Цель работы:

1. Закрепление навыков работы с интерфейсом Infinity HMI.
2. Работа с библиотекой символов.
3. Применение инструмента **Слои**.
4. Применение функций и условий для оптимизации работы мнемосхемы.
5. Создание новых объектов.

Упражнение 1: Создание модели технологического процесса.

1. Загрузите в Конфигуратор папку Teach из личной директории.
2. Запустите **Infinity HMI**.
3. Откройте свой файл **Фамилия5.xml** из личной директории.
4. Сохраните его под именем **Фамилия6.xml**
5. Проверьте работоспособность мнемосхемы.
6. Измените размеры символа (см Рис. 1) для того, чтобы на рабочем поле экрана справа от символа создать новые объекты.

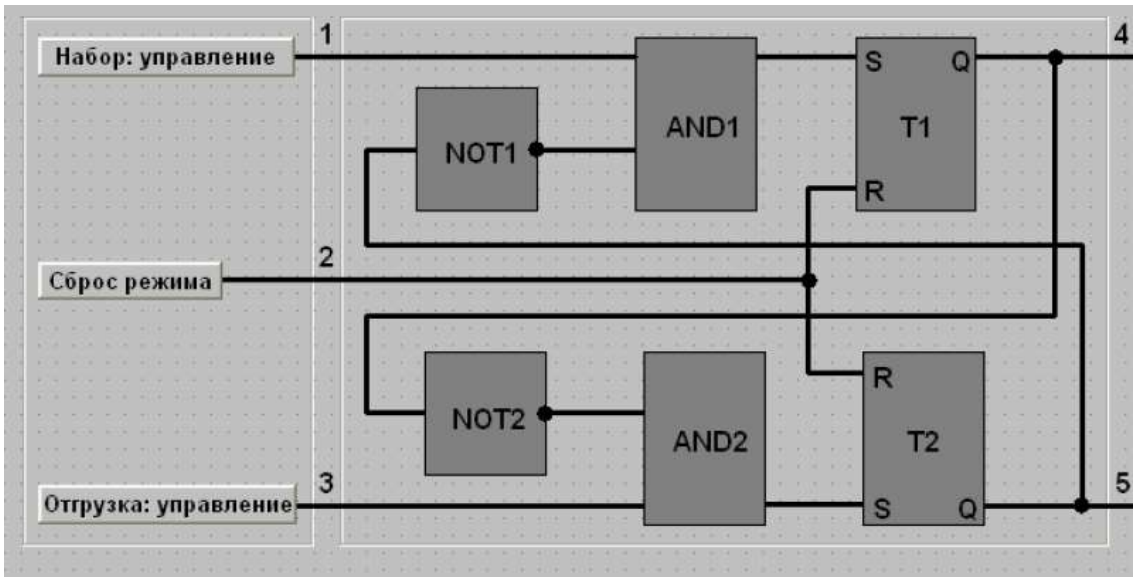

Рис. 1

7. В панели инструментов **Рисование** найдите кнопку **Библиотека символов** (или в меню выбрать пункт **Вид**, затем **Библиотека символов**) и в появившемся окне выберите **файл>открыть**. Укажите путь к библиотеке C:\Teach\Методические Указания\Library_Simbols.xml Рис. 2

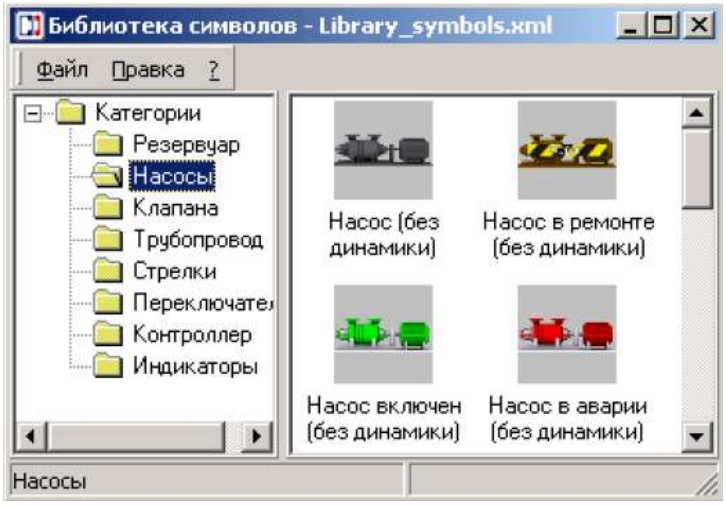

Рис. 2

8.

Библиотека символов является отдельным независимым от Infinity HMI приложением. **Библиотека символов** предназначена для хранения графических символов с целью дальнейшего их использования. Интерфейс пользователя **Библиотеки** аналогичен применяемому в Проводнике Windows. Главное окно **Библиотеки символов** разделено на две области. Левая область содержит древовидный список, в котором находятся каталоги и категории символов. В правой части окна расположены изображения символов текущей выбранной категории. Любой графический объект, находящийся в экранной форме, может быть сохранён в текущей выбранной категории путём выполнения простой операции переноса. Загрузку символов из библиотеки в экранные формы выполняется аналогично, путём переноса.

9. Перенесите на свободное место экранной формы объекты, необходимые для визуализации процесса, показанного на Рис. 3. **Переносите объекты с пометкой(с динамикой).** Закройте библиотеку символов.

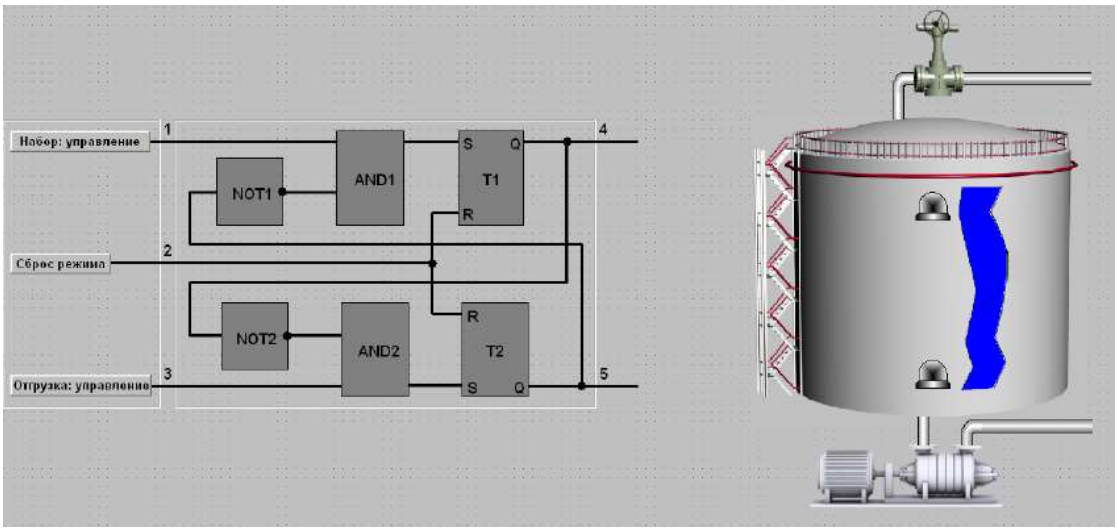

Рис. 3

10. Продолжите линии для соединения выхода **4** с задвижкой, а выхода **5** с насосом. Обязательное условие: **не применять разгруппировку символа и сохранять динамику, примененную к линиям.** Не имеет значения, будут эти линии прямыми или ломанными, важно показать связь.

11. Запустите **Конфигуратор** и убедитесь что в папке **Pump** и **Valve** есть сигналы с именем **Status** и параметрами, как у сигналов **Control** и **In** на Рис 5.

Сигнал **Control** (управление) – это дискретный сигнал относится к классу сигналов телеуправления (ТУ). Это команда контроллеру включить (выключить) исполнительный механизм (задвижку или насос). В свою очередь исполнительный механизм должен подтвердить контроллеру (а контроллер в OPC-сервер и дальше в мнемосхему диспетчеру) изменение своего состояния выдачей дискретного сигнала, который относится к классу сигналов телесигнализации (ТС). В этой лабораторной работе будет использована программная имитация этого подтверждения путем перекладки в динамике сигнала управления ТУ (**Control**) в сигнал состояния ТС (**Status**) задвижки и насоса. Между приходом ТУ на механизм и выдачей ТС есть задержка, обусловленная особенностями самого механизма (например, срабатывание конечного выключателя задвижки из положения "закрыто" в положение "открыто" после получения сигнала на открытие). Эта задержка будет реализована так же программно.

12. Примените к изображению задвижки динамику **Динамическое действие** и настройте её таким образом, чтобы один раз в **2500**мс в сигнал **Status** директории **Valve** записывалось значение сигнала **Control.** Это и будет программная перекладка сигнала ТУ в сигнал ТС с необходимой задержкой на **2,5** секунды.

13. Примените к изображению задвижки динамику для визуализации значения её сигнала ТС. Для этого кликните правой кнопкой мыши на изображении задвижки и выберите пункт **Редактировать псевдонимы**. В появившемся окне (Рис. 4) замените в каждой строке имя переменной **<<Valve>>** на путь к сигналу **Status** директории **Valve.**

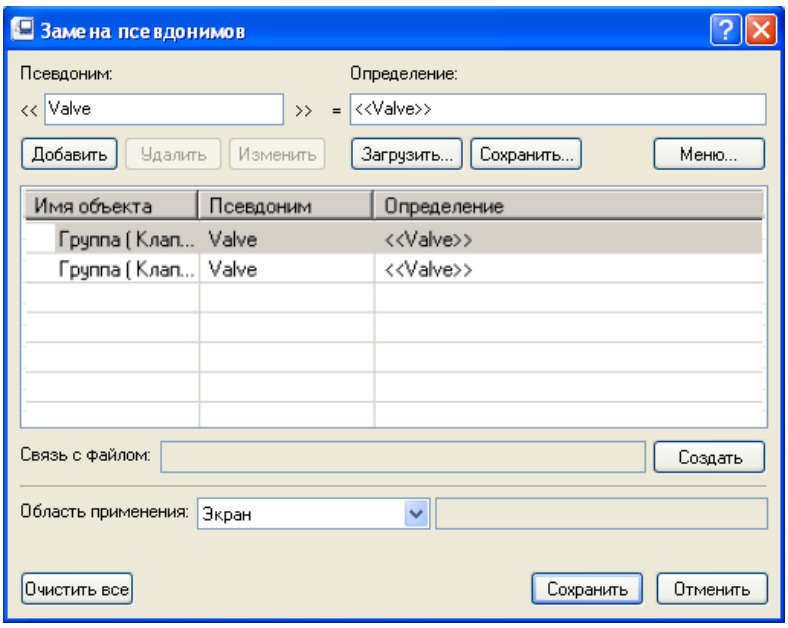

Рис. 4

14.Выделите изображение насоса и примените к нему соответствующую динамику для перекладки его сигналов ТУ в ТС с такой же задержкой.

15.Аналогичным образом настройте визуализацию значения сигнала ТС для насоса (Пункт 13).

16.Примените настройки, сохраните файл и проверьте работу мнемосхемы. Через время, определенное задержкой в **2,5** секунды, после записи в сигнал ТУ задвижки единицы, её цвет должен измениться на зелёный, это будет говорить о единичном уровне ТС. Та же картина будет наблюдаться и с насосом. Выключение исполнительных механизмов будет происходить с такой же задержкой относительно сигналов ТУ.

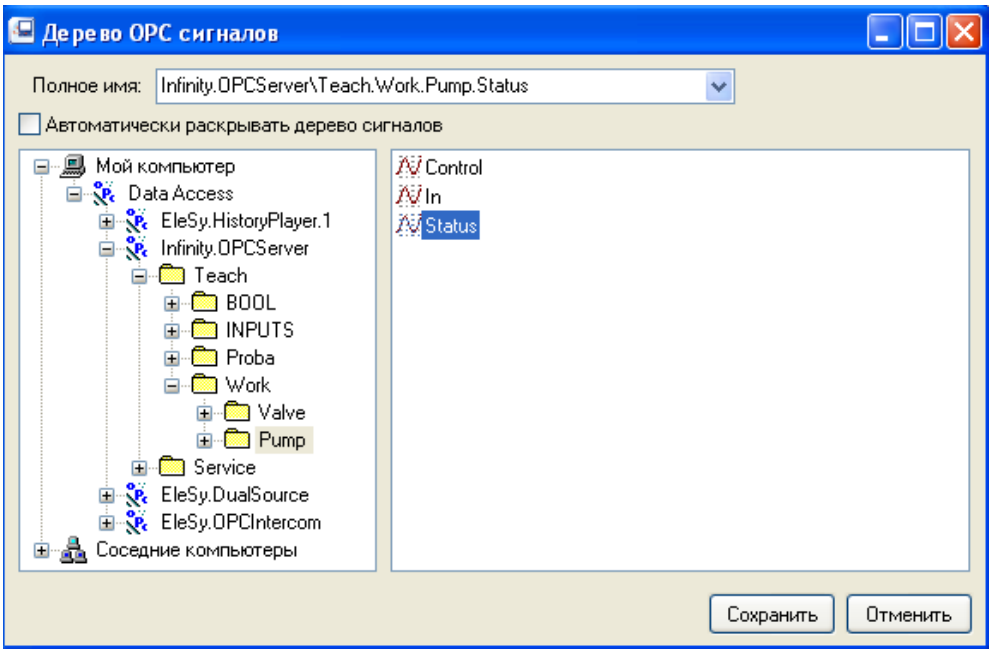

Рис. 5

17.Следующим шагом станет создание изменяющегося уровня в резервуаре.

18.Настройте псевдонимы в резервуаре, для визуализации уровня жидкости. (Пункт 13)

19.Примените к изображению резервуара динамику для реализации следующей логики: уровень в резервуаре равен поступлению через задвижку (ТС) минус отгрузка через насос (ТС). Время записи вычисленного значения выражения в сигнал **Level** равно **50**мс.

20.Выберите объект **Значение параметра** для визуализации значения сигнала **Level.** Экранная форма в режиме **Режим-Исполнение** выглядит, как на Рис6.

21.Сохраните файл.

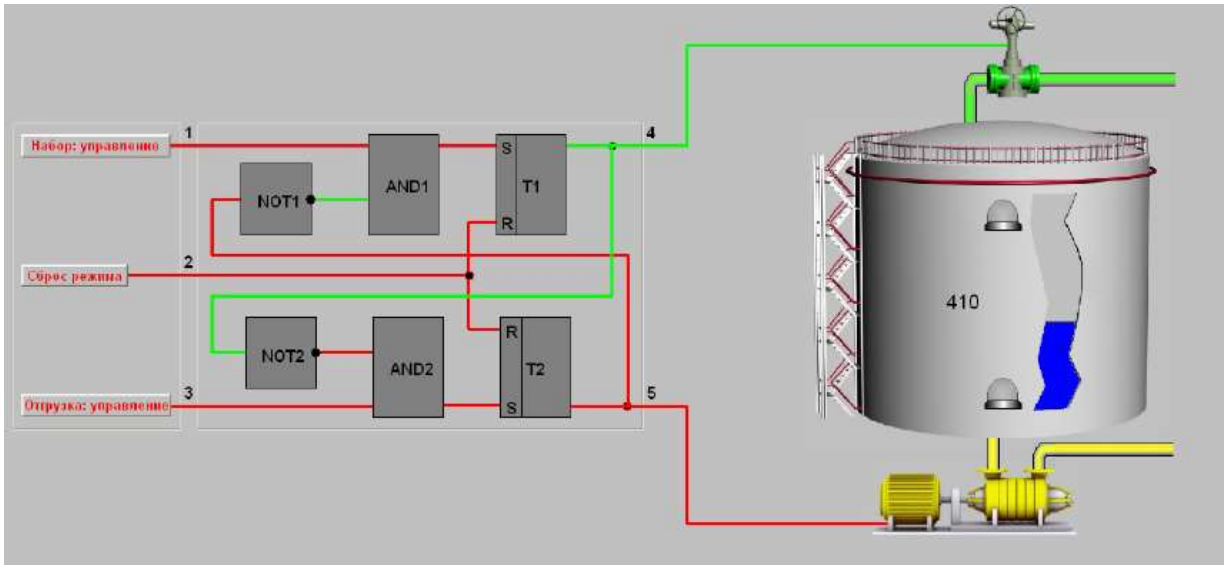

Рис. 6

Упражнение 2: Применение слоёв.

Слои в экранных формах **Infinity HMI** являются удобным средством для объединения наборов графических объектов, когда объекты, входящие в данный набор должны отображаться при в определенных условиях. В отсутствии этих условий объекты должны быть скрыты. Это очень разумно с точки зрения читаемости мнемосхемы в основное время, когда эти определенные условия не наступили. Экранная форма, таким образом, не перегружена лишними объектами, которые в обычное время мешают наблюдать за технологическим процессом. Условия могут быть разными: сигналы аварии, любые другие сигналы, изменение масштаба экранной формы. Каждая экранная форма изначально содержит один слой, называемый **Первичным** или **Системным.** Мнемосхема, содержащаяся в файле **Фамилия.xml** находится в этом **Первичном** слое и содержит объекты, которые оператор, предположим должен видеть всегда. По умолчанию редактор слоёв предлагает Имя слоя: **Layer2**

1. Сохраните файл под именем **Фамилия8.xml**

2. Нажмите на кнопку и создайте новый слой. Название по умолчанию **Layer2.**

3. Добавьте кнопку **Показать слой** и настройте **Свойства объекта,** как показано на Рис7.

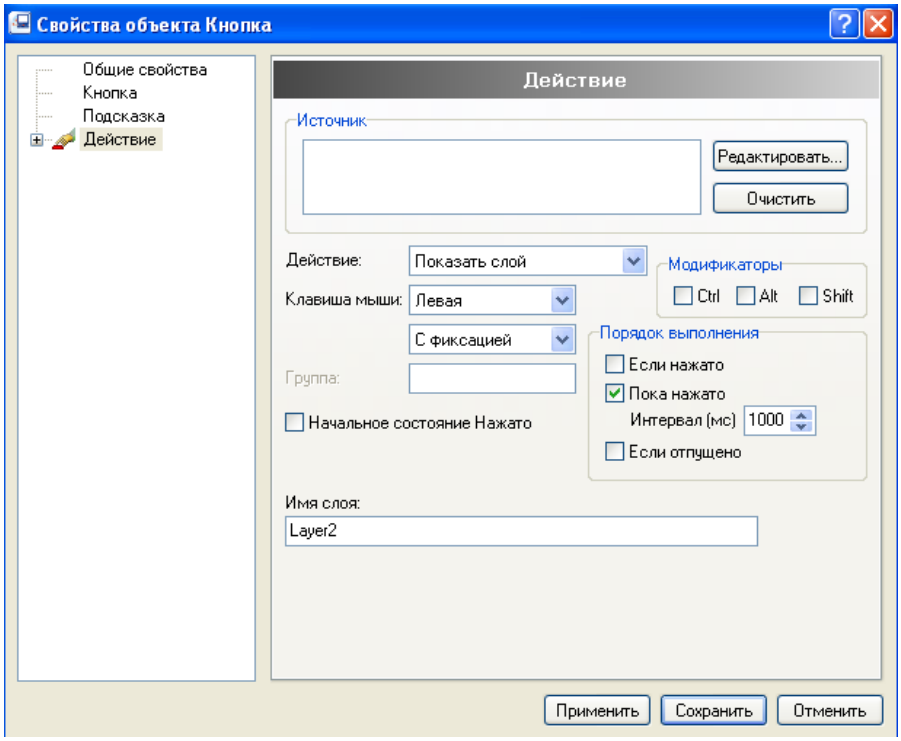

Рис. 7

4. Создайте в слое **Layer1** кнопку **Скрыть слой** и настройте свойства таким образом чтобы при нажатии на нее скрывался слой **Layer2**.

Представьте, что существует некая потребность иногда выводить значения всех сигналов, с которыми связана мнемосхема, не только цветной политикой, но и в цифровом виде. Реализация этого требования в Первичном слое приведёт к перегруженности экранной формы. Это удобно сделать в слое **Layer2**, который не будет виден в основном масштабе экранной формы.

5. Создайте в слое **Layer2** объекты **Значение параметра**, разместите их и привяжите к ним соответствующие сигналы и локальные переменные (Рис. 8).

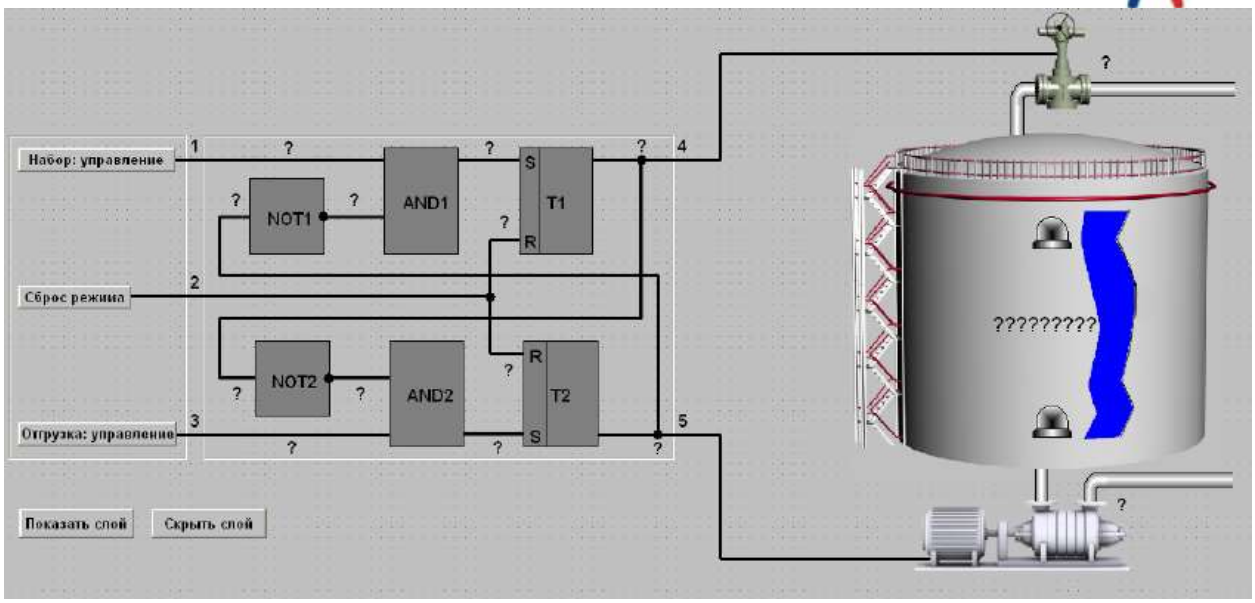

Рис. 8

6. Перейдите в режим **Исполнение,** слой **Layer2** должен быть виден при нажатии кнопки **Показать слой**.

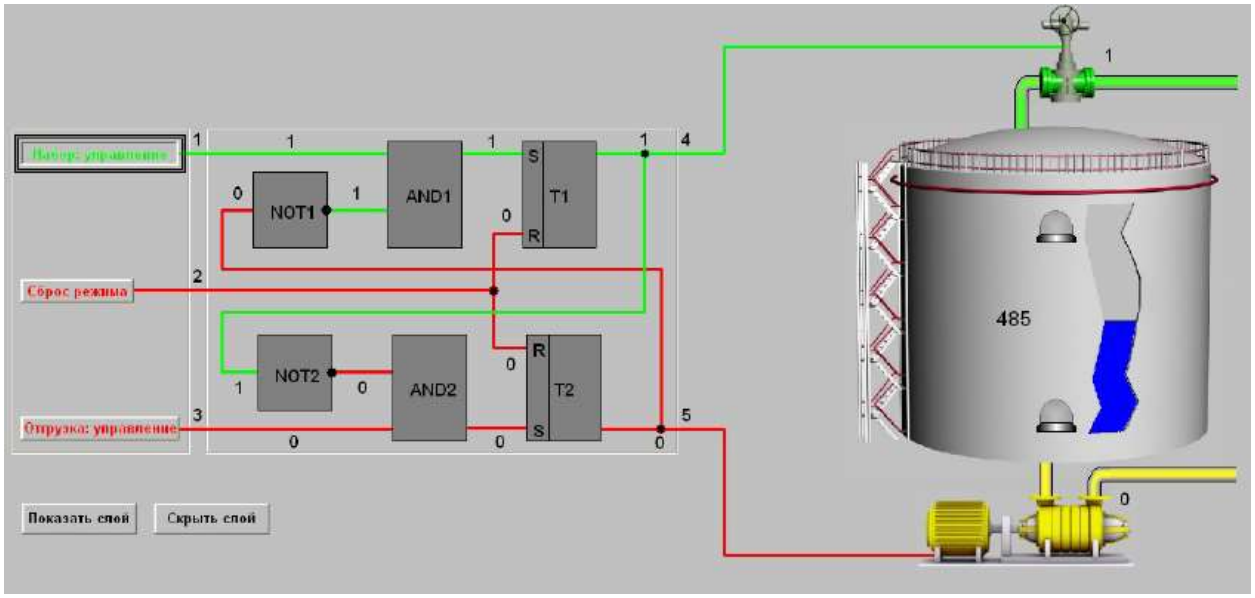

Рис. 9

Упражнение 3: Оптимизация мнемосхемы технологического процесса с помощью функций и условий.

1. Обратите внимание, что уровень в резервуаре при включенной задвижке может далеко превысить 1000 единиц, которые определяют 100% наполнения резервуара. Необходимо модифицировать алгоритм: если сигнал **Level** больше, чем 1000, то ему присваивается значение 1000, если меньше, то сигналу **Level** присваивается текущее значение, определяемое алгоритмом работы процесса. Откройте **Инспектор свойств**, в котором находится формула этого алгоритма, и с помощью функции **if** задайте подобное условие. Пример **x=if (У>1000, 1000, У+З-Н)**, где(**У** — это уровень, **З** — задвижка, **Н** - насос), если **У>1000**, то **У** присваивается **1000**, если **У<1000**, то в **У** записывается вычисленное значение **У+З-Н**. Для проверки работы условия увеличьте скорость набора и отгрузки в **10** раз.

2. Что бы настроить у элемента **Лампочка** динамику **Мигание**, необходимо вызвать Редактирование псевдонимов и ввести условие. Порог 800 единиц. (Рис. 10).

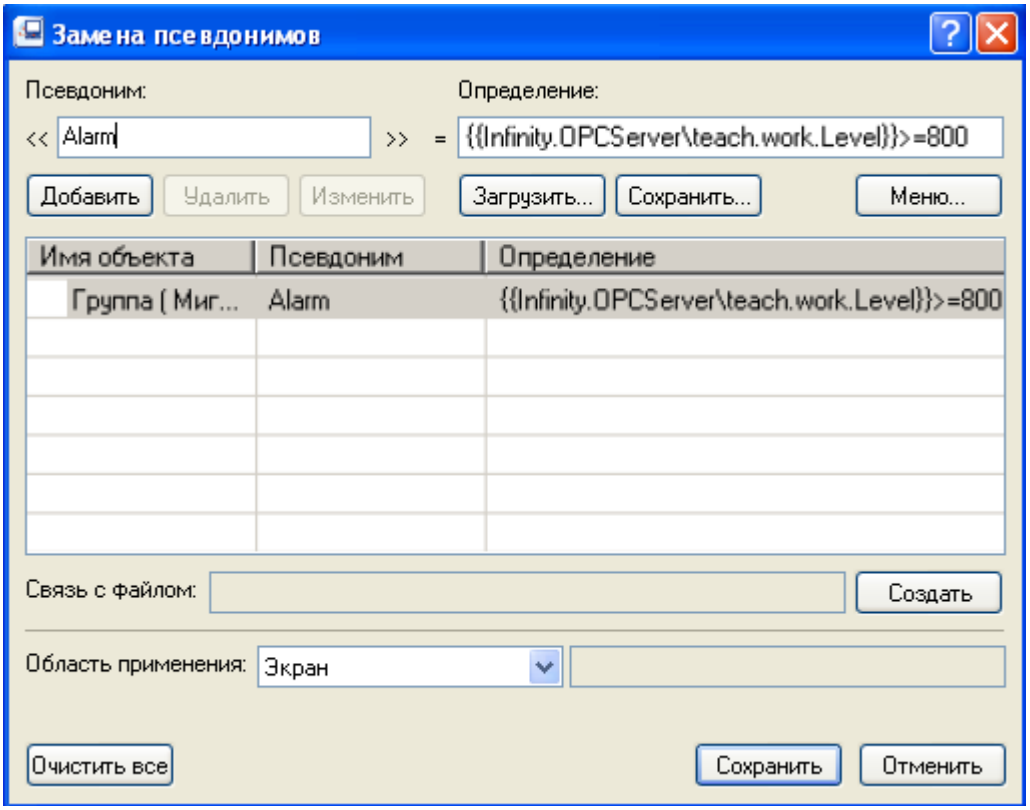

Рис. 10

3. Примените такую же динамику для нижней лампочки, порог для нижнего индикатора уровня – 200 единиц.

4. Проверьте мнемосхему в **Режиме-Исполнение.** При достижении указанных уровней цвет лампочек будет меняться на красный один раз в 250мс. Значение уровня не должно превышать 1000 единиц.

Упражнение 4: Замена локальных переменных на выражения.

Как правило, локальные переменные используются в качестве источников данных для различных анимационных эффектов. Как уже отмечалось, локальные переменные действуют только в пределах одной экранной формы. Также причиной их применения в работах №4 и №5 стало стремление упростить процесс создания мнемосхемы, сделать его ступенчатым. Теперь необходимо модифицировать две формулы для элементов T1 и T2, раскрыв локальные переменные. Это упражнение будет проверкой на внимательность и понимание логики, обрабатываемой в мнемосхеме.

1. Откройте **Свойства объекта** с вкладкой **Динамическоек действие**, которая выполняет функцию триггера. В приведенных примерах это символ триггера **T1** или **T2** (любой из них). В **Редакторе выражений** откройте эту формулу. Ниже приведена формула для **T1**. **x=!(!~~AND1~~&&!**

{{Infinity.OPCServer.Teach.Work.Valve.Control}})&&!

{{ Infinity.OPCServer.Work.Reset}}.

Ваша задача в замене локальных переменных **~~AND1~~** и **~~AND2~~** на выражения, которые вычисляются динамикой, примененной к символам элементов **AND1**, **AND2** и инверторов **NOT1**, **NOT2**. Закройте **Редактор выражений** и С**войства объекта**.

2. Откройте Редактор выражений для формулы, записываемой в **~~invert_Pump.Control~~:!{{ Infinity.OPCServer.Teach.Work.Pump.Control}}**

3.Скопируйте это выражение и подставьте вместо соответствующей локальной переменной в выражение для **AND1**:

{{Infinity.OPCServer.Teach.Work.Valve.In}}&&!

{{Infinity.OPCServer.Work.Pump.Control}}

4.Скопируйте это выражение и подставьте вместо выражения для **T1**:

!(!({{Infinity.OPCServer.Teach.Work.Valve.In}}&&!

{{Infinity.OPCServer.Teach.Work.Pump.Control}})&&!

{{Infinity.OPCServer.Teach.Work.Valve.Control}})&&!

{{Infinity.OPCServer.Teach.Work.Reset}}

5.Выражение, выделенное крупным шрифтом, и есть замена **~~AND1~~**. Обратите внимание на то, что выражение дополнительно заключено в скобки для общей инверсии, как и было для **~~AND1~~**.

6.Проделайте действия пунктов 2-5 для **T2**.

7.Замените локальные переменные в **Источнике данных** для 4 линий (динамика **Цвет**) на выражения инверсии и логического умножения, соответственно. То же проделайте для соответствующих 4 объектов в слое **Parametrs**.

8.Удалите динамику **Динамическое действие** на элементах **AND1**, **AND2** и инверторах **NOT1**, **NOT2** (Рис. 11). Теперь все выражения логики работы триггеров будут вычисляться в двух формулах.

9.Сохраните файл**.**

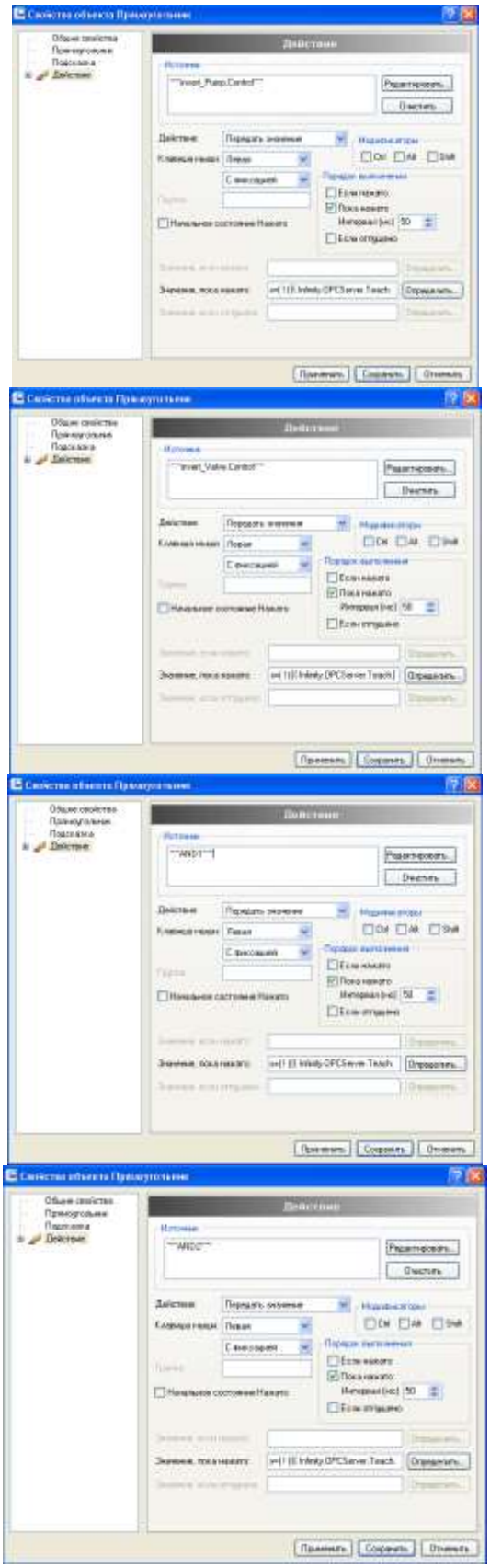

Рис. 11

10.Проверьте работу мнемосхемы. Внешне все должно выглядеть так же, как в мнемосхеме файла **Фамилия7.xml**, но при наведении указателя мыши на элементы **AND1**, **AND2** и инверторы **NOT1**, **NOT2** не будет отображаться применение к ним

динамики **Динамическое действие**. На этом лабораторная работа закончена.