г.

   3.2. Создание графической мнемосхемы «RS-триггер с прямыми входами на элементах ИЛИ-НЕ». Нормативное время выполнения – 1 час. Задание в приложении Д.

   3.3. Создание графической мнемосхемы «Функциональный блок Т». Нормативное время выполнения – 1 час. Задание в приложении Е.

4. InifinityHMI: Работа с библиотекой, слоями и псевдонимами. Нормативное время выполнения – 6 часов:

   4.1. Создание графической мнемосхемы «Блок автоматики». Нормативное время выполнения – 2 часа. Задание в приложении Ж.

   4.2. Создание модели технологического процесса. Нормативное время выполнения – 2 часа. Задание в приложении И.

   4.3. Применение слоев. Нормативное время выполнения – 2 часа. Задание в приложении К.

5. Знакомство со средой программирования OpenPCS. Создание ресурса, задач, программ на языках стандарта IEC 6 1131-3 и их отладка в PLC-симуляторе OpenPCS 2004. Нормативное время выполнения задания – 6 часов. Задание в приложении М.

6. Создание программ на языках стандарта IEC 6 1131-3 и их отладка в ПЛК ЭЛСИ-ТМ. Нормативное время выполнения – 4 часа. Задание в приложении Н.

Библиографический список

1. Эксплуатационная документация на программный пакет SCADA Infinity. Является неотъемлемой частью программного обеспечения и устанавливается на рабочее место студента вместе с ПО.

2. Система программирования OpenPCS от InfoTeam. Версия 3.5. Руководство пользователя. – Поставляется с программным обеспечением OpenPCS.

3. Д.В.Кряжевских. Информационные системы управления технологическими и производственными процессами. – Томск: Томский межвузовский центр дистанционного образования, 2007. – 206 с.

Приложение А
Управление сервером ввода-вывода Infinity Server. Создание, сохранение и загрузка конфигурации InfinityServer.

Цель работы: Познакомиться с интерфейсом служебных приложений, используемых для управления состоянием и конфигурацией сервера ввода-вывода InfinityServer: «Управляющий» и «Конфигуратор».
Задание считается выполненным, если студент:

1. Продемонстрировал запуск и останов сервера ввода-вывода InfintyServer.
2. Произвел замену базы данных сервера на чистую (пустую).
3. Создал в конфигурации сервера ввода-вывода следующие сигналы:

Имя сигнала	Тип
BOOL/In1	Boolean
BOOL/In2	Boolean
BOOL/Out1	Boolean
BOOL/Out2	Boolean
INPUTS/Bit	Boolean
INPUTS/IntRamp	Float
INPUTS/IntSin	Float
INPUTS/IntRandom	Float
INPUTS/IntSquare	Float

4. Задал каждому созданному сигналу значение **0** (нуль) и качество **216**.
5. Добавил в состав сервера модули «OPC -сервер» и «Модуль вычислений».
6. Для сигнала INPUTS/IntRamp задал процедуру вычислений, ежесекундно увеличивающую значение сигнала на **100**, а по превышению **1000** сбрасывающую значение сигнала в **0** (нуль).
7. Сохранил изменения в конфигурации в файл (рекомендуется в качестве имени файла использовать свою фамилию в латинской транслитерации).
8. Загрузил конфигурацию из файла. Данную операцию необходимо делать перед началом каждого очередного задания после загрузки чистой (пустой) базы данных сервера (см.п.2).

Приложение Б
Infinity HMI. Создание объекта «Прямоугольник» с динамикой цвета и размера в соответствии с изменениями заданного сигнала Infinity Server.

Цель работы: Познакомиться с основными возможностями компоненты InfinityHMI, научиться разрабатывать схемы с использованием графических примитивов, и задавать анимацию графических объектов, управляемую сигналами InfinityServer

Задание считается выполненным, если студент:

1. Нарисовал в рабочей области InfinityHMI прямоугольник высотой не менее половины рабочей области с отношением ширина/высота примерно 1/3, как показано на рисунке:
2. Задал управление высотой прямоугольника от сигнала сервера ввода-вывода INPUTS/IntRamp. При этом нижняя грань прямоугольника должна быть неподвижна, ширина неизменна, а высота меняться 0% до 100% при изменении при изменении значения сигнала от **0** до **1000**.
3. Задал управление цветом заливки прямоугольника от сигнала INPUTS/IntRamp. Цвет должен изменяться от красного к синему при изменении значения сигнала от **0** до **1000**.
4. Создал справа от прямоугольника поле текстового отображения значения сигнала INPUTS/IntRamp.
5. Перевел мнемосхему в режим исполнения для демонстрации циклических изменений размера и цвета прямоугольника вследствие изменения значения сигнала INPUTS/IntRamp.
6. Сохранил файл мнемосхемы (рекомендуется в качестве имени файла использовать свою фамилию в латинской транслитерации с добавлением номера задания).

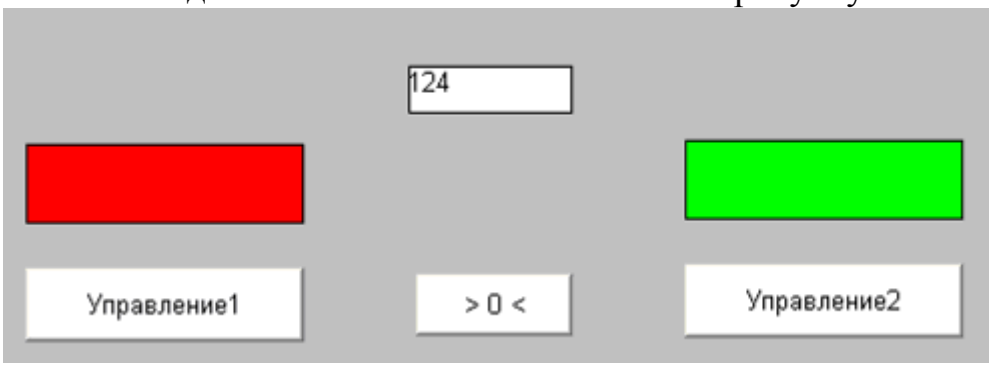
Приложение В
Infinity HMI. Изучение встроенного динамического объекта «Кнопка», работа с битовыми сигналами Infinity Server.

Цель работы: Научиться управлять сигналами InfinityServer при помощи графических элементов мнемосхем
Задание считается выполненным, если студент:

1. Создал в сервере сигналы.

Имя сигнала	Тип
Proba/Состояние1	Boolean
Proba/Состояние2	Boolean
Proba/Состояние3	Boolean
Proba/Состояние4	Boolean
Proba/Управление1	Boolean
Proba/Управление2	Boolean
Proba/summa	Word
Work/Reset	Boolean
Work/Level	Float
Work/Pump/Control	Boolean
Work/Pump/In	Boolean
Work/Pump/Status	Boolean
Work/Valve/Control	Boolean
Work/Valve/In	Boolean
Work/Valve/Status	Boolean

2. Установил каждому сигналу значение **0** и качество **216**.
3. Создал новую мнемосхему в InfinityHMI и разместил в рабочей области динамические элементы согласно рисунку:



4. Задать графическим элементам следующую динамику:

Графический элемент	Сигнал InfinityServer	Динамика
Кнопка «Управление1»	Proba/Управление1	Изменить булево значение при нажатии
Кнопка «Управление2»	Proba/Управление2	Изменить булево значение при нажатии
Кнопка «>0<»	Proba/summa	По нажатию присвоить значение **0** (нуль).

Прямоугольник левый	Proba/Управление1	Изменение цвета: **0** – красный, **1** - зеленый
Прямоугольник правый	Proba/Управление2	Изменение цвета: **0** – красный, **1** - зеленый
Поле отображения	Proba/summa	Отображение значения

5. Реализовал средствами InifinityHMI (не используя модуль вычислений InfinityServer) следующий алгоритм:

1 раз в секунду прибавлять **1** к значению сигнала Proba/summa, если сигнал Proba/Управление1 имеет значение **1** (true), отнимать **1** от значения сигнала Proba/summa, если сигнал Proba/Управление2 имеет значение **1** (true).

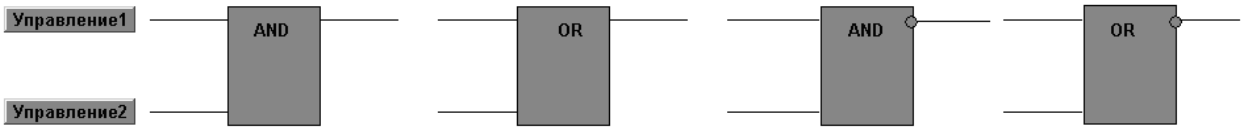
6. Сохранил мнемосхему и конфигурацию сервера ввода-вывода (рекомендуется в качестве имен файлов использовать свою фамилию в латинской транслитерации с добавлением номера задания).

Приложение Г

Создание графической мнемосхемы учебного стенда «Булевы функции».

Цель работы: Закрепить навыки работы с динамикой графических элементов.
Задание считается выполненным, если студент:

1. Создал мнемосхему с графическими элементами, представленными на рисунке:



Управление1	Управление2	Состояние1	Состояние2	Состояние3	Состояние4
?	?	?	?	?	?

2. Кнопки «Управление1» и «Управление2» должны переключать значение булевых сигналов Proba/Управление1 и Proba/Управление2 соответственно. Нажатие кнопок должно осуществляться с фиксацией. При нажатой кнопке значение сигнала должно быть **True**. При отжатой – **False**.
3. Входы логических элементов должны цветом отображать состояние сигналов: Proba/Управление1 – верхний вход и Proba/Управление2 – нижний вход.
4. Символы логических элементов должны записывать в сигналы сервера выполнение соответствующих логических операций над сигналами Proba/Управление1 и Proba/Управление2:

Имя сигнала	Операция
Proba/Состояние1	AND (логическое И)
Proba/Состояние2	OR (логическое ИЛИ)
Proba/Состояние3	!AND (И НЕ)
Proba/Состояние4	!OR (ИЛИ НЕ)

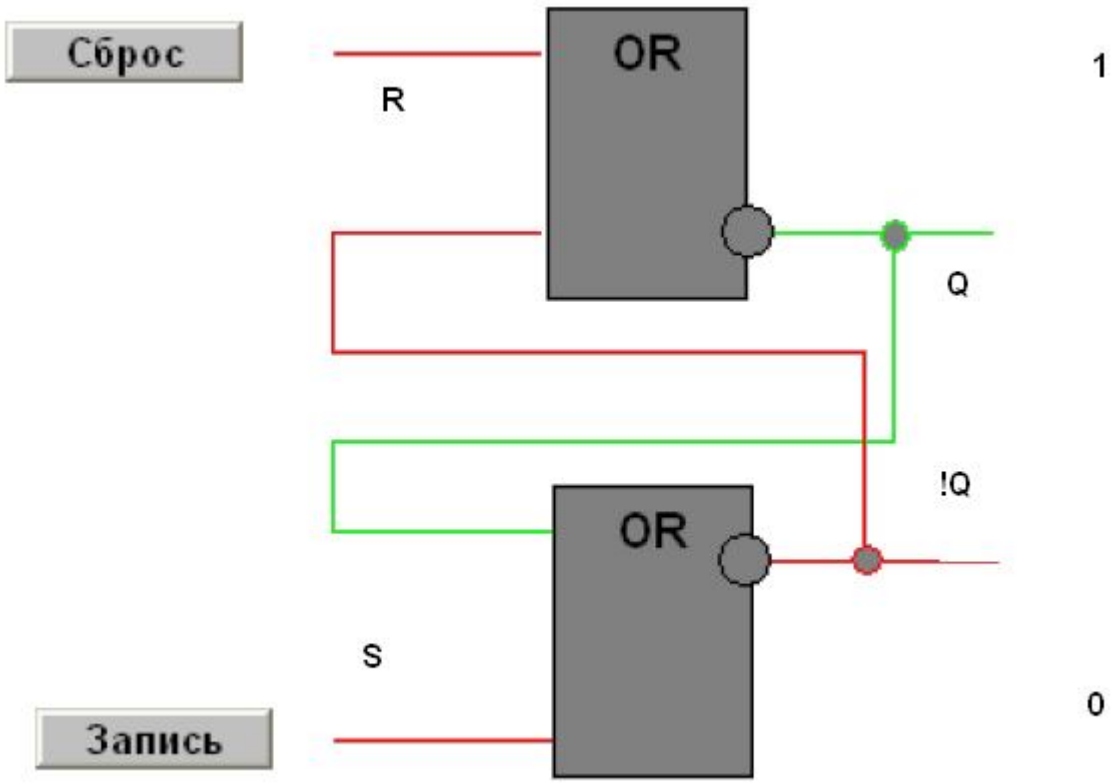
5. Выходы логических элементов должны отображать цветом значения соответствующих сигналов.
6. В полях таблицы должны отображаться значения сигналов, соответствующих заголовкам.

Приложение Д

Создание графической мнемосхемы «RS-триггер с прямыми входами на элементах ИЛИ-НЕ».

Цель работы: Закрепить навыки работы с динамикой графических элементов.

Задание считается выполненным, если студент с помощью сигналов сервера ввода-вывода, графических и динамических возможностей HMI создал в рабочей области мнемосхемы модель RS-триггера, представленную на рисунке



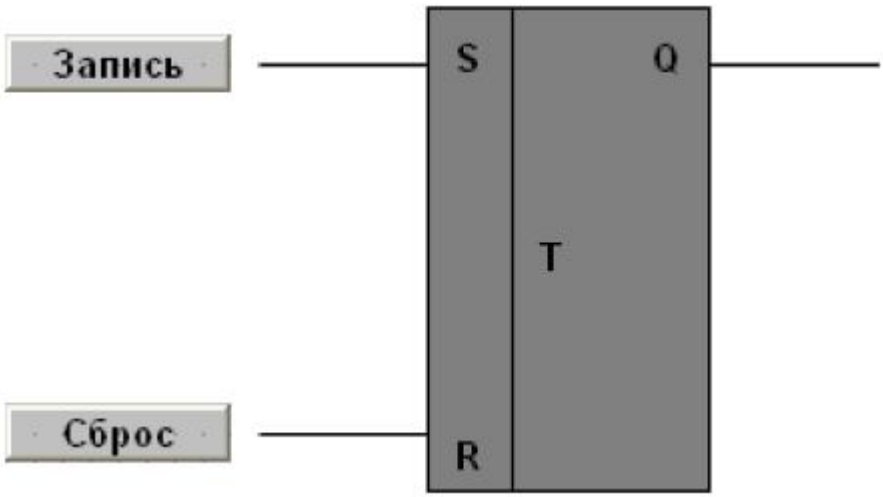
При выполнении задания рекомендуется использовать элементы, созданные при выполнении задания «Создание графической мнемосхемы учебного стенда «Булевы функции».

Приложение Е

Создание графической мнемосхемы «Функциональный блок Т».

Цель работы: Закрепить навыки работы с динамикой графических элементов.

Задание считается выполненным, если студент с помощью сигналов сервера ввода-вывода, графических и динамических возможностей HMI создал в рабочей области мнемосхемы модель RS-триггера с одним выходом в виде единого функционального блока, как показано на рисунке



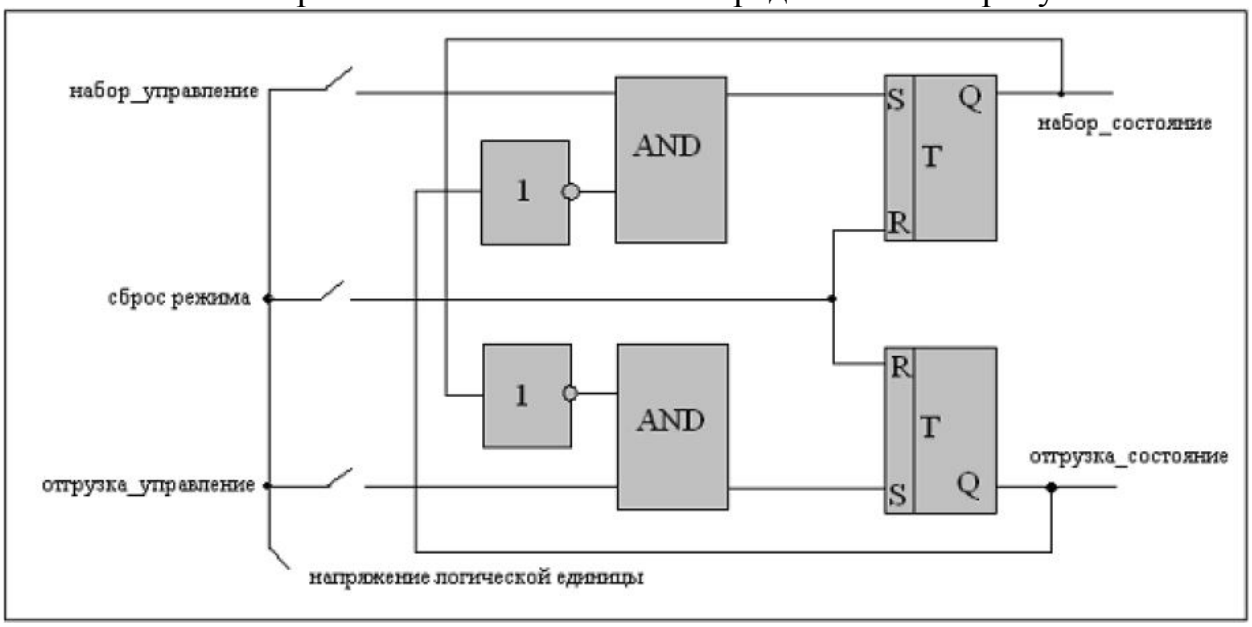
Приложение Ж

Создание графической мнемосхемы «Блок автоматики».

Цель работы: Закрепить навыки работы с динамикой графических элементов.

Задание считается выполненным, если студент с помощью сигналов сервера ввода-вывода, графических и динамических возможностей HMI создал в рабочей области мнемосхемы блок управления технологическим процессом, в котором участвуют два исполнительных механизма. Это задвижка, через которую в емкость поступает вода, и насос, который осуществляет отгрузку.

Логическая схема работы блока автоматики представлена на рисунке.



При выполнении рекомендуется использовать результаты выполнения заданий:

- Создание графической мнемосхемы учебного стенда «Булевы функции»
- Создание графической мнемосхемы «Функциональный блок Т»

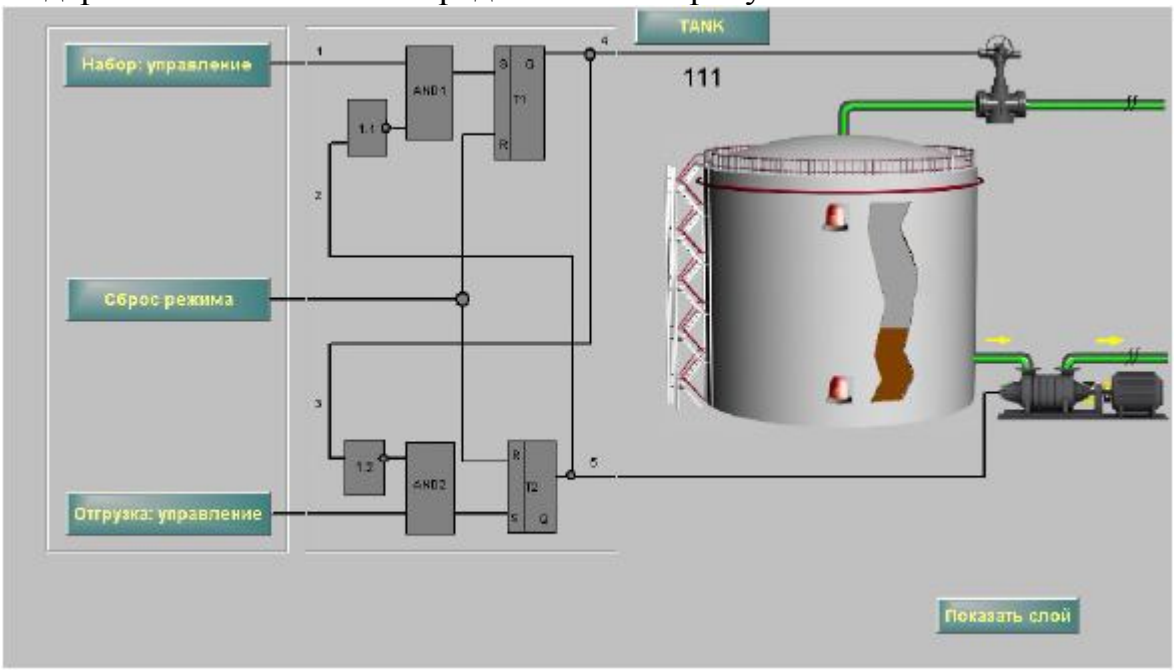
Приложение И

Создание модели технологического процесса.

Цель работы: Получить навыки работы с библиотекой графических динамических символов.

Задание считается выполненным, если студент создал мнемосхему управления резервуаром, используя предоставленную ему библиотеку символов Lybrary_Symbols.XML и результаты выполнения работы «Создание графической мнемосхемы «Блок автоматики».

Содержание мнемосхемы представлено на рисунке.



На мнемосхему следует добавлять объекты библиотеки со встроенной динамикой. Для управления динамикой графических объектов следует использовать окно редактирования псевдонимов, вызываемое контекстным меню объекта. Псевдонимам следует назначить имена OPC сигналов, соответствующих состояниям объектов.

Состояния графических динамических объектов Насос и Задвижка должны соответствовать состоянию соединенных с ними выходов блока управления. Уровень в резервуаре должен иметь значения от 0 до 1000. При отгрузке уровень должен уменьшаться на единицу каждые 50 мс. При наборе резервуара уровень должен увеличиваться на единицу каждые 50 мс. Символ уровня на динамическом объекте Резервуар должен отображать столбиком гистограммы текущее значение уровня пропорционально пределам изменения.

Приложении К

Применение слоев.

Цель работы: Получить навыки детализации графических мнемосхем при помощи подключаемых слоев.

Задание считается выполненным, если студент создал на мнемосхеме модели технологического процесса (Приложение И) дополнительные слои «Названия объектов» и «Значения параметров», а также одноименные кнопки для включения их отображения.

В слой «Названия» должны быть вынесены названия технологических объектов и элементов управления отображающихся на мнемосхеме.

В слой «Значения параметров» должны быть вынесены числовое отображение значения уровня в резервуаре и текстовое («Вкл»/«Выкл») отображение состояний объектов Клапан и Насос.

Приложение Л

Сведения о программирование контроллера на языках IEC-61131-3

Стандарт IEC 61131-3 описывает синтаксис и семантику пяти языков программирования ПЛК, - языков, ставших широко известными за более чем 30-летнюю историю их применения в области автоматизации промышленных объектов:

1. **SFC** (Sequential Function Chart) - графический язык, используемый для описания алгоритма в виде набора связанных пар: шаг (step) и переход (transition). Шаг представляет собой набор операций над переменными. Переход - набор условных логических выражений, определяющий передачу управления к следующей паре шаг-переход. По внешнему виду описание на языке SFC напоминает хорошо известные логические блок-схемы алгоритмов. SFC имеет возможность распараллеливания алгоритма. Однако SFC не имеет средств для описания шагов и переходов, которые могут быть выражены только средствами других языков стандарта. Происхождение: Grafcet (Telemechanique-Groupe Schneider).

2. **LD** (Ladder Diagram) - графический язык программирования, являющийся стандартизованным вариантом класса языков релейно-контактных схем. Логические выражения на этом языке описываются в виде реле, которые широко применялись в области автоматизации в 60-х годах. Ввиду своих ограниченных возможностей язык дополнен привнесенными средствами: таймера-ми, счетчиками и т.п. Происхождение: различные варианты языка релейно-контактных схем (Allen-Bradley, AEG Schneider Automation, GE-Fanuc, Siemens).

3. **FBD** (Functional Block Diagram) - графический язык по своей сути похожий на LD. Вместо реле в этом языке используются функциональные блоки, по внешнему виду - микросхемы. Алгоритм работы некоторого устройства на этом языке выглядит как функциональна схема электронного устройства: элементы типа "логическое И", "логическое ИЛИ" и т.п., соединенные линиями. Корни языка выяснить сложно, однако большинство специалистов сходятся во мнении, что это не что иное, как перенос идей языка релейно-контактных схем на другую элементную базу.

4. **ST** (Structured Text) - текстовый высокоуровневый язык общего назначения, по синтаксису ориентированный на Паскаль. Происхождение: Grafcet (Telemechanique-Groupe Schneider).

5. **IL** (Instruction List) - текстовый язык низкого уровня. Выглядит как типичный язык Ассемблера, что объясняется его происхождением: для некоторых моделей ПЛК фирмы Siemens является языком Ассемблера.

В рамках стандарта IEC 61131-3 к архитектуре конкретного процессора не привязан. Происхождение - STEP 5 (Siemens).

Перечисленные языки IEC 61131-3 используются ведущими фирмами изготовителями ПЛК, имеют длительную историю применения, достаточно распространены и известны пользователям по тем или иным модификациям. Несмотря на то, что во многих случаях такие модификации несущественны, это влечет определенные неудобства при работе с ПЛК различных фирм-изготовителей. Стандарт IEC 61131-3 позволяет программировать контроллеры всех производителей и модификаций на одинаковых языках и создавать средства программирования, не привязанные к конкретному ПЛК, такие как OpenPCS фирмы Infoteam или CoDeSys фирмы 3S.

Среда программирования OpenPCS состоит из двух частей: набора средств разработки и ядра-интерпретатора, исполняемого на целевом ПЛК. Набор средств разработки исполняется на персональном компьютере проектировщика, например, компьютере типа IBM PC, и состоит из редактора, отладчика и препроцессора, который переводит разработанный проектировщиком алгоритм к формату программы для ядра-интерпретатора. Этот набор имеет современный пользовательский интерфейс, позволяет тестировать алгоритм в режиме эмуляции и получать листинг алгоритма.

После создания пользовательская программа совместно с ядром-интерпретатором загружается в целевой ПЛК для исполнения. Ядро-интерпретатор, как следует уже из его названия, транслирует пользовательский алгоритм во время исполнения. Это позволяет сконцентрировать машинно-зависимый код и таким образом снизить накладные расходы при переходе на другой ПЛК. Неплохой подход, однако, необходимо отметить, что интерпретационная модель имеет недостаток – она всегда снижает показатели эффективности исполнения программы.

Программа, исполняющаяся в контроллере, получает информацию из внешней среды через входные переменные **X**. На основе полученных данных исполняющая система производит действия, заданные программой, и результат выводит через выходные переменные **Y**. Для промежуточных вычислений используются внутренние переменные **Z**. Контроллер с загруженной в него программой представлен на рисунке 1.

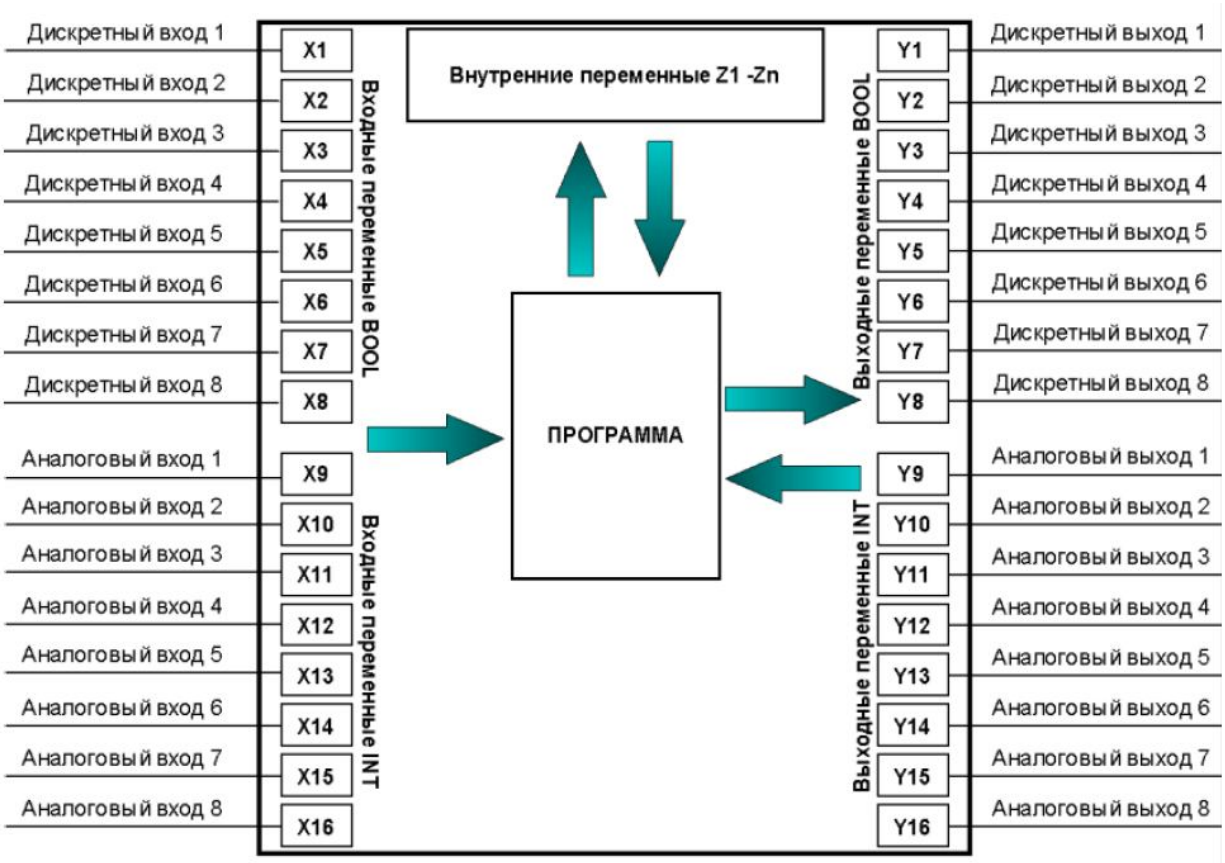


Рисунок 1. Обобщенное представление ПЛК

Приложение М
Знакомство со средой программирования OpenPCS. Создание ресурса, задач, программ на языках стандарта IEC 6 1131-3 и их отладка в PLC-симуляторе OpenPCS 2004

Цель работы: Познакомиться с системой программирования OpenPCS, научиться создавать ресурсы, разрабатывать программы на языках IEC 61131-3 и отлаживать их во встроенном симуляторе.

В данной работе студенту следует разработать программу управления объектом «Резервуар» на 4 различных языках программирования стандарта IEC-61131-3: ST, IL, LD и FBD. Объект резервуар имеет 2 управляемых элемента – насос для наполнения и клапан для опорожнения. Одновременное включение насоса и клапана не имеет смысла и не допускается. Все 4 программы должны иметь следующую логику работы: 3 входных сигнала (кнопки пользователя) Valve_In, Pump_In и Reset_In управляют состоянием выходных сигналов управления Valve_Control и Pump_Control. При подаче сигнала на вход Valve_In (нажатии кнопки) должен подаваться сигнал на выход Valve_Control, но только в том случае, если сигнал на выходе Pump_Control отсутствует. При подаче сигнала на вход Pump_In должен подаваться сигнал на выход Pump_Control, но только в том случае, если сигнал Valve_Control отсутствует. При подаче сигнала на выход Reset_In сигналы на выходах Pump_Control и Valve_Control должны выключаться (останов работы резервуара).

Запуск разработанных программ следует осуществлять на симуляторе ПЛК, встроенном в среду разработки OpenPCS. Для одновременной демонстрации работы всех четырех программ следует использовать для них разные входные и выходные адреса контроллера согласно таблице 1б

Таблица 1б. Физические адреса сигналов управления резервуаром в симуляторе ПЛК

Язык программиро-вания / Сигнал	ST	IL	LD	FBD
Valve_In	AT%I0.0	AT%I1.0	AT%I2.0	AT%I3.0
Pump_In	AT%I0.2	AT%I1.2	AT%I2.2	AT%I3.2
Reset_In	AT%I0.1	AT%I1.1	AT%I2.1	AT%I3.1
Valve_Control	AT%Q0.0	AT%Q1.0	AT%Q2.0	AT%Q3.0
Pump_Control	AT%Q0.2	AT%Q1.2	AT%Q2.2	AT%Q3.2

Задание считается выполненным, если студент разработал программу на всех 4 языках, загрузил ее в симулятор ПЛК и продемонстрировал преподавателю ее работу.

Приложение Н
Создание программ на языках стандарта IEC 6 1131-3 и их отладка в ПЛК ЭЛСИ-ТМ

Цель работы: Освоить отладку программ в ПЛК ЭЛСИ ТМ.

Установленная на рабочем месте студента программа OpenPCS позволяет программировать любой из 12 ПЛК ЭЛСИ-ТМ в лаборатории СУ ТП. Однако для визуального наблюдения состояния индикаторов ПЛК и блоков реле целесообразно подключаться и программировать только ПЛК стенда, входящего в состав данного рабочего места. Поэтому в начале выполнения данной работы студенту следует указать в параметрах соединения с ПЛК IP-адрес **192.168.0.1XX**, где **XX**-номер рабочего места. Студент, намеренно нарушающий эту рекомендацию, может быть удален из лаборатории по решению преподавателя. По техническим причинам преподаватель может указать студенту иной IP-адрес ПЛК, к которому следует подключаться.

Для облегчения работы с реальным ПЛК ЭЛСИ-ТМ следует импортировать в проект готовые файлы **Variables.POE** и **Transport.ST**, в которых уже установлена связь между переменными среды программирования и входными/выходными сигналами ПЛК.

В ходе этой работы студент должен создать и отладить два функциональных блока:

1. Start_Stop. Повторяет задачу управления резервуаром, описанную в приложении Б. В основной программе следует связать входные и выходные переменные с сигналами контроллера из файла **Variables.POE** согласно таблице 1в.

Таблица 1в

Вход/выход функционального блока	Внешняя переменная
Valve_In	D_In_3_1
Pump_In	D_In_3_3
Reset_In	D_In_3_2
Valve_Control	D_Out_1_1
	D_Out_2_1
Pump_Control	D_Out_1_2
	_Out_2_2

Проверка правильности выполнения задания осуществляется при помощи кнопок, расположенных в правой нижней части лабораторного стенда. Нажатие верхней кнопки должно включать наполнение одного из резервуаров модели. Нажатие нижней – откачку этого же резервуара. Средняя кнопка осуществляет сброс режима управления и останов работы резервуара.

2. Функциональный блок DC, осуществляющий перевод числа из десятичной системы счисления в двоичную. Для ввода десятичных чисел

будут использоваться кнопки стенда. Верхняя кнопка управляет единицами, средняя- десятками, нижняя – сотнями. Нажатие на кнопку означает прибавление единицы к заданной ранее цифре в соответствующем разряде десятичного числа. Для вывода двоичного числа будет использоваться четвертый блок реле. Соответственно у функционального блока DC должно быть 3 входных и 8 выходных сигналов. В основной программе их следует связать с внешними переменными файла **Variables.POE,** в соответствии с таблицей 2в

Таблица 2в

Сигналы DC	Внешние переменные	Тип
Un	D_In_3_1	вход
Dec	D_In_3_2	вход
Hund	D_In_3_3	вход
zerob	D_Out_4_1	выход
frstb	D_Out_4_2	выход
scndb	D_Out_4_3	выход
thrdb	D_Out_4_4	выход
fourb	D_Out_4_5	выход
fiveb	D_Out_4_6	выход
sixb	D_Out_4_7	выход
sevnb	D_Out_4_8	выход

Приложение Г
Создание программы для управления моделью объекта «Резервуарный парк» с помощью тумблеров

Цель работы: Создать программу для управления режимом работы модели объекта управления «Резервуарный парк»

В ходе этой работы студентам следует создать программу на языке FBD, которая будет реализовывать следующий алгоритм работы:

a) Тумблеры служат в роли включения определенного режима работы макета
b) Макет должен выполнять следующие режимы:

1 – стоп и сброс всех переменных отвечающих за работу клапанов и насосов

2 – набор в резервуар 1. Включение насоса происходит с задержкой 0.5с по отношению к клапану и наполняется до уровня 500.

3 – откачка из резервуара 1. Выключение клапана с задержкой 0.5с по отношению к насосу.

4 – набор в резервуар 2. Включение насоса происходит с задержкой 0.5с по отношению к клапану и наполняется до уровня 500

5 – откачка из резервуара 2, выключение клапана с задержкой 0.5с по отношению к насосу

6 – перекачка из 1 емкости во 2, насосы включаются с задержкой 0.5с, набор осуществляется до уровня 500.

7 – перекачка из 2 емкости во 1, насосы включаются с задержкой 0.5с, набор осуществляется до уровня 500.

Внешние переменные, используемые в программе:

Таблица 1г

Название сигнала	Тип переменной	Назначение
D_In_3_1	Bool	Сигнал верхней кнопки
D_In_3_2	Bool	Сигнал средней кнопки
D_In_3_3	Bool	Сигнал нижней кнопки
D_Out_1_1	Bool	Насос наполнения резервуара 1
D_Out_1_2	Bool	Насос откачки резервуара 1
D_Out_1_3	Bool	Насос наполнения резервуара 2
D_Out_1_4	Bool	Насос откачки резервуара 2
D_Out_2_1	Bool	Клапан наполнения резервуара1
D_Out_2_2	Bool	Клапан откачки резервуара 1
D_Out_2_3	Bool	Клапан наполнения резервуара 2
D_Out_2_4	Bool	Клапан откачки резервуара 2
Out_Uint	Uint	Номер режима работы

Таблица 1г

Название сигнала	Тип переменной	Назначение
Level3	Uint	Уровень в общей емкости
Analog_1	Uint	Уровень в резервуаре 1
Analog_2	Uint	Уровень в резервуаре 2

Максимальный уровень Level3 следует приравнять 2000 и вычислять его исходя из показаний датчиков уровня первой и второй емкости.

Необходимо создать функциональный блок на языке ST, который будет запоминать включенный режим. Данный блок необходим для запоминания текущего режима работы макета, т.е. при включении режима будет невозможно включить другой, не сбросив предыдущий.

Работа считается выполненной, если разработанная программа загружена в ПЛК и ее правильная работа проверена при помощи тумблеров.