Министерство науки и высшего образования
Российской Федерации

Федеральное государственное автономное
образовательное учреждение высшего образования
«Пермский национальный исследовательский
политехнический университет»

С.И. Сташков

АЛГОРИТМИЗАЦИЯ СИСТЕМ ПРОГРАММНО-ЛОГИЧЕСКОГО УПРАВЛЕНИЯ

Утверждено
Редакционно-издательским советом университета
в качестве учебного пособия

Издательство
Пермского национального исследовательского
политехнического университета
2021

УДК 681.51.015 (075.8)
С788

Рецензенты:
кандидат технических наук, доцент *П.Ю. Сокольчик*
(Пермский национальный исследовательский
политехнический университет);
генеральный директор *С.С.Власов*
(ООО «Промышленная кибернетика», г. Пермь)

Сташков, С.И.

С788 Алгоритмизация систем программно-логического управления: учеб. пособие / С.И. Сташков. – Пермь: Изд-во Перм. нац. исслед. политехн. ун-та, 2021. – 98 с.

ISBN 978-5-398-02651-1

Рассмотрены системы программно-логического управления технологическими процессами и подходы к их алгоритмизации. Приведены примеры алгоритмизации такого вида систем управления с использованием микропроцессорных средств автоматизации.

Предназначено для студентов технических специальностей в области автоматизации технологических процессов и производств очной и заочной форм обучения.

УДК 681.51.015 (075.8)

ISBN 978-5-398-02651-1

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

При управлении технологическими процессами особенности задач управления определяются в зависимости от характера протекания процессов, которые можно разделить на непрерывные и периодические технологические процессы. Непрерывные технологические процессы характеризуются тем, что питание и отгрузка при их ведении производятся непрерывно, системы управления такими процессами по каналу управления являются линейными, и задачи управления в этом случае сводятся к задачам регулирования. Периодические же процессы можно отнести к непрерывно-дискретному классу технологических процессов, для которых характерна нестационарность режимов работы. При этом управление такими процессами должно быть реализовано с учетом развития реальной ситуации во времени (системы реального времени).

Характер дискретных процессов предполагает применение для их проектирования и алгоритмизации аппарата дискретной математики и методов математической логики. Алгоритмы современных систем дискретного и программно-логического управления строятся на основе подходов бинарной, нечеткой и других видов логики, не рассмотренных в настоящем пособии. Настоящее же учебное пособие написано с целью осветить некоторые вопросы таких учебных дисциплин, как «Алгоритмизация и проектирование систем логического управления» и «Системы дискретного управления», и ориентировано большей частью на практические особенности процесса алгоритмизации систем управления. Поэтому для изучения всех вопросов в рамках указанных выше дисциплин приводится список необходимой литературы.

В частности, общие вопросы математической логики и дискретной математики освещены в необходимом для изучения указанных дисциплин объеме в [3; 5; 10]. Также данные работы описывают особенности применения аппарата бинарной логики к описанию работы электроконтактных схем. Способы минимизации памя-

ти логических автоматов подробно рассмотрены в [5; 6]. Кроме того в [6] подробно описаны принципы построения циклограмм работы дискретных устройств. Основы программирования промышленных программно-логических контроллеров, на основе которых строятся современные системы программно-логического управления, рассмотрены в [4]. Разработка систем управления, построенных на подходах нечеткой логики, подробно описана в работах [1; 2; 9]. Современные подходы управления дискретными процессами – switch-технологии – описаны в [7; 8].

Настоящее учебное пособие предназначено для студентов технических специальностей в области автоматизации технологических процессов и производств.

Глава 1. СИСТЕМЫ ЛОГИЧЕСКОГО УПРАВЛЕНИЯ

1.1. Общие сведения

В некоторых случаях весь процесс управления технологическим процессом состоит в реализации заданной последовательности переключательных операций. Такое управление процессами переключательного типа осуществляется в соответствии с некоторой оптимальной стратегией переключения, полученной в виде логического алгоритма управления. Особенности алгоритмов такого типа обусловлены дискретным характером решаемых задач. Управление производственными процессами переключательного типа, обеспечивающее осуществление необходимых логических операций, задаваемых с помощью моделей, при переключении оборудования, называют логическим управлением. Для составления формализованного описания данного производственного процесса переключательного типа с последующей реализацией по нему логического управления широко используют аппарат дискретной математики.

Функционирование сложного технологического объекта может быть представлено в виде некоторого логического описания, т.е. в форме логических высказываний. Многие высказывания являются сложными, т.е. они состоят из лингвистических переменных, связанных логическими операциями. В алгебре логики множество логических операций, но существует базис из пяти операций – конъюнкция, дизъюнкция, отрицание, импликация, эквиваленция. Каждая логическая операция определяется таблицей истинности, представленной в табл. 1.1.

Таблица 1.1

Таблица истинности базовых логических операций

x	y	$x \wedge y$	$x \vee y$	$\overline{x}$	$\overline{y}$	$x \to y$	$x \leftrightarrow y$
0	0	0	0	1	1	1	1
0	1	0	1	1	0	1	0
1	0	0	1	0	1	0	0
1	1	1	1	0	0	1	1

Любое высказывание, полученное таким образом с помощью логических операций, может рассматриваться как высказывательная форма (или формула).

1.2. Понятие операции. Виды операций

Операция, реализуемая в технологическом объекте, – это развернутое во времени целенаправленное действие, которое характеризуется своей целью и способом достижения этой цели. Есть множество различных операций, реализуемых в технологическом объекте. При этом операции могут повторяться неограниченное число раз. Однако считается, что операция завершается только в следующих случаях:

1) по достижении поставленной цели (доведение параметров объекта до заданных значений, перемещение рабочих органов в требуемое положение и т.д.);

2) по истечении отведенного для нее времени;

3) под влиянием внешних событий, прерывающих операцию.

Операции могут быть параллельными и последовательными. Если две операции могут выполняться одновременно (на разном оборудовании технологического объекта), то они называются параллельными. В случае когда начало одной операции совпадает с завершением другой, операции называются последовательными.

1.3. Дискретный процесс и его особенности

Дискретный процесс, реализуемый на технологическом объекте, – это конечное множество операций, на котором заданы бинарные отношения параллельных и последовательных операций.

Дискретный процесс всегда имеет начало и конец, поэтому для него может фиксироваться подмножество начальных и подмножество финальных операций.

Можно выделить два вида дискретных процессов – циклический и конвейерный. Циклический дискретный процесс – это дискретный процесс при котором начальная стадия следует за финальной. Конвейерный дискретный процесс – это дискретный процесс, при котором начальные стадии могут следовать до завершения процесса другой стадии.

Как циклический, так и конвейерный процессы должны удовлетворять двум условиям:

1) любая операция может быть повторно начата только после своего завершения;

2) дискретный процесс, начавшись, всегда может быть доведен до конца, т.е. в ходе его выполнения не должно возникать «тупиковых» ситуаций, не имеющих продолжения.

Таким образом, дискретный процесс, реализуемый на технологическом объекте, представляет собой один из возможных вариантов поведения этого объекта.

Процесс считается корректным:

1) если он соответствует двум приведённым выше условиям;

2) согласован с возможностями того технологического объекта, в котором протекает.

Характерной чертой рассматриваемых процессов является их полная детерминированность, т.е. в явном виде задана логика их функционирования. Следовательно, задана и логика управления, определяющая оптимальную стратегию переключения производственного оборудования при выполнении производственной задачи, для решения которой предназначен данный процесс.

В качестве информации о таком типе процессов применяются сигналы двух уровней, условно обозначаемых символами «0» и «1». Два уровня сигнала соответствуют двум возможным состояниям параметра.

Состояние объекта в каждый момент времени характеризуется совокупностью дискретных значений (0 или 1) достаточно большого числа параметров и может быть представлено таблицей состояний (таблицей истинности). Каждая строка этой таблицы соответствует одному из возможных состояний объекта, для оценки свойств которого используется N параметров. В каждой строке фиксируется один из возможных наборов значений входных параметров x и соответствующее им значение выходного параметра y. При этом часть таблицы, связанная с входами исследуемой системы, зависит только от числа входов k и не зависит от специфики решаемой задачи. Столбцы же, описывающие выходные переменные, напротив, заполняются только с учетом специфики решаемой

задачи. Глубина таблицы определяется равной 2^k, а чередование нулей и единиц в части таблицы, описывающей входные переменные x, определяется интервалом $m = 2^{k-1}$ (см. табл. 1.2). При этом индекс входных переменных k в заголовке таблицы должен возрастать справа налево.

Таким образом, задача алгоритмизации процессов переключательного типа сводится к отысканию функций, обеспечивающих выполнение заданных логических отношений между совокупностью значений входных параметров x и выходных управляемых параметров y.

Большинство встречающихся на практике логических автоматов, описывающих «логику» процесса управления переключательного типа, относятся к категории систем управления с памятью, в которых набор выходных сигналов, выработанных в некоторый момент времени, зависит не только от входных сигналов, поданных в тот же момент, но и от сигналов, поступивших ранее. Таким образом, реакция такого автомата определяется набором входных сигналов x и внутренним состоянием z.

Традиционный подход к проектированию систем логического управления связывает процесс создания математической модели системы с техническими средствами, на которых она реализуется. Важное место при проектировании систем логического управления занимает системный подход, разработка и совершенствование математических методов синтеза систем. Исторически особое место занимают матричные методы анализа и синтеза систем, методы формализации и методы минимизации памяти логических автоматов.

1.4. Минимизация памяти логических автоматов

Область определения логической функции с любым числом переменных можно представить в виде гиперкуба. Любую логическую функцию трех переменных можно отобразить в виде куба, на котором элементы куба соответствуют всевозможным конъюнкциям логической функции трех переменных. Так, вершины куба соответствуют конъюнкциям 3-го ранга, ребра – 2-го, а грани куба – конъюнкциям одной переменной.

Минимизация булевых функций на гиперкубе решается в классе совершенной дизъюнктивной нормальной формы (СДНФ) и предполагает следующие шаги:

1) строится куб;

2) отмечаются согласно данной логической функции вершины, ребра, грани;

3) минимизируется функция с помощью понижения рангов при совпадении.

Недостаток данного способа минимизации очевиден: при количестве переменных более трех возникает затруднение с графической интерпретацией состояний объекта управления. Поэтому существует способ минимизации булевых функций в виде развертки гиперкуба. Такой метод минимизации получил название карт Карно. Минимизация методом карт Карно – это графический способ минимизации булевых функций.

Карта Карно – это плоская развертка *n*-мерного булевого куба (гиперкуба), представляющая собой операции попарного неполного склеивания и элементарного поглощения.

Рассмотрим пример.

Логическая функция четырех переменных $f(x_1, x_2, x_3, x_4)$ принимает значение, равное единице, на следующих наборах аргументов: 0, 2, 4, 5, 6, 7, 9, 11. Необходимо провести минимизацию данной функции методом карт Карно и оценить эффективность минимизации.

Решение. Составим таблицу истинности для заданной функции. Глубина таблицы определяется $2^k = 2^4 = 16$. Таким образом, таблица имеет 16 строк, нумерация которых начинается с «0». Определим чередование «0» и «1» для каждой переменной. Для x_1 рассчитаем $m(x_1) = 2^{k-1} = 2^{1-1} = 2^0 = 1$. Аналогично рассчитаем $m(x_2) = 2^{k-1} = 2^{2-1} = 2^1 = 2$, $m(x_3) = 2^{k-1} = 2^{3-1} = 2^2 = 4$ и $m(x_4) = 2^{k-1} = 2^{4-1} = 2^3 = 8$. Таким образом, чередование «0» и «1» в столбце x_1 осуществляется каждую строчку, x_2 – через строчку, x_3 – каждые четыре строки и x_4 – каждые восемь строк. Полученная таблица истинности представлена в табл. 1.2.

Таблица 1.2

Таблица истинности

№	x_4	x_3	x_2	x_1	$y = f(x_1, x_2, x_3, x_4)$
0	0	0	0	0	1
1	0	0	0	1	0
2	0	0	1	0	1
3	0	0	1	1	0
4	0	1	0	0	1
5	0	1	0	1	1
6	0	1	1	0	1
7	0	1	1	1	1
8	1	0	0	0	0
9	1	0	0	1	1
10	1	0	1	0	0
11	1	0	1	1	1
12	1	1	0	0	0
13	1	1	0	1	0
14	1	1	1	0	0
15	1	1	1	1	0

Запишем СДНФ этой функции и проведем ее минимизацию. Для этого запишем переменные, стоящие в конъюнкциях, со знаком отрицания, если они принимают значение «0», и без отрицания – если «1». СДНФ данной функции будет иметь вид:

$$y = (\overline{x_1} \wedge \overline{x_2} \wedge \overline{x_3} \wedge \overline{x_4}) \vee (\overline{x_1} \wedge x_2 \wedge \overline{x_3} \wedge \overline{x_4}) \vee (\overline{x_1} \wedge \overline{x_2} \wedge x_3 \wedge \overline{x_4}) \vee$$
$$\vee (x_1 \wedge \overline{x_2} \wedge x_3 \wedge \overline{x_4}) \vee (\overline{x_1} \wedge x_2 \wedge x_3 \wedge \overline{x_4}) \vee (x_1 \wedge x_2 \wedge x_3 \wedge \overline{x_4}) \vee$$
$$\vee (x_1 \wedge \overline{x_2} \wedge \overline{x_3} \wedge x_4) \vee (x_1 \wedge x_2 \wedge \overline{x_3} \wedge x_4).$$

Для получения минимизированной дизъюнктивной нормальной формы (МДНФ) составим карту Карно. При этом чередование операции отрицания пар переменных осуществляется в соответствии с кодом Грея, как показано в табл. 1.3.

На основании составленной карты получим МДНФ:

$$y = (x_1 \wedge \overline{x_3} \wedge x_4) \vee (x_3 \wedge \overline{x_4}) \vee (\overline{x_1} \wedge \overline{x_4}).$$

Таблица 1.3

Карта Карно

Переменные	$\overline{x_3}\,\overline{x_4}$	$\overline{x_3}x_4$	x_3x_4	$x_3\overline{x_4}$
$\overline{x_1}\,\overline{x_2}$	1	0	0	1
$\overline{x_1}x_2$	1	0	0	1
x_1x_2	0	1	0	1
$x_1\overline{x_2}$	0	1	0	1

Оценим степень выполненного процесса минимизации. Для этого сравним ранги исходного и минимизированного выражения. Ранг СДНФ заданного выражения составляет 49, а МДНФ – 10.

1.5. Контрольные вопросы и задания

1) Написать таблицу истинности логических операций «отрицание», «конъюнкция», «дизъюнкция», «импликация» и «эквиваленция».

2) Записать следующий текст высказывательной формулой:

Вариант 1. «Человек не успеет на самолет, если не сможет вовремя прибыть в аэропорт, что возможно только тогда, когда его часы идут неверно или вызванное им такси задержится».

Вариант 2. «Если актеры приедут на гастроли, то концерт состоится только тогда, когда все билеты будут проданы и не будет дождя».

Вариант 3. «Кандидат победит на выборах, если наберет более 50 % голосов, которые состоятся только тогда, когда на выборы придет минимум 30 % населения, и не более 5 % бюллетеней будут считаться недействительными».

Вариант 4. «Строительство моста будет закончено в срок, если будут все необходимые для этого строительные материалы, которые поставят только тогда, когда будет подписан договор с транспортной компанией и бетонный завод закончит изготовление несущих конструкций».

3) Построить таблицу истинности логической функции трех аргументов $y(x_1x_2x_3)$ для значений y, заданных в табл. 1.4.

Таблица 1.4

Значения выходной переменной *y*

Вариант 1	Вариант 2	Вариант 3	Вариант 4
0	1	1	0
1	0	0	1
0	1	0	1
1	1	1	0
1	0	0	1
0	0	1	0
0	1	1	0
1	1	0	1

4) На основании построенной таблицы (задание № 3) записать *y* в формах СДНФ и СКНФ.

5) Определить ранг полученных формул.

6) Упростить выражения.

7) Оценить ранг упрощенных выражений.

8) Минимизировать функцию четырех переменных методом карт Карно. Значения *y* для функции четырех переменных заданы в табл. 1.5.

Таблица 1.5

Значения *y* для функции четырех переменных

Вариант 1	Вариант 2	Вариант 3	Вариант 4
0	0	1	0
0	0	1	0
0	0	1	0
0	0	1	0
0	0	0	1
1	0	0	1
0	1	0	0
0	0	1	1
1	0	0	0
1	1	0	0
1	0	0	1
0	0	0	1
1	1	0	0
1	1	0	0
0	1	0	0
1	1	1	0

Глава 2. АЛГОРИТМИЗАЦИЯ СИСТЕМ ПРОГРАММНО-ЛОГИЧЕСКОГО УПРАВЛЕНИЯ НА БАЗЕ SIEMENS SIMATIC S7-300

2.1. Siemens Simatic S7-300: общие сведения

Внешний вид промышленного контроллера Siemens Simatic S7-300 представлен на рис. 2.1.

Рис. 2.1. Siemens Simatic S7-300:
1 – индикаторы состояния; *2* – переключатель

Simatic Siemens S7-300 является модульным программируемым контроллером, предназначенным для решения задач автоматизации низкого и среднего уровня сложности.

Программируемые контроллеры S7-300 могут включать в свой состав:

– модуль центрального процессора (CPU). В зависимости от степени сложности решаемых задач в программируемом контроллере могут использоваться более 20 типов центральных процессоров;

– блоки питания (PS) для питания контроллера от сети переменного или постоянного тока;

– сигнальные модули (SM), предназначенные для ввода и вывода дискретных и аналоговых сигналов, в том числе FailSafe и моду-

ли со встроенными Ex-барьерами. Поддерживаются отечественные ГОСТ градуировки термометров сопротивления и термопар;

– коммуникационные процессоры (CP) – интеллектуальные модули, выполняющие автономную обработку коммуникационных задач в промышленных сетях AS-Interface, PROFIBUS, Industrial Ethernet, PROFINET и системах PtP связи. Применение загружаемых драйверов для CP 341 позволяет расширить коммуникационные возможности контроллера поддержкой обмена данными в сетях MODBUS RTU и Data Highway. Для организации модемной связи в составе S7-300 могут использоваться коммуникационные модули семейства SINAUT ST7;

– функциональные модули (FM) – интеллектуальные модули, оснащенные встроенным микропроцессором и способные выполнять задачи автоматического регулирования, взвешивания, позиционирования, скоростного счета, управления перемещением и т.д. Целый ряд функциональных модулей способен продолжать выполнение возложенных на них задач даже в случае остановки центрального процессора;

– интерфейсные модули (IM) для подключения стоек расширения к базовому блоку контроллера, позволяющие использовать в системе локального ввода-вывода до 32 модулей различного назначения. Модули IM 365 позволяют создавать 2-, модули IM 360 и IM 361 – 2-, 3- и 4-рядные конфигурации.

Все модули устанавливаются на профильную шину S7-300 и фиксируются в рабочих положениях винтами. Объединение модулей в единую систему выполняется с помощью шинных соединителей (входят в комплект поставки каждого модуля), устанавливаемых на тыльной части корпуса.

Преимуществом ПЛК Simatic Siemens S7-300 является произвольный порядок размещения модулей в монтажных стойках. Фиксированные посадочные места занимают только модули PS, CPU и IM. Наличие съемных фронтальных соединителей (заказываются отдельно), позволяют производить быструю замену модулей без демонтажа их внешних цепей и упрощают выполнение операций подключения внешних цепей модулей. Механическое кодирование фронтальных соединителей исключает возможность возникновения

ошибок при замене модулей. Применение гибких и модульных соединителей TOP Connect существенно упрощает выполнение монтажных работ и снижает время их выполнения.

Все центральные процессоры S7-300 характеризуются следующими показателями:

– высокое быстродействие;

– загружаемая память в виде микрокарты памяти MMC емкостью до 8 МБ (MMC используется для загрузки программы, сохранения данных при перебоях в питании CPU, хранения архива проекта с символьной таблицей и комментарии, а также для архивирования промежуточных данных);

– развитые коммуникационные возможности, одновременная поддержка большого количества активных коммуникационных соединений;

– работа без буферной батареи.

Центральные процессоры CPU 3xxC и CPU 31xT-2 DP оснащены набором встроенных входов и выходов, а их операционная система дополнена поддержкой технологических функций, что позволяет использовать их в качестве готовых блоков управления.

Типовой набор встроенных технологических функций позволяет решать задачи скоростного счета, измерения частоты или длительности периода, ПИД-регулирования, позиционирования, перевода части дискретных выходов в импульсный режим. Все центральные процессоры S7-300 оснащены встроенным интерфейсом MPI, который используется для программирования, диагностики и построения простейших сетевых структур.

Некоторые технические характеристики центральных процессоров S7-300 приведены в табл. 2.1.

Система команд центральных процессоров включает в свой состав более 350 инструкций и позволяет выполнять:

– логические операции, операции сдвига, вращения, дополнения, операции сравнения, преобразования типов данных, операции с таймерами и счетчиками;

– арифметические операции с фиксированной и плавающей точкой, извлечение квадратного корня, логарифмические операции, тригонометрические функции, операции со скобками;

– операции загрузки, сохранения и перемещения данных, операции переходов, вызова блоков и др.

Таблица 2.1

Основные технические характеристики центральных процессоров S7-300

CPU	312	312C	313C
Рабочая память	32КБ	32КБ	64КБ
Загружаемая память (MMC)	64КБ-4 МБ	64КБ-4МБ	64КБ-8МБ
Время выполнения операций, мкс:	–	–	–
• логических	0,2	0,2	0,1
• с фиксированной точкой	5,0	5,0	2,0
• с плавающей точкой	6,0	6,0	3,0
Количество флагов/таймеров/счетчиков	1024/128/ 128	1024/128/ 128	2048/256/ 256
Количество каналов ввода-вывода – дискретных / аналоговых, не более	256/64	256/64	1016/253
Встроенные интерфейсы	MPI	MPI	MPI
Количество активных коммуникационных соединений, не более	6	6	8
Количество встроенных	–	–	–
• дискретных входов/ выходов	–	10/6	24/16
• аналоговых входов/ выходов	–	–	4 AI (I/U) + + 1 AI (Pt100)/2 AO
Встроенные функции:	–	–	–
• скоростные счетчики, кГц	–	2×10	3×30
• импульсные выходы, кГц	–	2×2,5	3×2,5
• ПИД-регулирование	–	Нет	Есть
• позиционирование	–	Нет	Нет
Габариты, мм	40×125×130	80×125×130	120×125×130

Программирование контроллера осуществляется с помощью инструментального программного пакета STEP7 для Windows 98/NT/2000/XP.

STEP7 является специализированным языком для программирования контроллеров SIMATIC S7 и предназначен в основном для обработки и выполнения логических операций.

Редактор программ STEP7 содержит языки программирования по стандарту IEC 61131-3 FBD (язык функциональных блоков), LD (язык лестничных диаграмм), STL (список инструкций).

2.2. Загрузка конфигурации и программы в контроллер

Контроллер S7-300 (см. рис. 2.1) имеет индикаторы состояния:

– SF – индикатор ошибок (загорается при наличии групповой ошибки, ошибки программирования и в случае отказа модуля);

– BATF – индикатор батареи (загорается, когда батарея отсутствует или разряжена);

– DC5V – индикатор наличия напряжения 5 В;

– RAN – первый индикатор режима работы (мигает при запуске CPU и горит в режиме работы);

– STOP – второй индикатор режима работы (горит при режиме останова, редко мигает при запросе сброса памяти, часть мигает при выполнении сброса памяти).

Также контроллер S7-300 имеет переключатель, предназначенный для выбора режима работы контроллера:

– RAN (режим работы);

– STOP (режим останова);

– MRES (сброс памяти).

Сброс памяти необходим:

– в случае начального запуска;

– перед загрузкой новой конфигурации и программы;

– если CPU запрашивает сброс памяти (мигает индикатор STOP).

Сброс памяти осуществляется последовательностью действий:

– включение питания;

– перевод переключателя в режим STOP;

– перевод переключателя в режим MRES и удерживание его в таком положении, пока вновь не засветится индикатор STOP;

– перевод переключателя в режим STOP;

– второй перевод переключателя в режим MRES и удерживание его в таком положении, пока вновь не засветится индикатор STOP;

– перевод переключателя в режим STOP.

2.3. Создание программы в Simatic Manager

Для начала работы необходимо создать новый проект в среде Simatic Manager. Для этого нужно выбрать в меню File пункт New. В появившемся окне (рис. 2.2) в поле Name необходимо ввести имя создаваемого проекта и нажать «Ok». При этом проект будет сохранен в указанной в поле Storage location директории.

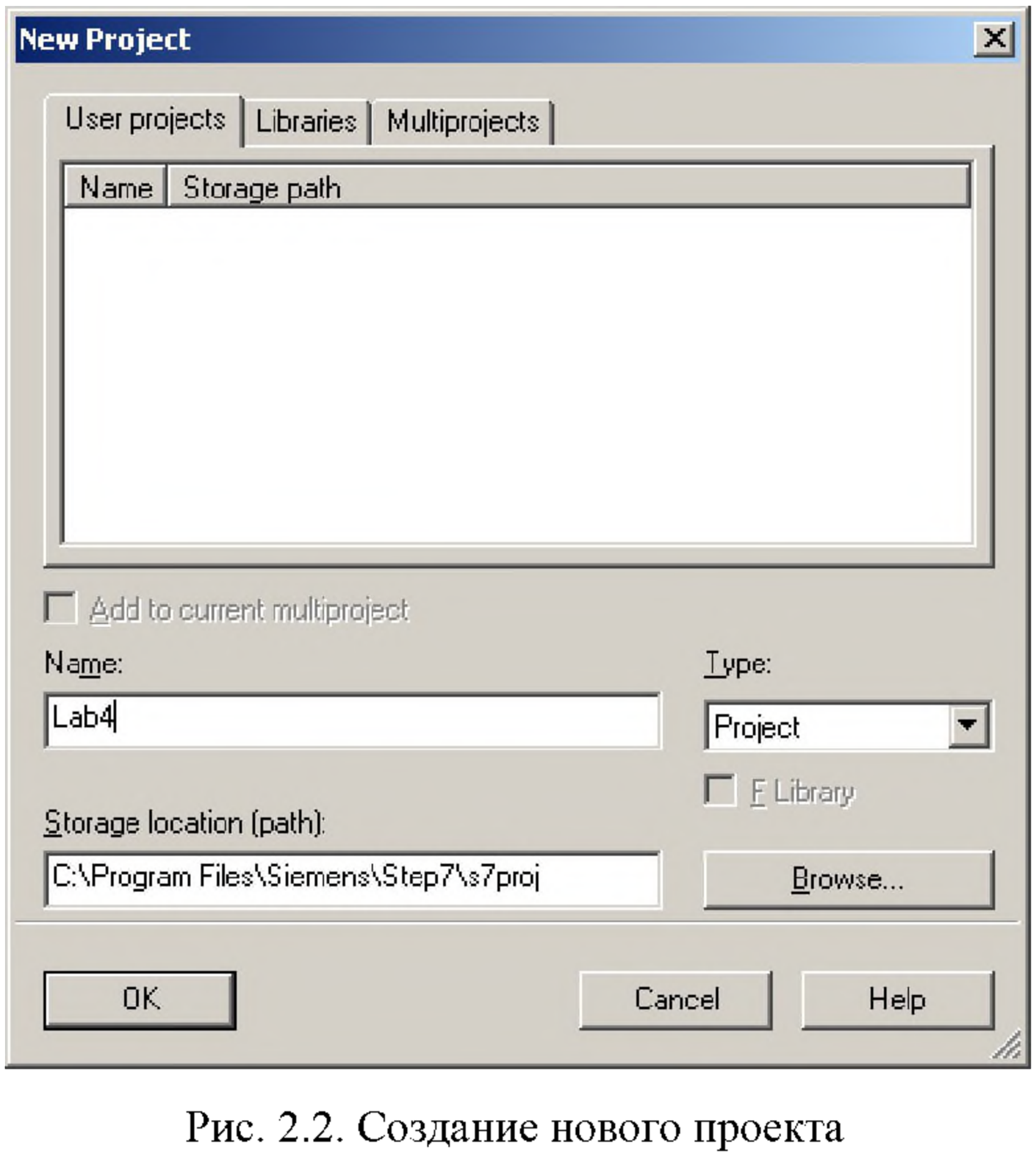


Рис. 2.2. Создание нового проекта

В окне нового проекта необходимо создать новую станцию контроллера SIMATIC 300 Station. Для этого правой клавишей мыши вызывается контекстное меню и во вкладке Insert new object выбирается SIMATIC 300 Station (рис. 2.3).

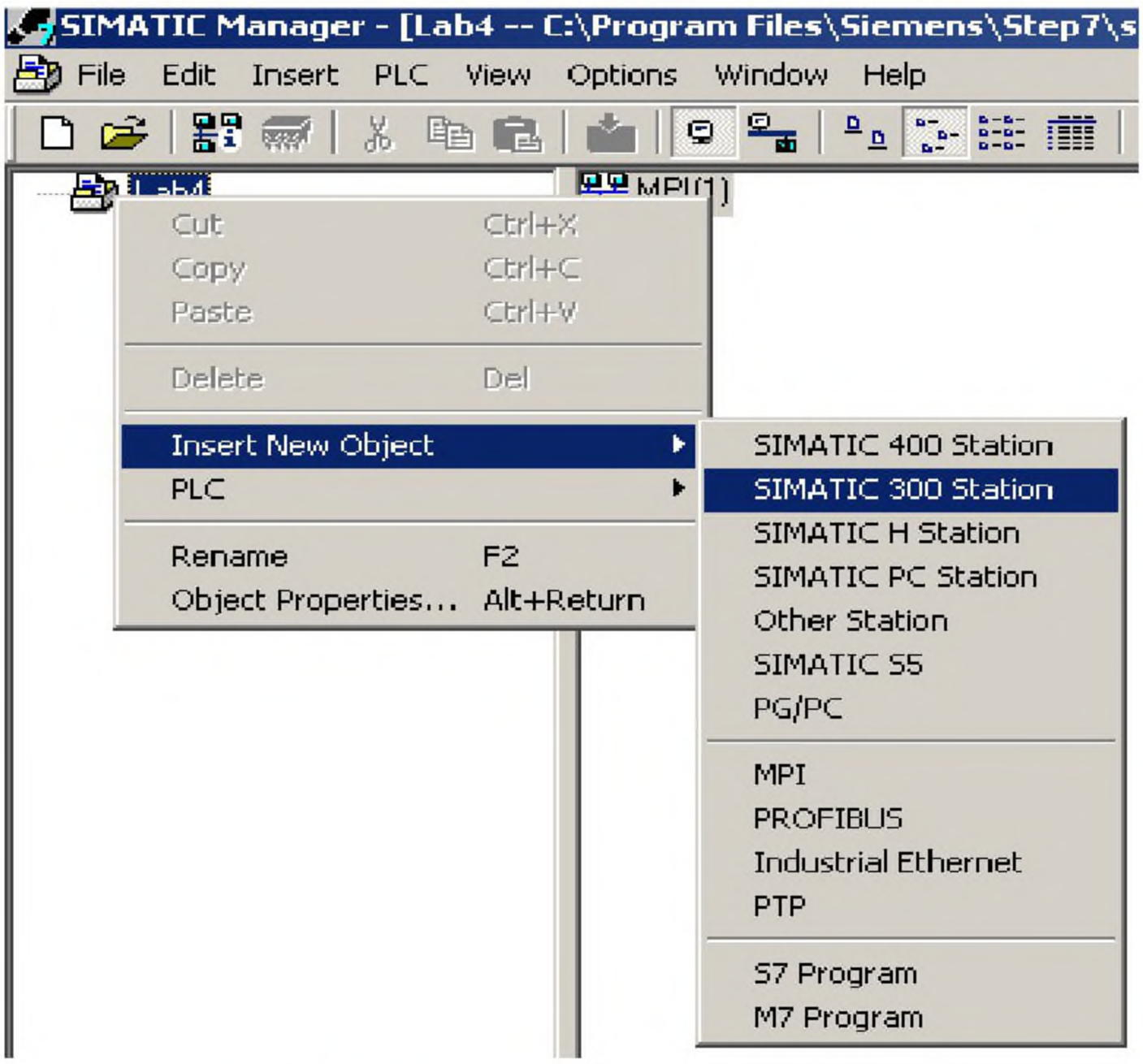


Рис. 2.3. Создание новой станции

Для конфигурирования аппаратной части контроллера необходимо перейти в дереве аппаратов проекта в каталог S7-300, при этом в рабочей области окна Simatic Manager будет доступен пакет Hardware (рис. 2.4). Двойным нажатием левой клавиши мыши на Hardware необходимо вызвать окно конфигурации (рис. 2.5).

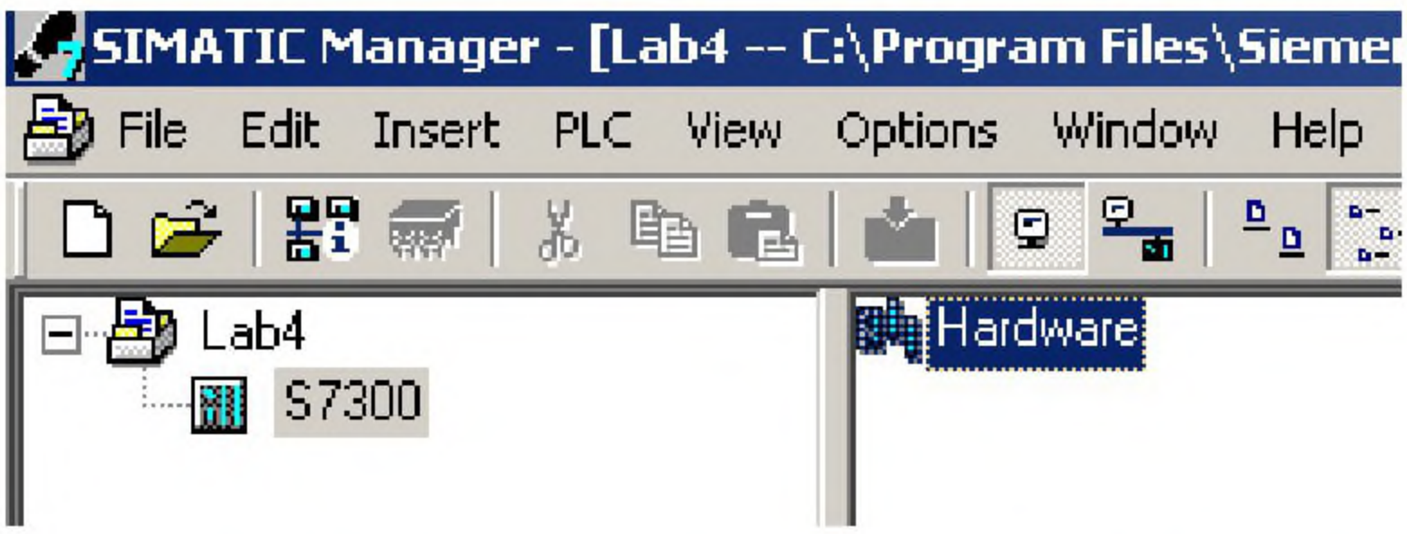


Рис. 2.4. Создание аппаратной конфигурации контроллера

При создании аппаратной конфигурации из библиотеки аппаратных модулей на рабочий лист в окне конфигурации переносятся последовательно следующие объекты:

– монтажная рейка для контроллера Rack 300;

– системный блок питания PS 307 5A;

– необходимый процессорный модуль CPU.

В результате конфигурация будет иметь вид, представленный на рис. 2.5.

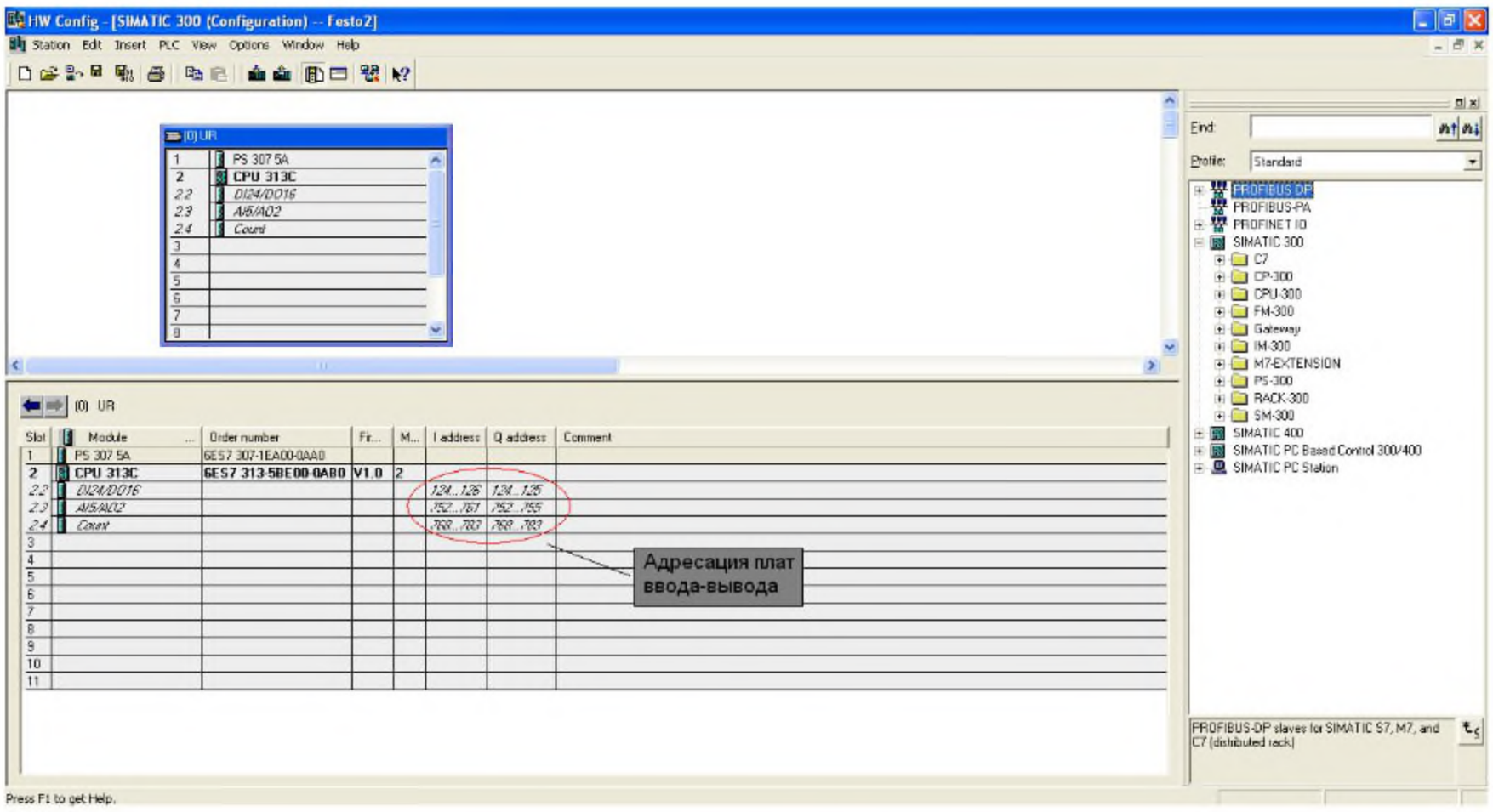


Рис. 2.5. Окно конфигурации

Полученную аппаратную конфигурацию необходимо скомпилировать и прогрузить в контроллер. Для этого в окне HW config сперва необходимо нажать на кнопку Save and compile а затем на Download to module (рис. 2.6).

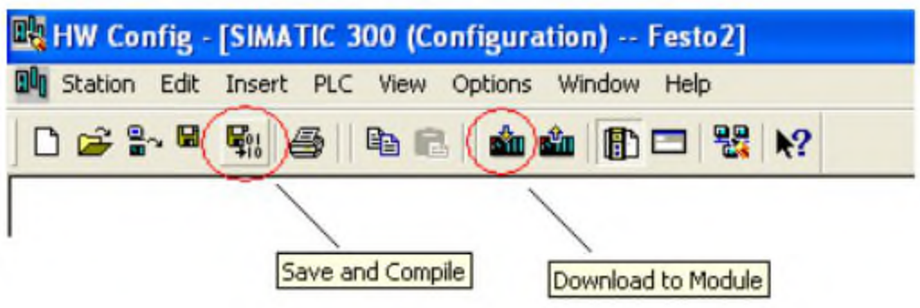


Рис. 2.6. Компилирование и загрузка конфигурации в контроллер

После загрузки в контроллер его аппаратной конфигурации можно начинать разработку программы пользователя контроллера. Основным программным модулем, который исполняется контроллером каждый цикл сканирования, является организационный блок OB1,

который появляется при создании программы автоматически. Другие блоки можно создать, кликнув правой клавишей мыши на категории Blocs в дереве категорий. Для вызова окна свойств блока необходимо кликнуть по нему правой клавишей мыши и выбрать в открывшемся контекстном окне пункт Properties (рис. 2.7).

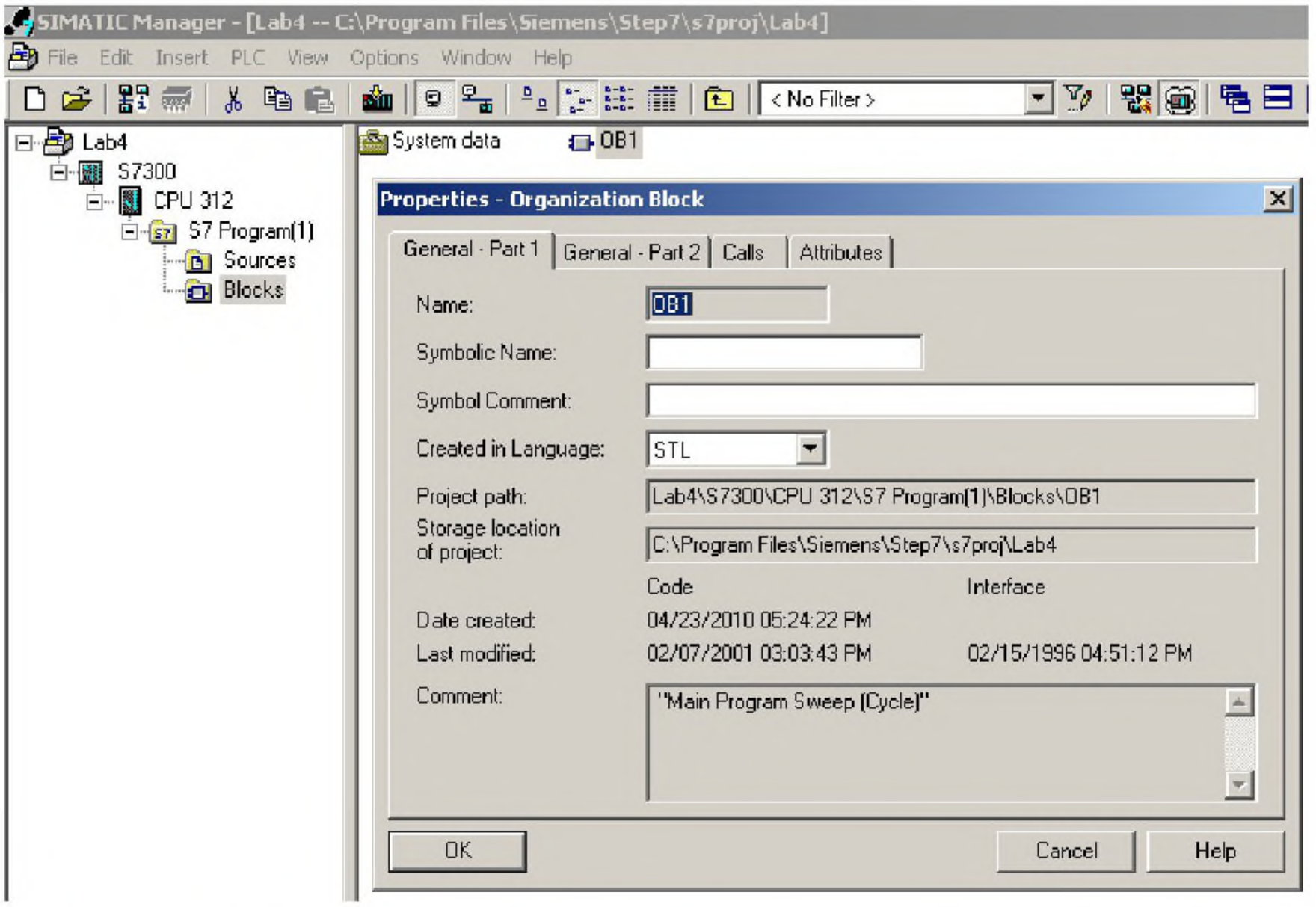


Рис. 2.7. Основные свойства организационного блока OB1

На рис. 2.8 представлен экран среды разработки программного кода программы пользователя контроллера.

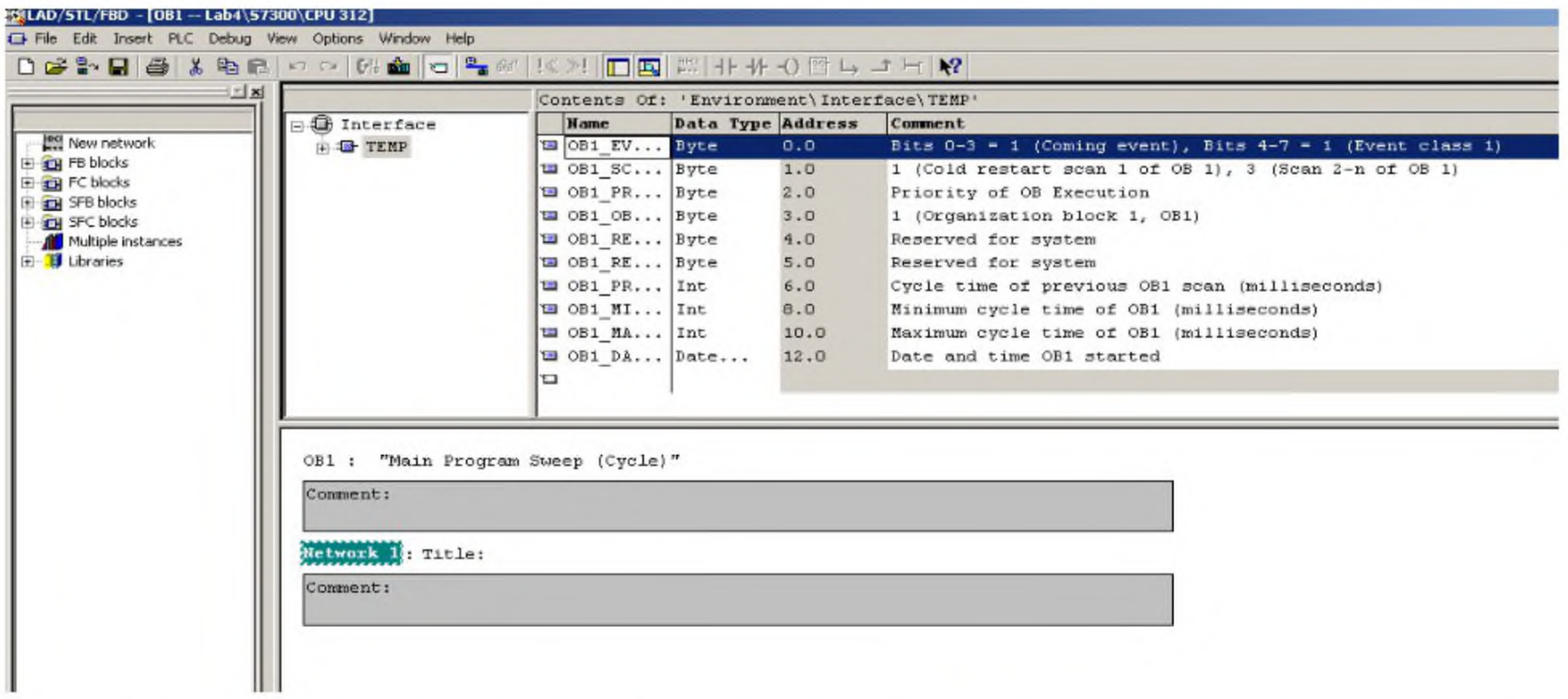


Рис. 2.8. Основной экран среды разработки программных модулей S7

Для вызова окна с содержимым блока OB1 необходимо два раза кликнуть левой клавишей мыши на самом блоке. Для удобства дальнейшей работы рекомендуется создать символьные имена переменным в памяти контроллера, по которым к ним можно будет адресоваться из программных модулей программы пользователя (рис. 2.9). Для этого в окне LAD/CTL/FBD для блока OB1 необходимо выбрать в меню Options пункт symbol table. В открывшемся окне в таблице в колонке symbol необходимо указать имя переменной, в колонке address необходимо указать адрес входа/выхода платы УСО контроллера, в колонке Data type указать тип данных.

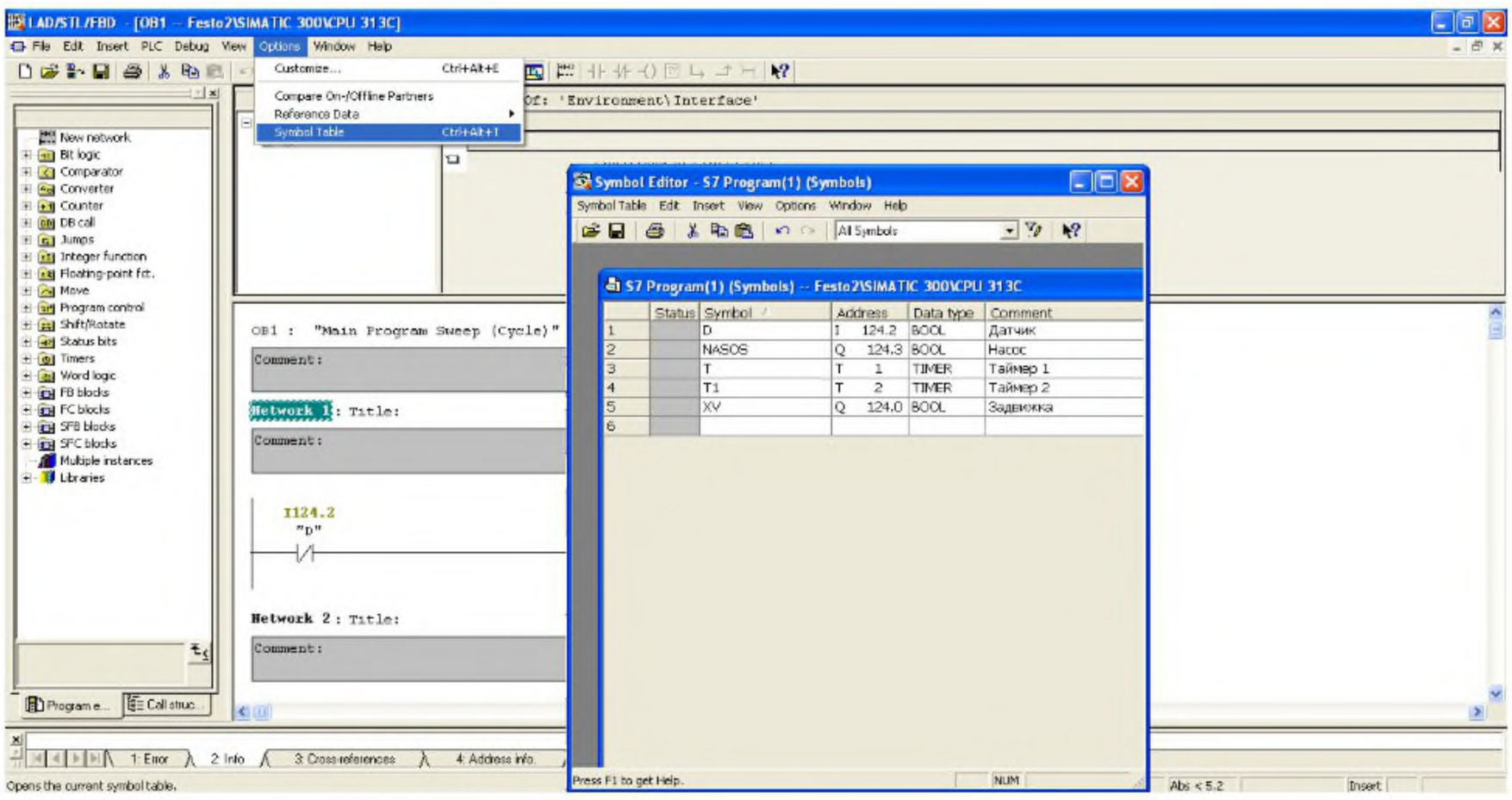


Рис. 2.9. Создание символьных имен

В программе «ПАЗ» будет использован язык LAD (Ladder Diagram). Данный язык удобен для программирования небольших задач. В случае необходимости программный пакет STEP7 позволяет переходить на другие языки при написании программы. Следует отметить, что обратный переход не всегда возможен. Выбор языка осуществляется во вкладке wiew окна LAD/CTL/ FBD (рис. 2.10).

В левой части окна LAD/STL/FBD, представленном на рис. 2.11, находится панель, в которой отображены основные блоки STEP7.

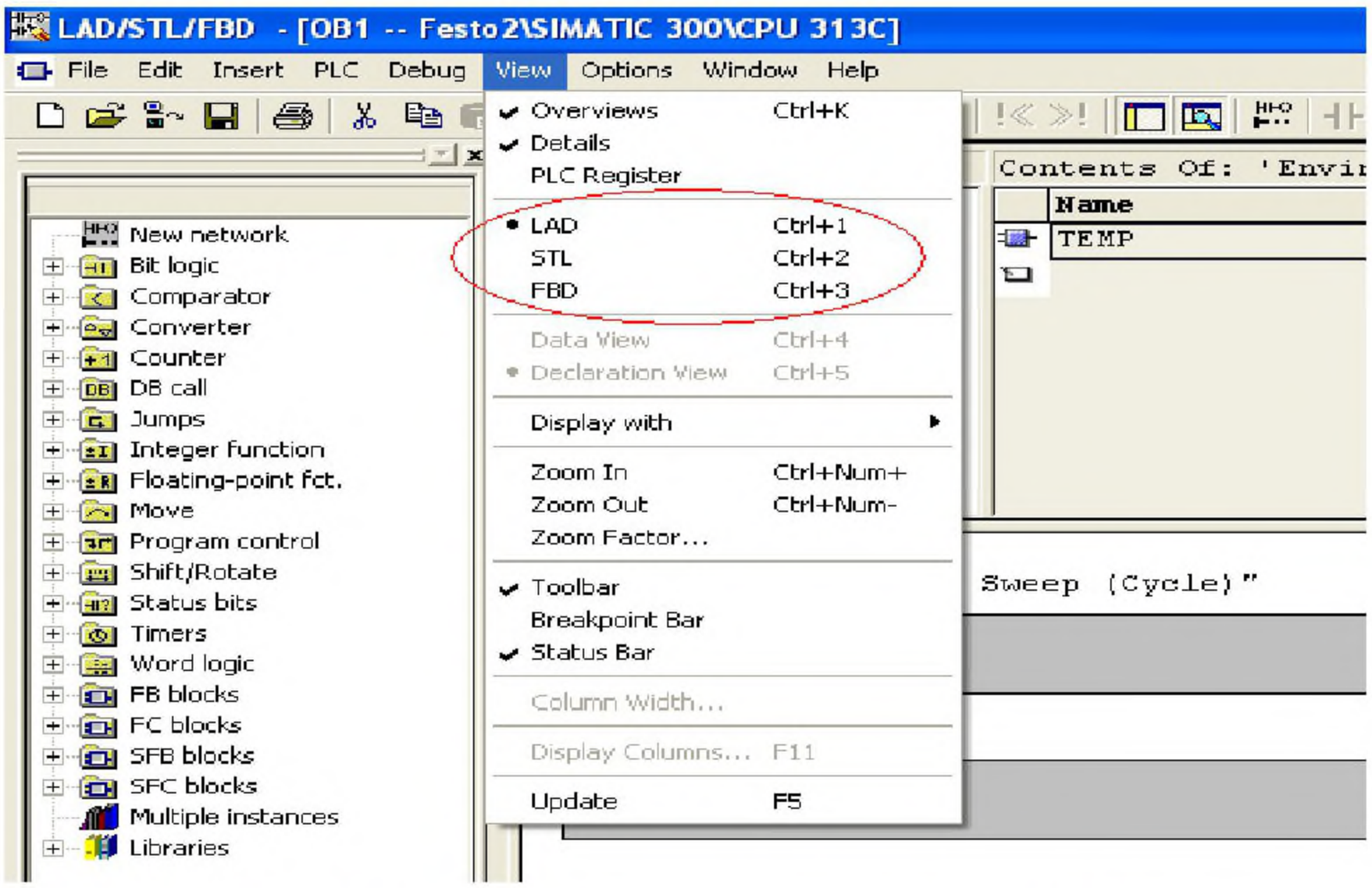


Рис. 2.10. Переход между языками в STEP7

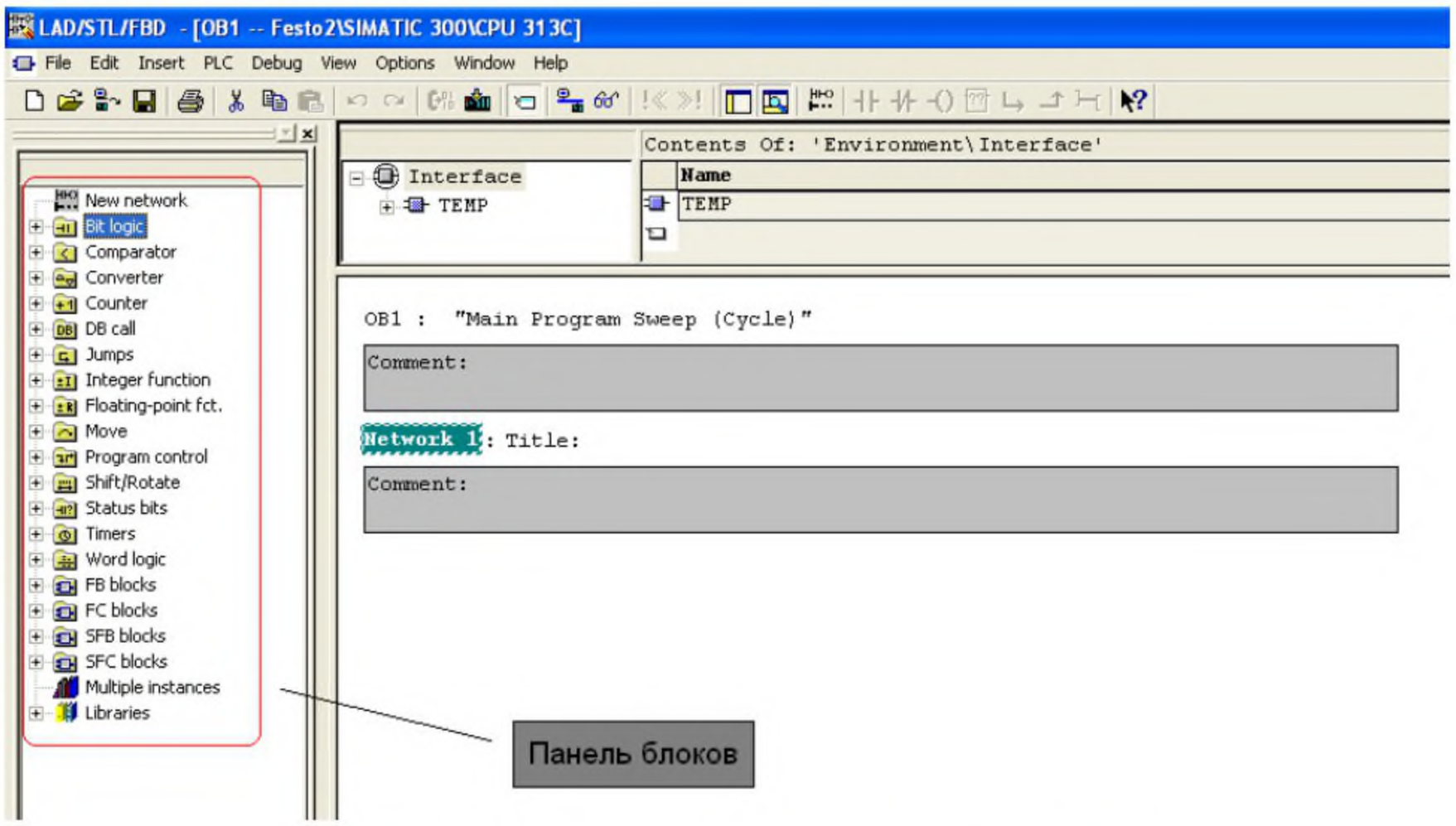


Рис. 2.11. Общий вид панели блоков

2.4. Контрольные вопросы

1. Система программирования STEP7.
2. Языки программирования, входящие в состав МЭК 61131-3 (пять языков).
3. Среда Simatic Manager: общие сведения, поддерживаемые языки программирования, этапы создания проекта.
4. Понятие ПАК, ПЛК, ПЛУ.
5. Контроллеры Simatic Siemens S7-300.
6. Загрузка конфигурации и программы пользователя в Simatic Siemens S7-300.
7. Платы УСО ПАК: классификация, виды сигналов, интерфейсы, протоколы.

Глава 3. АЛГОРИТМИЗАЦИЯ СИСТЕМЫ ПРОГРАММНО-ЛОГИЧЕСКОГО УПРАВЛЕНИЯ СТАНЦИЕЙ РОЗЛИВА

3.1. Описание установки

Внешний вид установки Festo «Станция розлива» представлен на рис. 3.1.

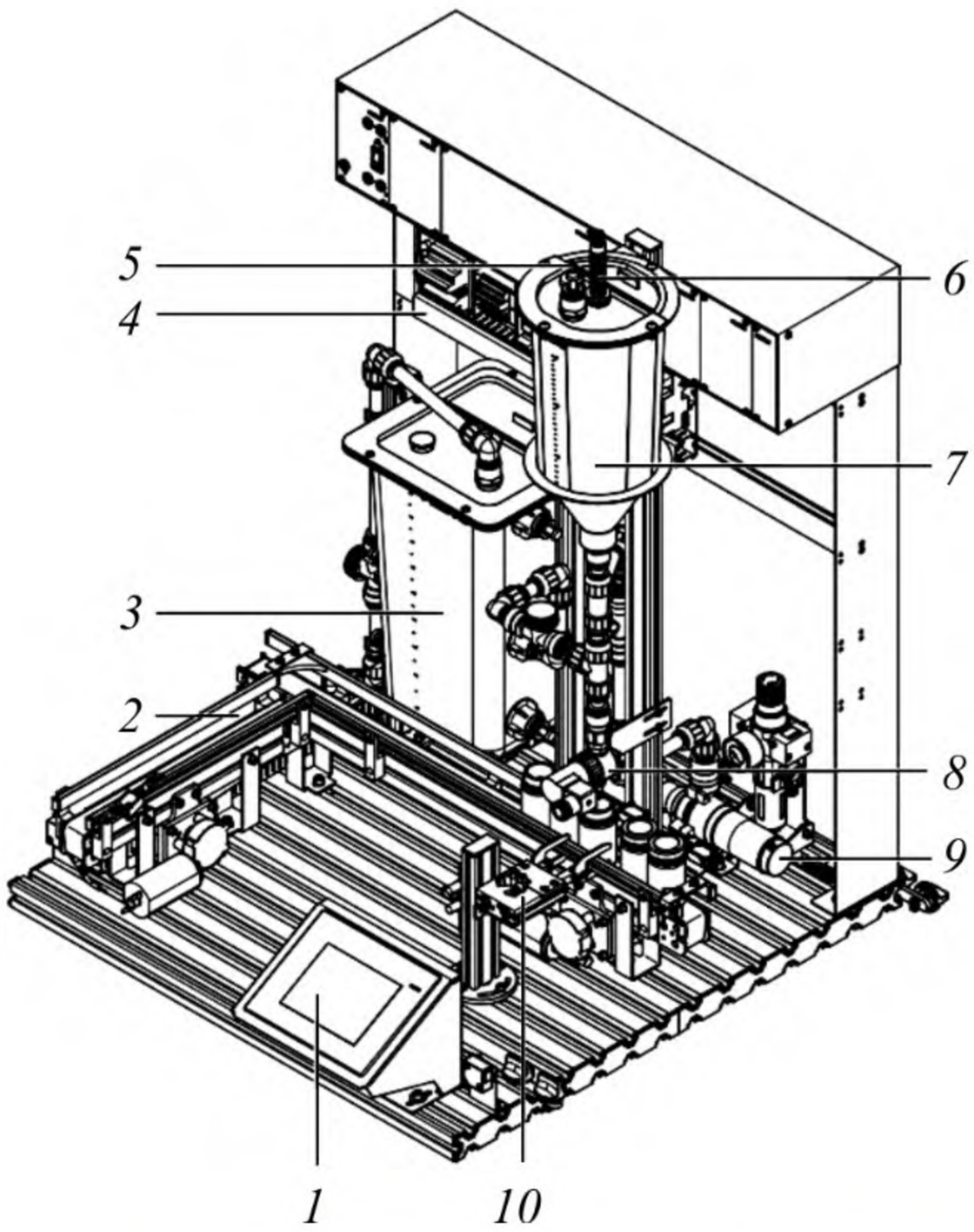


Рис. 3.1. Установка Festo «Станция розлива»: *1* – ЖК дисплей; *2* – конвейер; *3* – емкость B401; *4* – клеммная колодка; *5* – поплавковый контактор 4B10; *6* – ультразвуковой уровнемер 4B1; *7* – емкость B402; *8* – пневматический клапан V403; *9* – насос P401; *10* – пневматический механизм 4M4

Структурная схема установки Festo «Станция розлива» представлена на рис. 3.2.

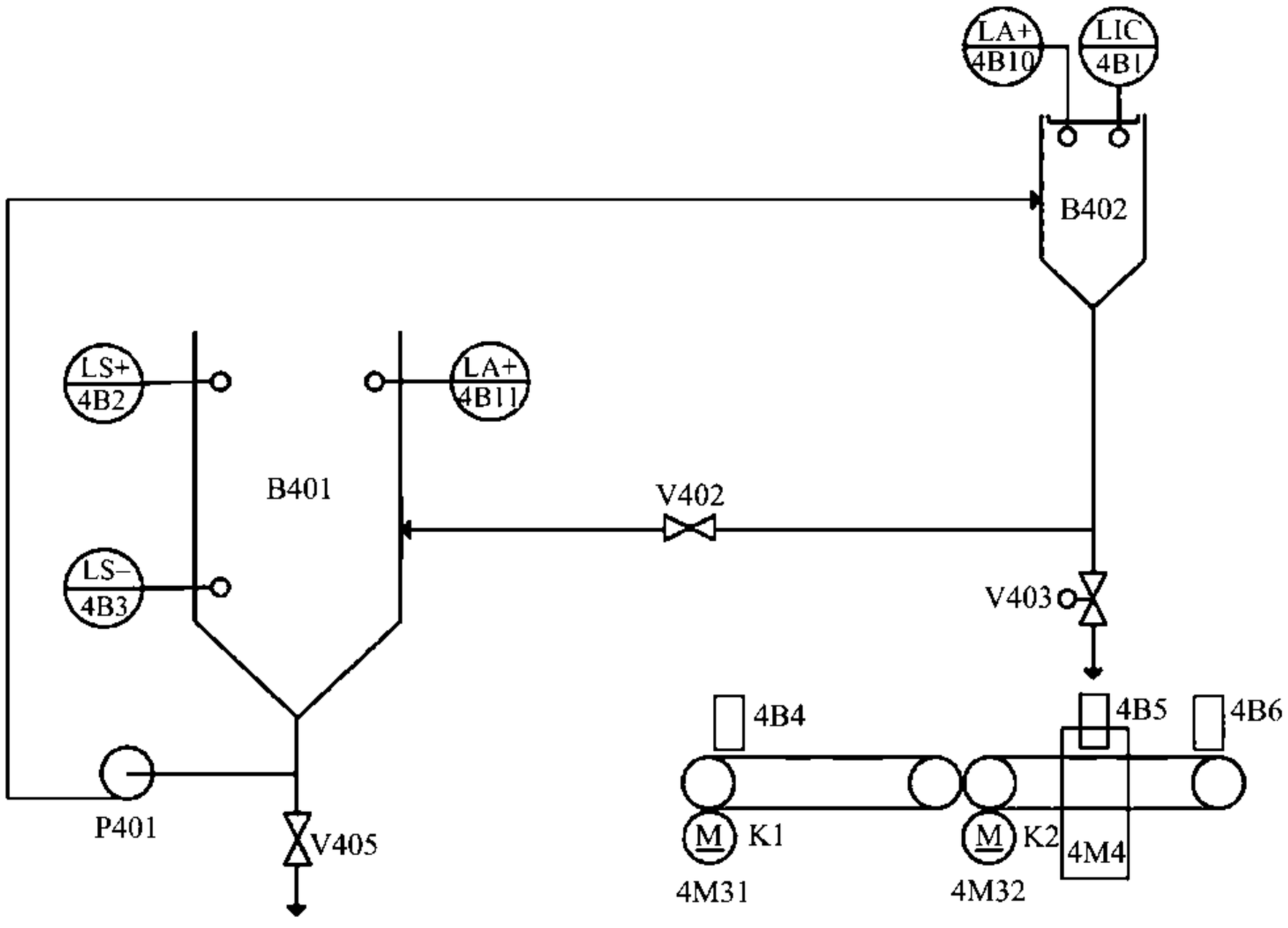


Рис. 3.2. Структурная схема лабораторного стенда

При помощи электронасоса центробежного действия Р401 жидкость из резервуара В401 перекачивается в резервуар В402 до уровня, который определяется размером дозы в одну емкость и количеством дозируемых емкостей. Измерение уровня в резервуаре В402 осуществляется при помощи ультразвукового уровнемера 4В1. Также в резервуаре В402 находится поплавковый контактор 4В10, работающий в качестве блокировки по превышению верхней критической отметки уровня в резервуаре (блокировка реализуется отключением насоса Р401). Для сигнализации превышения уровня в резервуаре В401 по нижнему и верхнему пределам установлены емкостные уровнемеры 4В3 и 4В2 соответственно. Также в резервуаре В401 находится поплавковый контактор 4В11, работающий в качестве блокировки по превышению верхней критической отметки уровня в резервуаре.

В составе установки имеются два конвейера К1 и К2 с нереверсивными электроприводами постоянного напряжения 4М31 и 4М32 соответственно.

В начале конвейера К1 имеется датчик положения 4В4. Если датчик 4В4 зафиксировал емкость, автоматически осуществляется

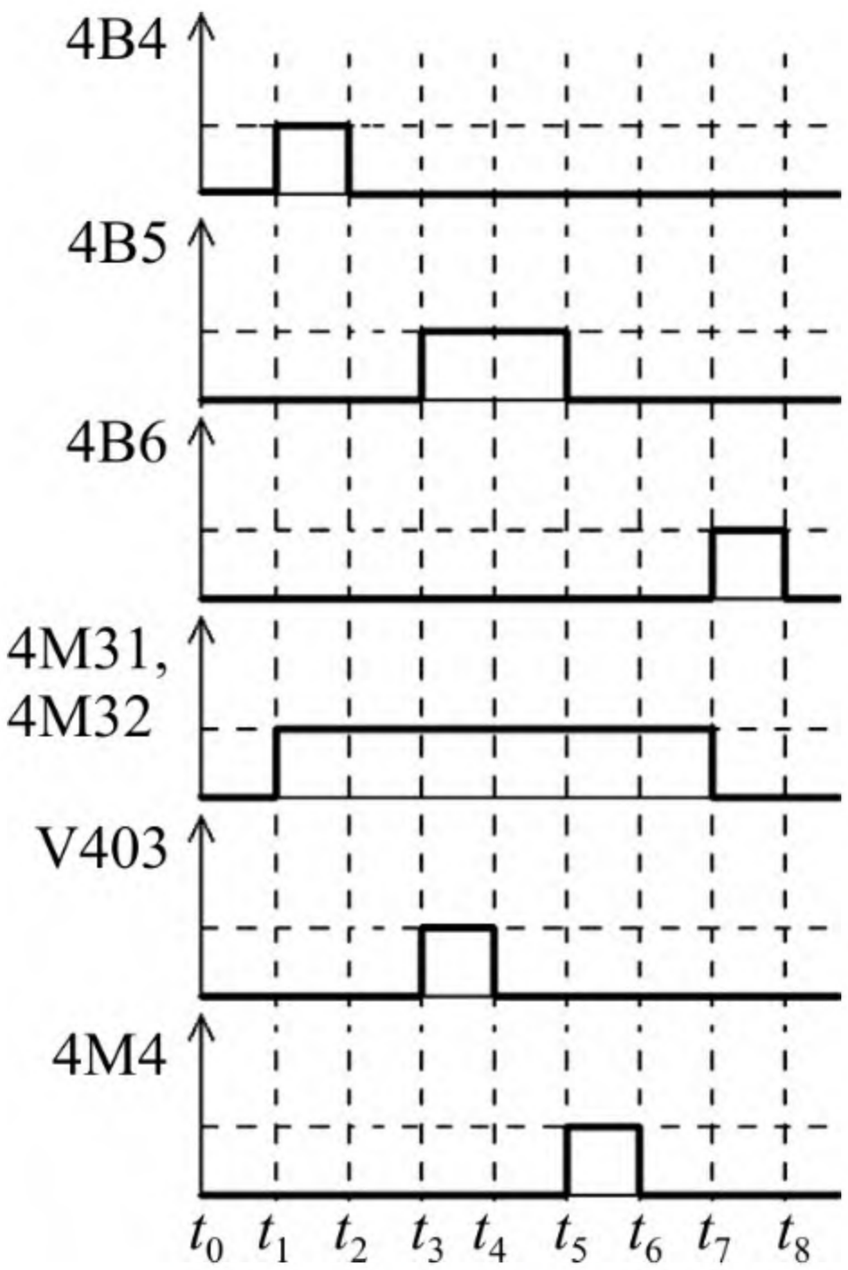


Рис. 3.3. Временная диаграмма управления «Станцией розлива»

запуск конвейеров К1 и К2. При прохождении емкости мимо датчика положения 4В5 срабатывает пневматический механизм 4М4, фиксирующий емкость, и при помощи пневматического клапана V403 происходит дозирование жидкости из резервуара В402 в емкость. По завершении дозировки пневматический механизм 4М4 переходит в исходное состояние и емкость продолжает движение по конвейеру К2 до датчика положения 4В6. Пока датчик 4В6 будет фиксировать емкость, дозировка в следующую емкость осуществлена не будет.

Временная диаграмма описанного выше процесса управления представлена на рис. 3.3. В табл. 3.1 представлен полный перечень адресов имен переменных.

Таблица 3.1

Общая таблица символьных имен установки «Станция розлива»

Symbol	Address	Comment
4A1	I 124.0	Сигнал с компаратора 4A1
4B2	I 124.1	Уровнемер 4B2
4B3	I 124.2	Уровнемер 4B3
4B4	I 124.3	Сенсор 4B4
4B5	I 124.4	Сенсор 4B5
4B6	I 124.5	Сенсор 4B6
P401	Q 124.0	Дискретный сигнал на насос P401
V403	Q 124.1	Клапан налива V 403
4M31, 4M32	Q 124.2	Конвейер 4M31, 4M32
4M4	Q 124.3	Пневматический переключатель 4M4
4B10	PIW 752	Уровнемер 4B10
P401	PQW 752	Аналоговый сигнал на насос P401
4B10	–	Реле 4K10 в обход контроллера
4B11	–	Реле 4K10 в обход контроллера

3.2. Алгоритмизация работы установки

В процессе алгоритмизации работы установки используются только дискретные сигналы, поэтому таблица символов будет иметь вид, представленный в табл. 3.2. Таблица символов необходима для того, чтобы объявить все используемые блоки, входы, выходы и т.д. и присвоить им соответствующие адреса и типы данных.

Таблица 3.2

Значения входов и выходов

№	Название	Адрес	Тип	Примечание
1	4A1	I 124.0	BOOL	Сигнал с компаратора 4A1
2	key	I 124.3	BOOL	Сенсор 4B4
3	key2	I 124.4	BOOL	Сенсор 4B5
4	key3	I 124.5	BOOL	Сенсор 4B6
5	block	M 0.0	BOOL	Внутренняя переменная
6	T1	T 1	TIMER	Внутренняя переменная
7	T2	T 2	TIMER	Внутренняя переменная
8	T3	T 3	TIMER	Внутренняя переменная
9	DP401	Q 124.0	BOOL	Дискретный сигнал на насос P 401
10	V403	Q 124.1	BOOL	Дозирующий клапан V 403
11	M1	Q 124.2	BOOL	Привод конвейера 4M31,4M32
12	4M4	Q 124.3	BOOL	Пневматический переключатель 4M4

После загрузки в контроллер конфигурации можно начинать разрабатывать программу пользователя для контроллера. Основным программным модулем является организационный блок OB1.

Программа, созданная на языке лестничных диаграмм, представлена на рис. 3.4, 3.5 и 3.7. Первый модуль программы, представленный на рис. 3.4, реализует заполнение резервуара B402 до значения, установленного при помощи компаратора 4A1.

Схема запуска и останова конвейера (см. рис. 3.5) реализована с использованием SR-триггера. Когда на вход SET триггера поступает сигнал «1» от датчика положения 4B4 с адресом I 124.3, установленного в самом начале конвейера, нереверсивные электродвигатели 4M31 и 4M32 с адресом Q 124.2 запускаются и могут остановиться только в том случае, если на вход RESET триггера поступает

сигнал «1» от последнего датчика положения 4В6 с адресом I 124.5, установленного на конце конвейера.

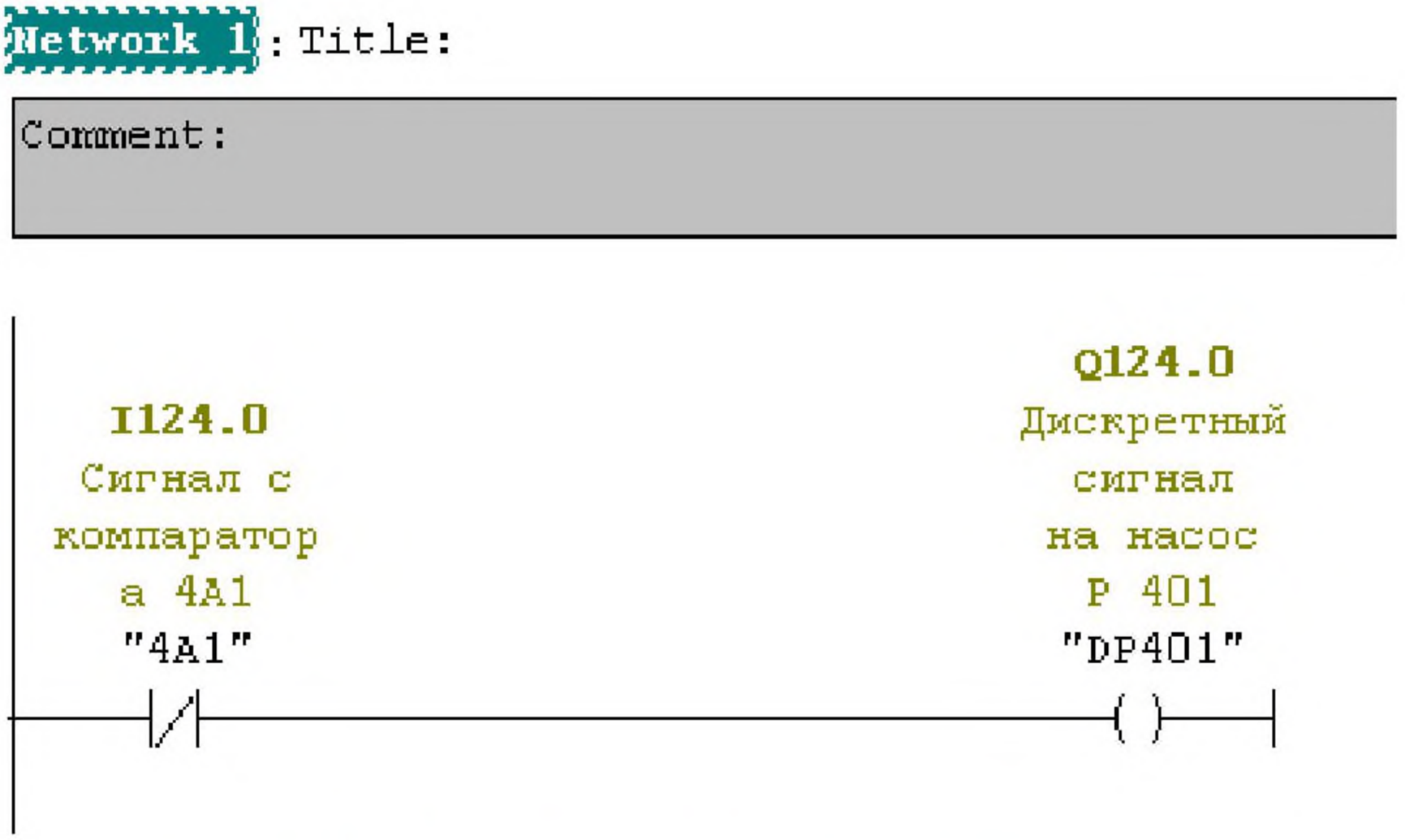


Рис. 3.4. Программа заполнения резервуара

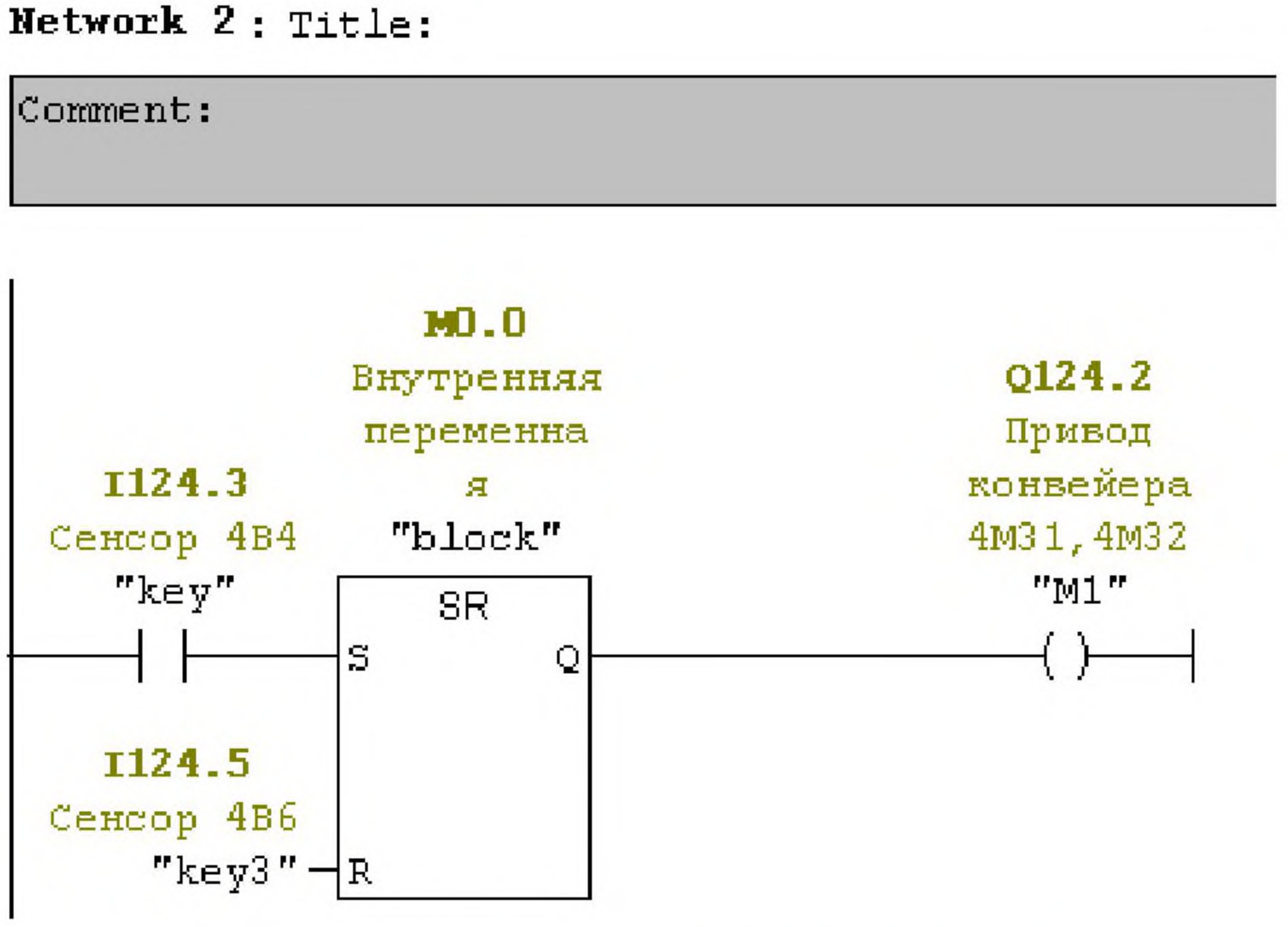


Рис. 3.5. Программа запуска и останова конвейера

Процесс дозирования жидкости по емкостям реализован с использованием таймеров Т1 и Т2 и блока запаздывания Т3. Описание работы таймеров приведено в табл. 3.3, временная диаграмма их работы представлена на рис. 3.6.

Таблица 3.3

Описание работы таймеров

Таймер	Название	Описание
SP_PULSE	Импульс	Максимальное время, в течение которого выходной сигнал остается равным 1, совпадает с заданным временем *t*. Выход сбрасывается раньше, если входной сигнал меняется на 0
SE_PEXT	Импульс с памятью	Выходной сигнал остается равным 1 в течение заданного времени независимо от того, как долго остается равным 1 входной сигнал
SD.ODT	Задержка включения	Выходной сигнал устанавливается в 1 только по истечении заданного времени, при этом входной сигнал все еще должен быть равен 1
SS_ODTS	Задержка включения с памятью	Выходной сигнал устанавливается в 1 только по истечении заданного времени независимо от того, как долго остается равным 1 входной сигнал
SF_OFFDT	Задержка выключения	Выходной сигнал устанавливается в 1, когда устанавливается в 1 входной сигнал, и остается равным 1, пока таймер работает. Отсчет времени начинается когда входной сигнал меняется с 1 на 0

Программа реализации процесса дозирования жидкости (см. рис. 3.7) реализована следующим образом. В тот момент, когда емкость оказывается у второго датчика положения 4B5 с адресом I 124.4, срабатывает таймер Т1 (S_Pulse) и подается сигнал на открытие клапана V403 с адресом Q 124.1, через который жидкость начинает наливаться в емкость в течение времени, указанного в таймере. Через некоторое время запускается таймер Т2, который отвечает за срабатывание пневматической задвижки 4M4 с адресом Q 124.3.

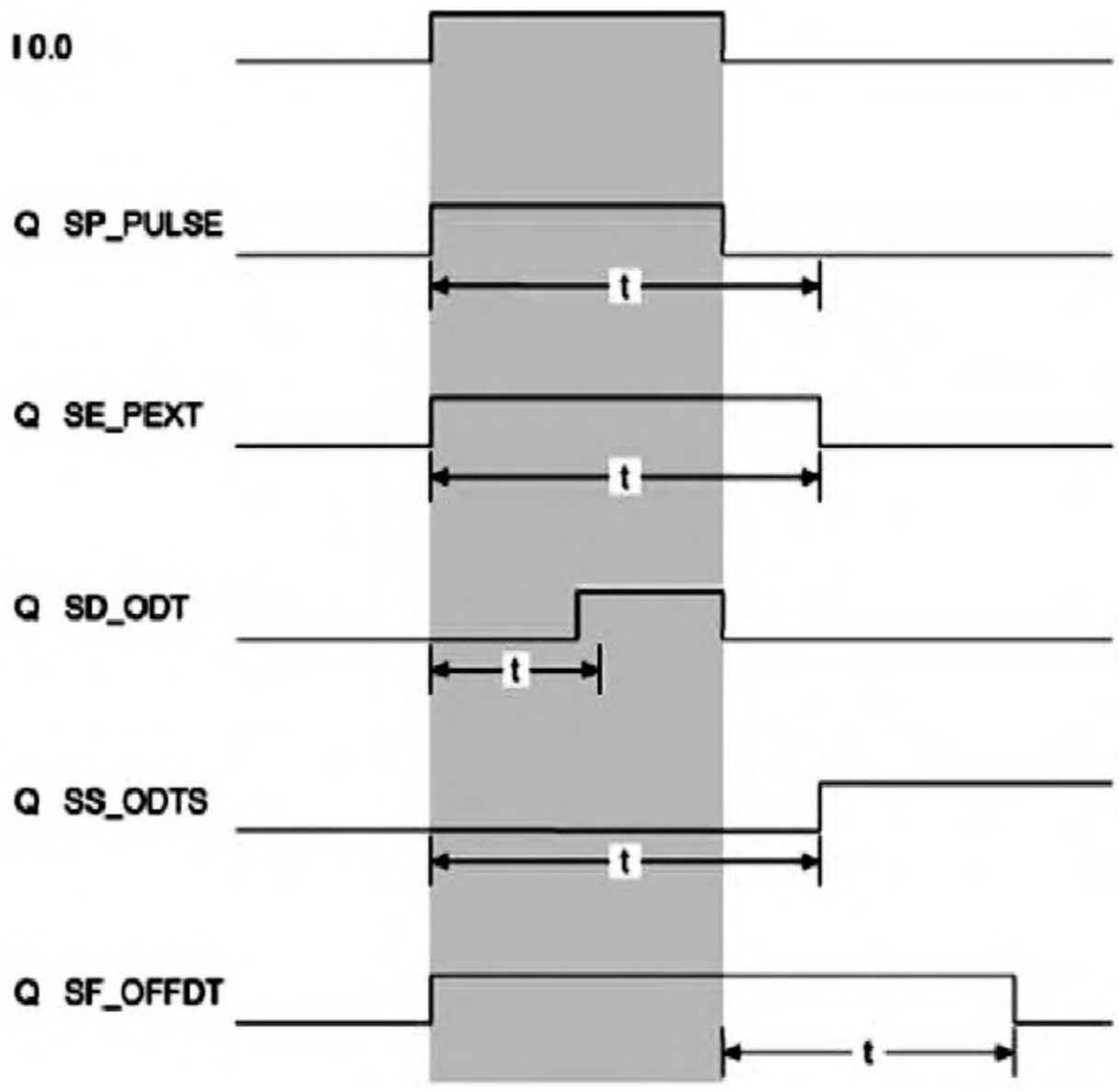


Рис. 3.6. Временная диаграмма таймеров

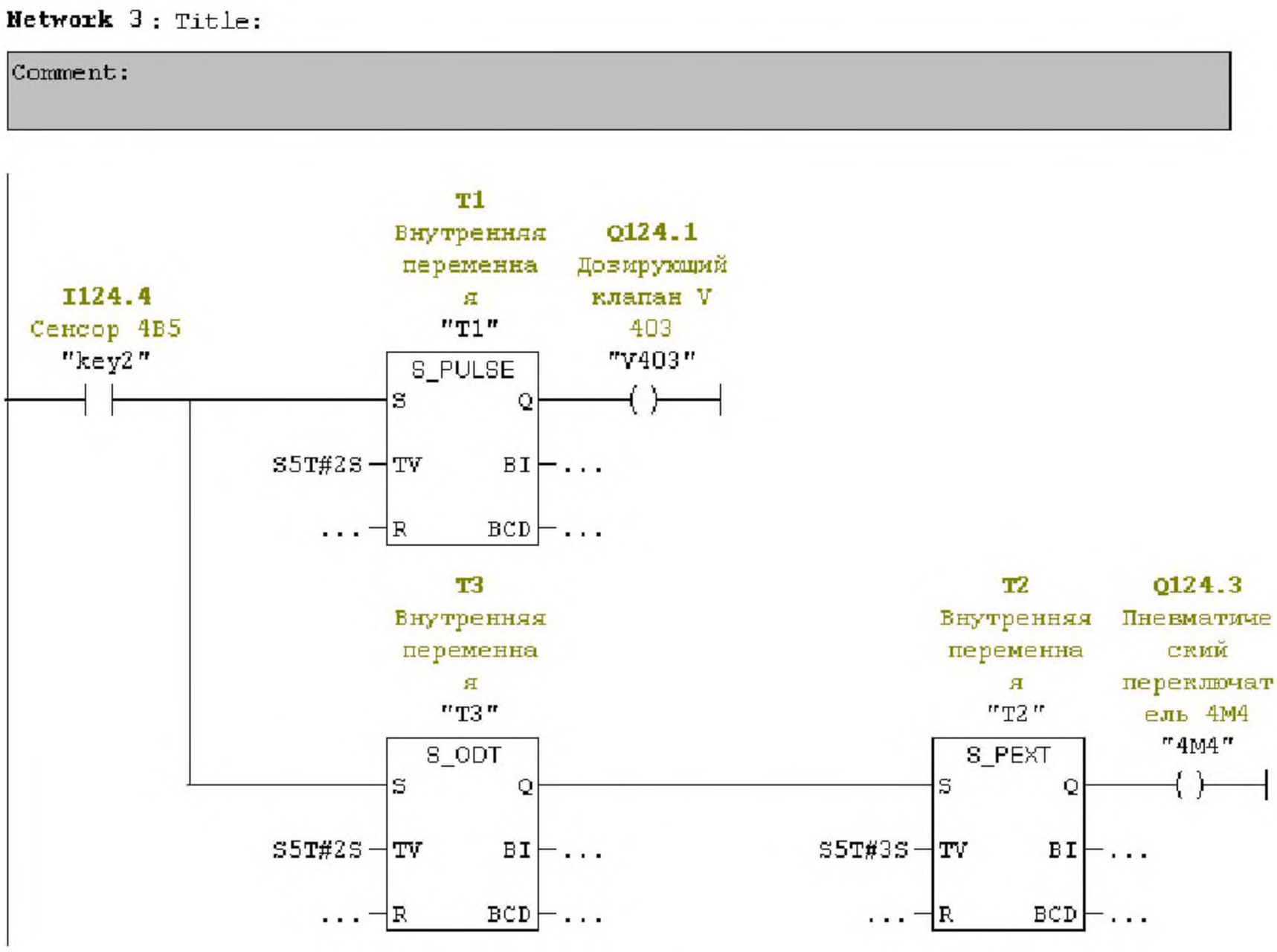


Рис. 3.7. Программа реализации процесса дозирования жидкости

Глава 4. АЛГОРИТМИЗАЦИЯ СИСТЕМЫ ПРОГРАММНО-ЛОГИЧЕСКОГО УПРАВЛЕНИЯ КОМПАКТНОЙ СТАНЦИЕЙ

4.1. Описание установки

Внешний вид установки Festo «Компактная станция» представлен на рис. 4.1.

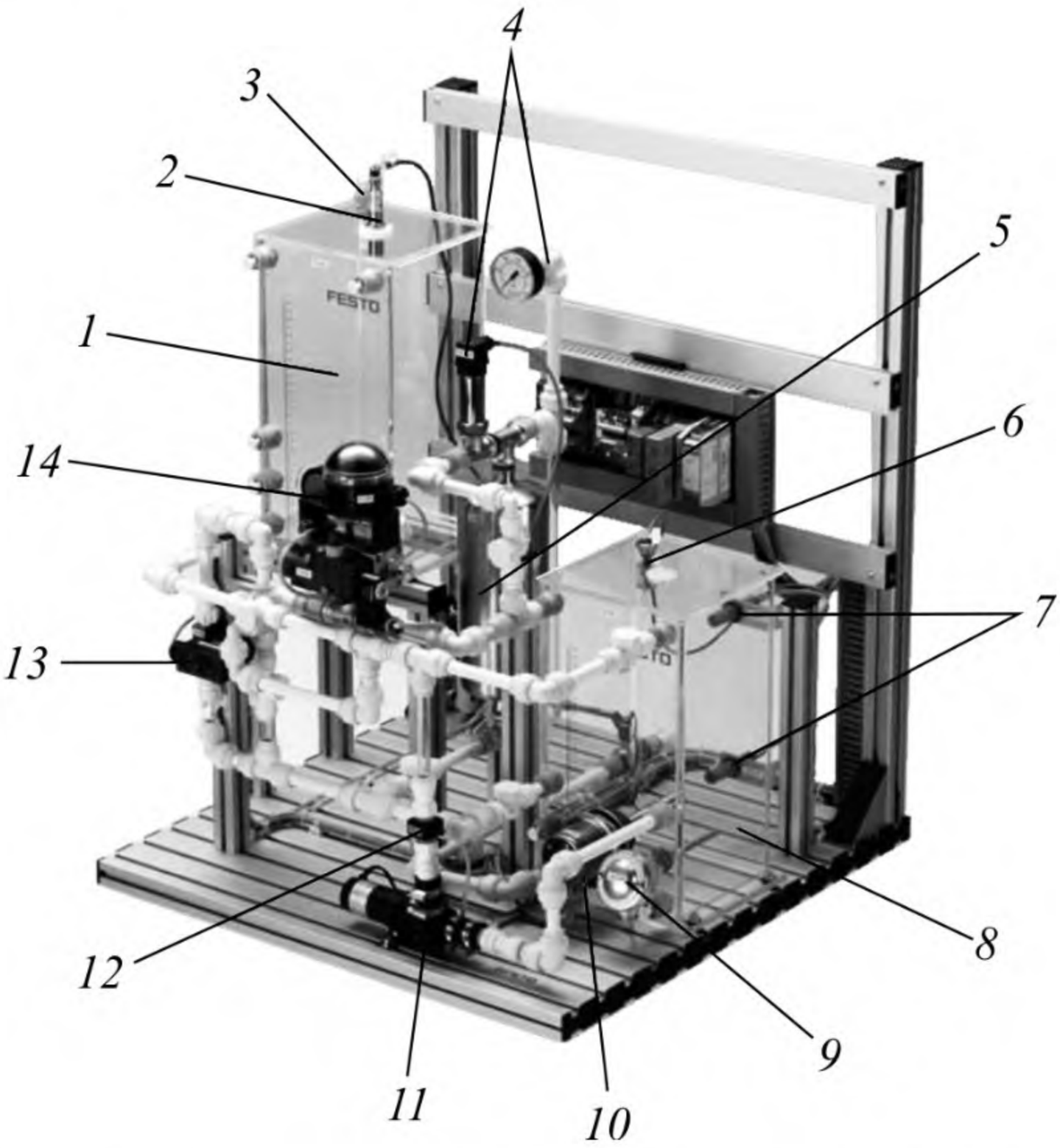


Рис. 4.1. Лабораторный стенд Festo «Компактная станция»: *1* – емкость B102; *2* – ультразвуковой уровнемер B101; *3* – поплавковый контактор S112; *4* – датчик давления B103; *5* – резервуар давления D103; *6* – поплавковый контактор S111; *7* – емкостные уровнемеры B113 и B114; *8* – емкость B101; *9* – термометр сопротивления B104; *10* – нагреватель E104; *11* – насос P101; *12* – датчик расхода B102; *13* – пропорциональный клапан V106; *14* – пневмоклапан V102

Структурная схема установки Festo «Компактная станция» представлена на рис. 4.2.

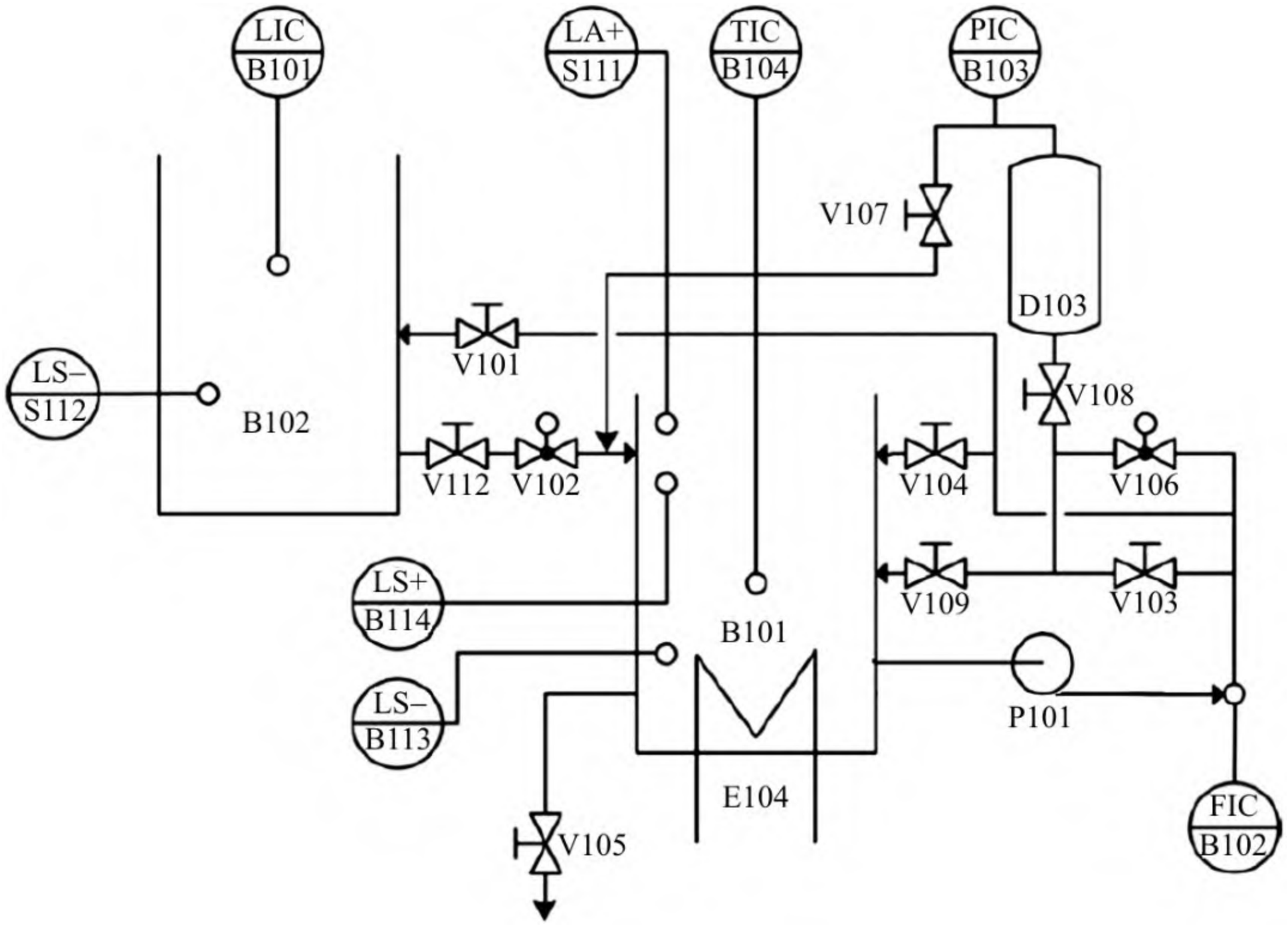


Рис. 4.2. Структурная схема лабораторного стенда

Первый контур управления, реализованный на базе лабораторного стенда Festo «Компактная станция», – управление уровнем жидкости в резервуаре B102. При помощи электронасоса центробежного действия P101 жидкость из резервуара B101 перекачивается в резервуар B102. Уровень жидкости в резервуаре B102 измеряется при помощи ультразвукового уровнемера B101. Также в резервуаре B102 имеется поплавковый контактор S112, предназначенный для блокировки по верхнему уровню. Слив жидкости из резервуара B102 в резервуар B101 может быть реализован при помощи пневмоклапана V102. Сигнализацию по нижнему и верхнему уровням в емкости B101 можно реализовать при помощи емкостных уровнемеров B113 и B114 соответственно. Также в резервуаре B101 имеется поплавковый контактор S111, предназначенный для блокировки по верхнему уровню.

Второй контур – управление температурой в емкости B101 – реализован при помощи электронагревателя E104 и термометра сопротивления B104 с градуировкой Pt100.

Третий контур – управление расходом – реализован в байпасной линии резервуара B101. Измерение расхода реализовано при помощи расходомера B102, управляющее воздействие осуществляется центробежным насосом P101 и пропорциональным клапаном V106.

Четвертый контур – управление давлением в резервуаре D103 – реализован при помощи гидравлических линий связи резервуаров D103 и B101. Измерение давления реализовано при помощи датчика давления B103, управляющее воздействие осуществляется центробежным насосом P101.

Таблица символьных имен установки Festo «Компактная станция» представлена в табл. 4.1.

Таблица 4.1

Общая таблица символьных имен установки
«Компактная станция»

Symbol	Address	Comment
PumpStat	I 124.0	Состояние насоса
S111	I 124.1	Уровнемер S 111
S112	I 124.2	Уровнемер S 112
B113	I 124.3	Емкостной уровнемер B 113
B114	I 124.4	Емкостной уровнемер B 114
S 115	I 124.5	Концевой включатель S115 клапана V 102
S 116	I 124.6	Концевой выключатель S116 клапана V 102
Button1	I 125.0	Кнопка Start
Button2	I 125.1	Кнопка Stop
Key	I 125.2	Ключ
Reset	I 125.3	Кнопка Reset
B101	PIW 752	Уровнемер B 101
B102	PIW 754	Расходомер B 102
B103	PIW 756	Датчик давления B 103
B104	PIW 758	Термометр B 104
PumpAI	PQW 752	Аналоговый сигнал насоса P101
V106	PQW 753	Клапан электрический V 106
V102	Q 124.0	Пневматический клапан V 102

Окончание табл. 4.1

Symbol	Address	Comment
E104	Q 124.1	Тэн E104
K1	Q 124.2	Реле переключения режимов работы насоса К 1
P101	Q 124.3	Дискретный сигнал на насос Р 101
Buttonlamp1	Q 125.0	Лампа Start
Buttonlamp2	Q 125.1	Лампа Reset
Q1	Q 125.2	Лампа Q1
Q2	Q 125.3	Лампа Q2

4.2. Алгоритмизация работы установки

4.2.1. Программа аварийной защиты уровня в емкости B102

Примем следующие обозначения: E-1 – рассматриваемая емкость B102, LS – поплавковый сигнализатор уровня S112 (адрес I 124.2), XV1 – пневматический клапан V102 (адрес Q 124.0), NS1 – насос P101 (адрес Q 124.3). Контур регулирования приведен на рис. 4.3.

Рис. 4.3. Контур системы аварийной защиты

При достижении уровня в емкости E-1 критической отметки (при срабатывании датчика LS) на 15 с выключается насос NS1 и открывается клапан XV1, установленный на линии слива воды из емкости. Блок-схема программы представлена на рис. 4.4.

Программа работает следующим образом. При уровне, который не достиг поплавкового датчика LS (сигнал с датчика «0»), необходимо использовать нормально-замкнутый контакт, чтобы на насос NS1 поступил сигнал «1» и началась закачка жидкости в емкость Е-1.

При достижении уровня поплавкового датчика LS, нормально-разомкнутый контакт замыкается, на таймер приходит «1», которая держится 15 с, отключив насос NS1 на 15 с.

При достижении уровня поплавкового датчика LS нормально-разомкнутый контакт замыкается, на таймер приходит «1», которая держится 15 с и открывает клапан XV1 на 15 с.

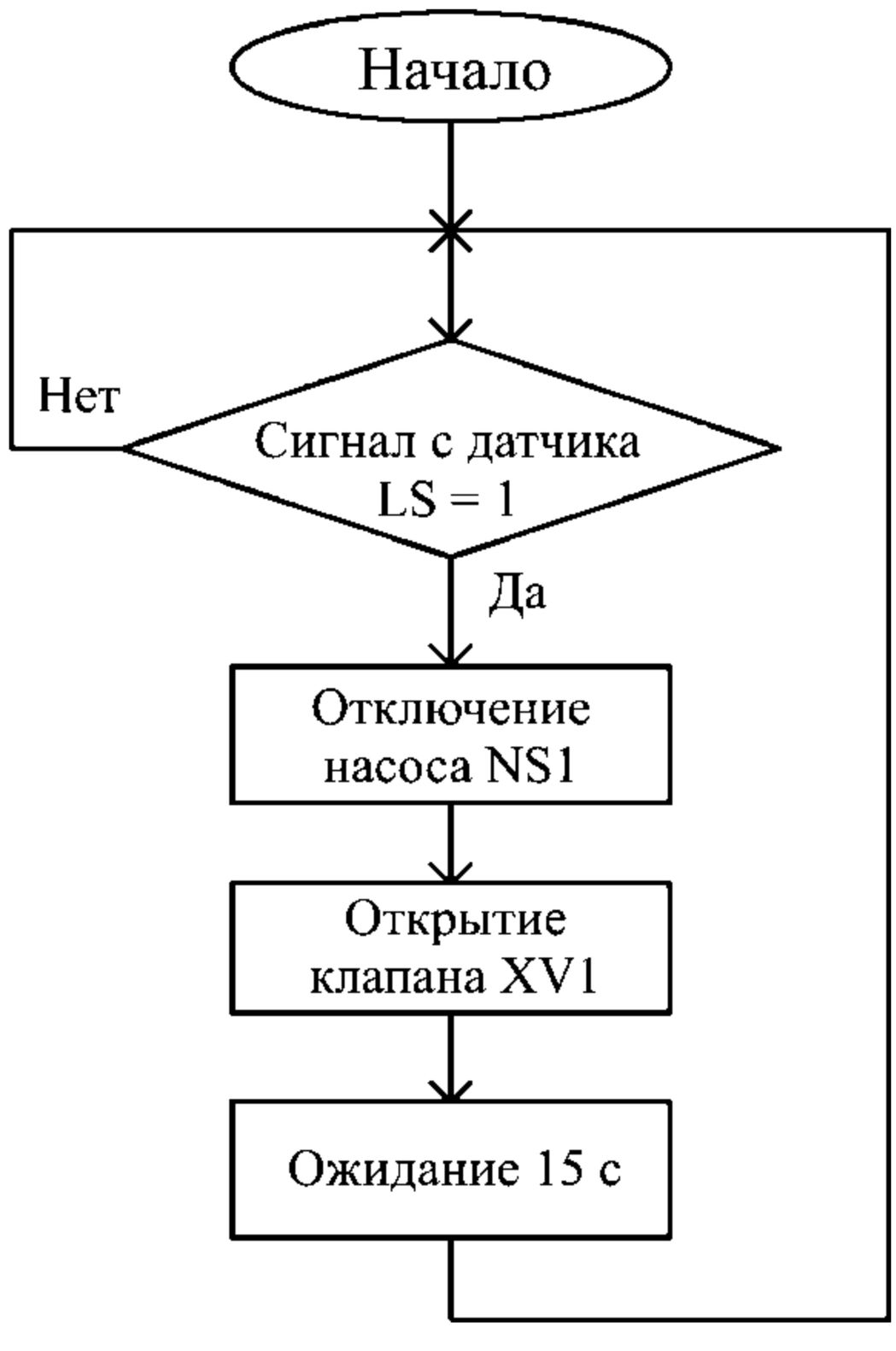


Рис. 4.4. Блок-схема программы
аварийной защиты

Листинг программы приведен на рис. 4.5.

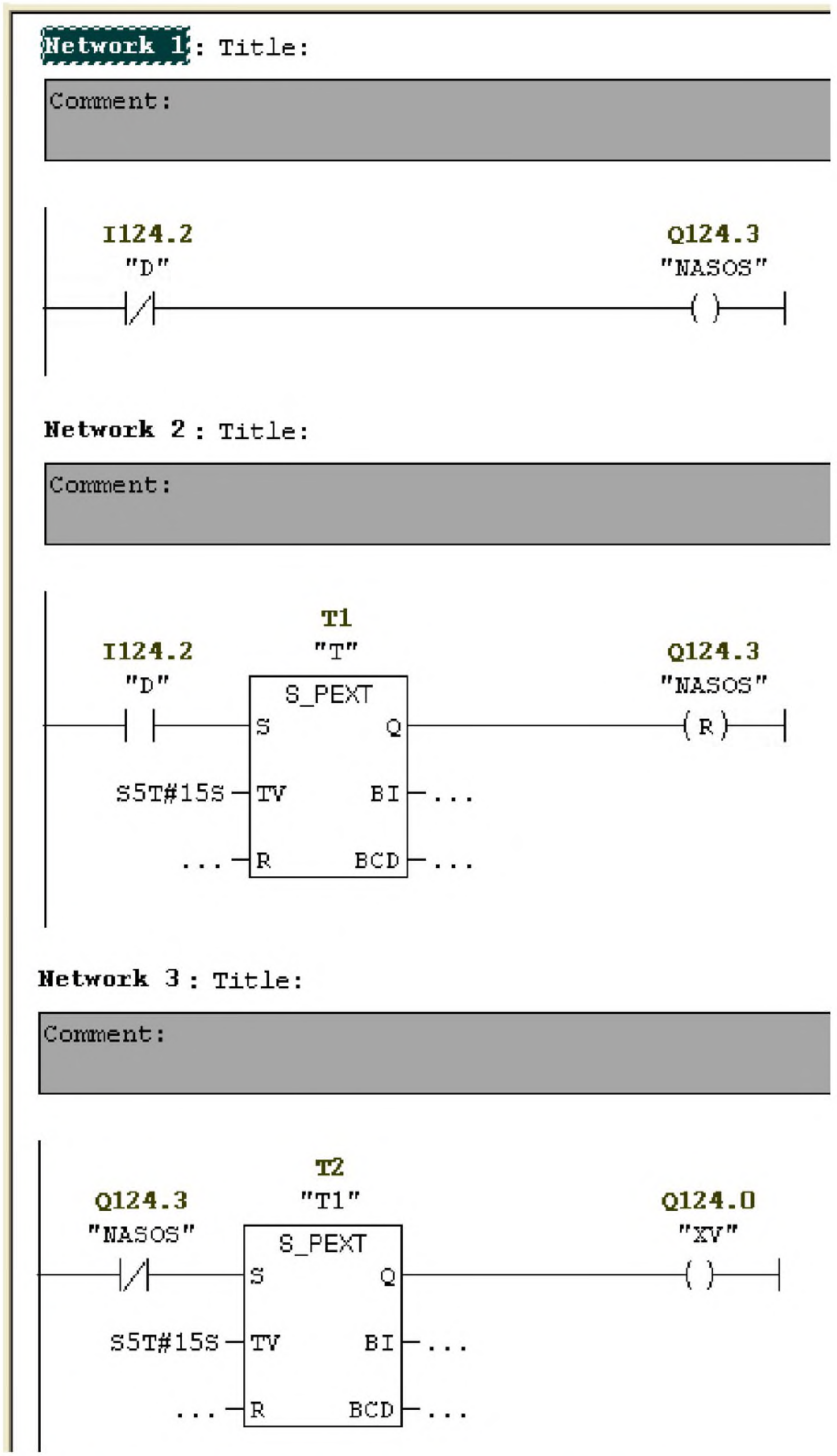


Рис. 4.5. Листинг программы аварийной защиты

4.2.2. Программа позиционного регулирования уровня

Для создания данной программы необходимо отмасштабировать аналоговый сигнал (4–20 мА), который приходит с ультразвукового датчика B101. Исходя из данных датчика, получаем пропорцию, представленную в табл. 4.2 и на рис. 4.6.

Таблица 4.2

Масштабирование аналогового сигнала

Вход аналогового согнала, мА	Код, выдаваемый аналоговым модулем	Уровень на входе датчика
0	W#16#0000	–
4	W#16#159A	50 мм
20	W#16#6C00	300 мм

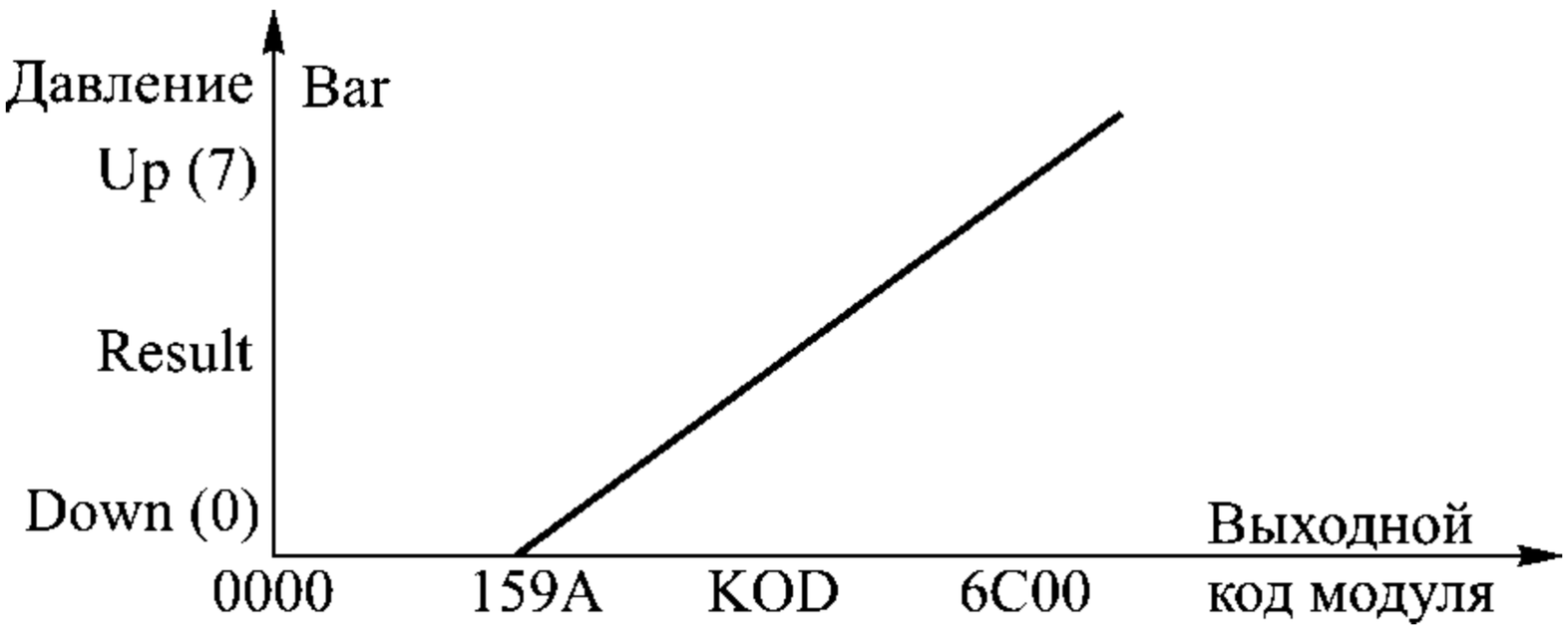


Рис. 4.6. Графическое представление пропорции

Оформим данную задачу как функцию FC с формальными параметрами. В табл. 4.3 представлено, как объявлены переменным функции FC локальные переменные.

Исходя из данной пропорции можно составить формулу:

$$\text{RESULT} = \text{Down} + \frac{(\text{KOD} - \text{W\#16\#159A})}{\text{W\#16\#5666}} \cdot (\text{Up} - \text{Down}).$$

На рис. 4.7 представлена программа масштабирования аналогового сигнала 4–20 мА, написанная на языке STL (низкоуровненый язык инструкций). Затем в блоке OB1 вызывается функция FC1 (см. рис. 4.8).

Таблица 4.3

Локальные переменные

Decl	Symbol	Data Type	Initial Value	Comment
In	KOD	Int	0	Измеренное значение аналогового входа (в нашем случае) PIW752
In	Up	Real	0,00	Уровень, соответствующий выходному току датчика 20 мА
In	Down	Real	0,00	Уровень, соответствующий выходному току датчика 4 мА
Out	Result	Real	0,00	Результат (вещественное значение измеренного уровня, мм)
Temp	temp	Real	0,00	Временная переменная для хранения промежуточного результата

```
Network 1: Title:
Comment:

      L     #Up                    //загружаем в ACCU1 Up
      L     #Down                  //загружаем в ACCU1 Down, Up в ACCU2
      -R                           //Up-Down
      T     #temp                  //Сохранить в temp разницу Up-Down
      L     PIW  752               //KOD
      L     W#16#159A              //Загрузить в ACCU1 код соот-щий 4 мА
      -I                           //KOD-w#16#159A
      ITD                          //преобразовать рез-т в длинное целое
      DTR                          //преобразовать результат в вещественное число
      L     #temp                  //загрузить в аккумулятор Up-Down
      *R                           //(Up-Down)*(KOD-w#16#159A)
      L     W#16#5666              //загрузить в ACCU1 (w#16#6C00-w#16#159A)
      ITD                          //преобразовать рез-т в длинное целое
      DTR                          //преобразовать результат в вещественное число
      /R                           //разделить ACCU2 на (w#16#6C00-w#16#159A)
      L     #Down                  //загрузить в ACCU1 Down
      +R                           //сложить аккумуляторы
      T     #Result                //записать результат в переменную Result
```

Рис. 4.7. Программный код масштабирования аналогового сигнала

```
Network 1: Title:
Comment:

CALL  FC     1
 Up     :=3.000000e+002
 Down   :=5.000000e+001
 KOD    :=PIW752
 Result:=MD1
```

Рис 4.8. Вызов FC1 в OB1

На рис. 4.8 и 4.9 представлена программа позиционного регулятора с уставкой 100 мм и гистерезисом 10 мм (написана на языке LAD).

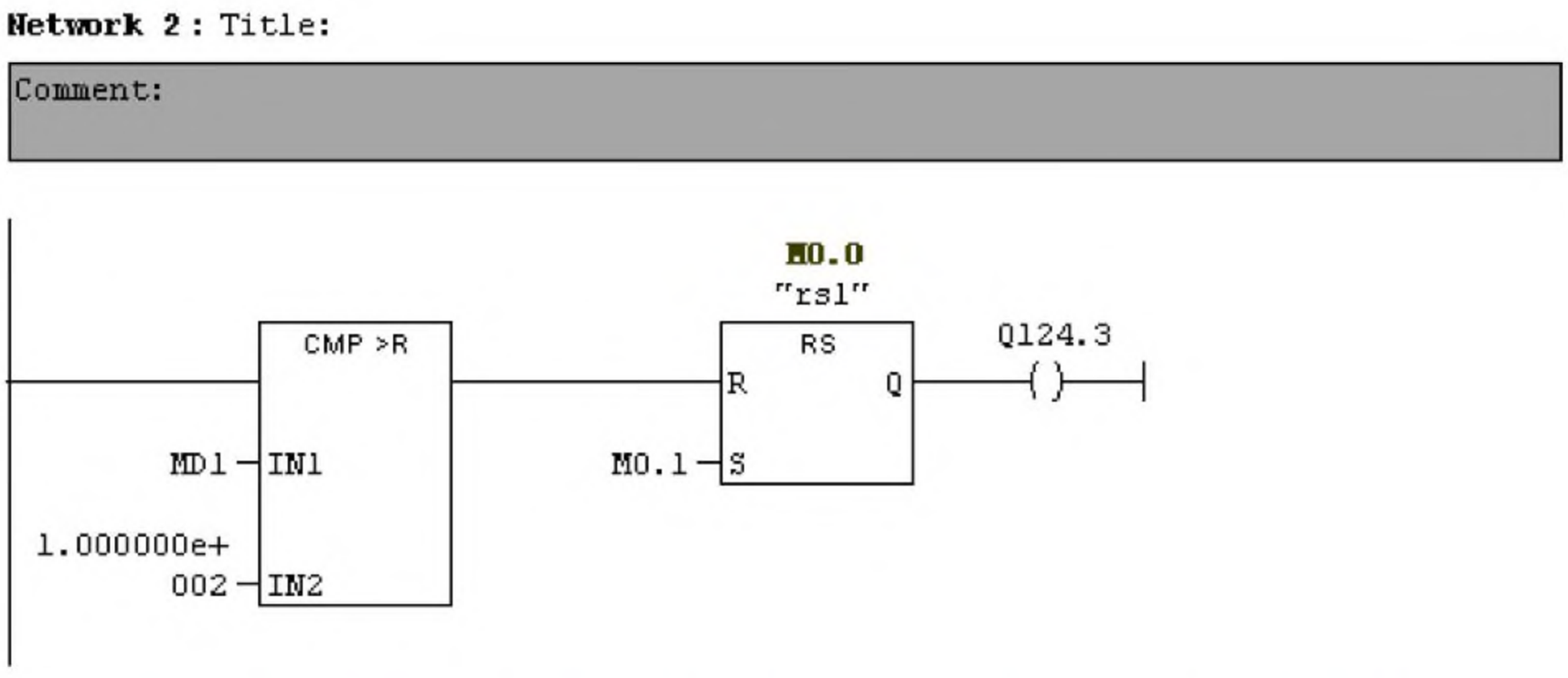


Рис. 4.8. Программный код позиционного регулятора

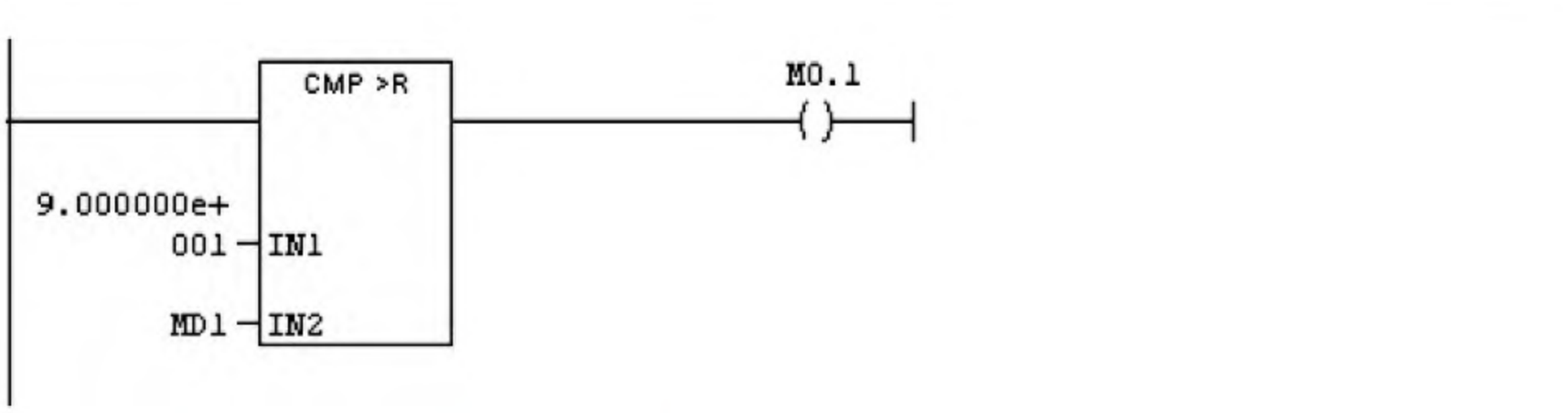


Рис. 4.9. Программный код позиционного регулятора

Network 1: если сигнал с датчика B101 (PIW752), отмасштабированный и записанный в меркер MD1, больше чем 100, то на вход *R* RS-триггера приходит «1», на катушке Q124.3 – «0», следовательно, насос NS1 перейдет в режим «останов».

Network 2: вторая часть программы служит для того, чтобы запустить насос, когда уровень ниже 90. Если 90 больше, чем сигнал с датчика B101 (PIW752), отмасштабированный и записанный в меркер MD1, то на катушку приходит «1», меркер M0.1 взводится в «1», этот сигнал идет на вход *S* RS-триггера, на выходе на катушке Q124.3 – «1», следовательно, насос перейдет в режим «пуск».

4.2.3. Реализация ПИД-регулятора

В данной программе используется такое же масштабирование аналогового сигнала с ультразвукового датчика B101, как и в программе позиционного регулирования уровня. Сигнал записывается в меркер MD0 = PV (переменная).

Рассмотрим дискретное представление закона работы непрерывного ПИД-регулятора.

Уравнение имеет вид

$$\mu(t) = k_{\text{п}}\left[\varepsilon(t) + \frac{1}{T_u}\int_0^t \varepsilon(t)\,dt + T_{\text{д}}\frac{d\varepsilon(t)}{dt}\right].$$

Если время подвергнуть процедуре квантования, а ошибкой округления в аналого-цифровом преобразователе (АЦП) пренебречь, то исходное уравнение регулятора можно преобразовать в разностное.

Рассмотрим случай, когда производные вычисляются через первую разность, а интегрирование реализуется методом прямоугольников. В этом случае уравнение примет вид

$$\mu[kT] = k_{\text{п}}\left[\varepsilon[kT] + \frac{T}{T_u}\sum_{i=0}^{k}\varepsilon\left[(i-1)T\right] + T_{\text{д}}\frac{\varepsilon[kT] - \varepsilon\left[(k-1)T\right]}{T}\right].$$

Такой алгоритм реализации ПИД-регулятора называют *позиционным*, или *нерекурсивным*. В любой момент времени положение ИМ (позиция) должно соответствовать числу на выходе цифрового фильтра, а в памяти регулятора запоминаются все предыдущие значения сигнала рассогласования. Так как цифровой регулятор – это устройство реального времени (real time), то при программной реализации алгоритма могут возникнуть явления, связанные с переполнением разрядной сетки («залипание»). В связи с этим алгоритмы реальных контроллеров строятся по другой схеме.

Рассмотрим значения выходного сигнала регулятора в предыдущий момент времени:

$$\mu_{k-1} = k_{\text{п}}\left[\varepsilon_{k-1} + \frac{T}{T_u}\sum_{i=0}^{k-1}\varepsilon\left[(i-1)T\right] + T_{\text{д}}\frac{\varepsilon_{k-1} - \varepsilon_{k-2}}{T}\right],$$

тогда

$$\mu_k - \mu_{k-1} = k_{\text{п}}\left(\varepsilon_k - \varepsilon_{k-1}\right) + k_{\text{п}}\frac{T}{T_u}\varepsilon_{k-1} + k_{\text{п}}T_{\text{д}}\frac{\varepsilon_k - 2\varepsilon_{k-1} + \varepsilon_{k-2}}{T}$$

или

$$\mu_k - \mu_{k-1} = k_n\left(1 + \frac{T_{Д}}{T}\right)\varepsilon_k + k_{П}\left(-1 + \frac{T}{T_{И}} - \frac{2T_{Д}}{T}\right)\varepsilon_{k-1} + \frac{k_{П}T_{Д}}{T}\varepsilon_{k-2}.$$

Таким образом, получаем *разностный* (скоростной) алгоритм, использующий *рекурсию*, так как на текущем шаге вычисляется приращение к предыдущему.

Моделирование работы ПИД-регулятора реализуется на основе функционального блока (ФБ). Основное отличие функционального блока от функции в том, что в нем присутствуют статические переменные, сохраняющие свое значение между вызовами функционального блока. Хранятся эти значения в специальных экземплярных блоках данных связываемых с функциональным блоком. Создаются такие блоки данных, либо вручную, либо автоматически при попытке вызова ФБ с указанием имени несуществующего блока данных.

В случае ПИД-регулятора использование ФБ обусловлено тем, что необходимо сохранение ошибок на предыдущем и предпредыдущем тактах. При реализации используется алгоритм в приращениях П, И, Д части относительно предыдущего значения выхода регулятора.

На рис. 4.10 представлена интерфейсная часть (перечень используемых переменных, их вид и тип) ФБ ПИД-регулятора.

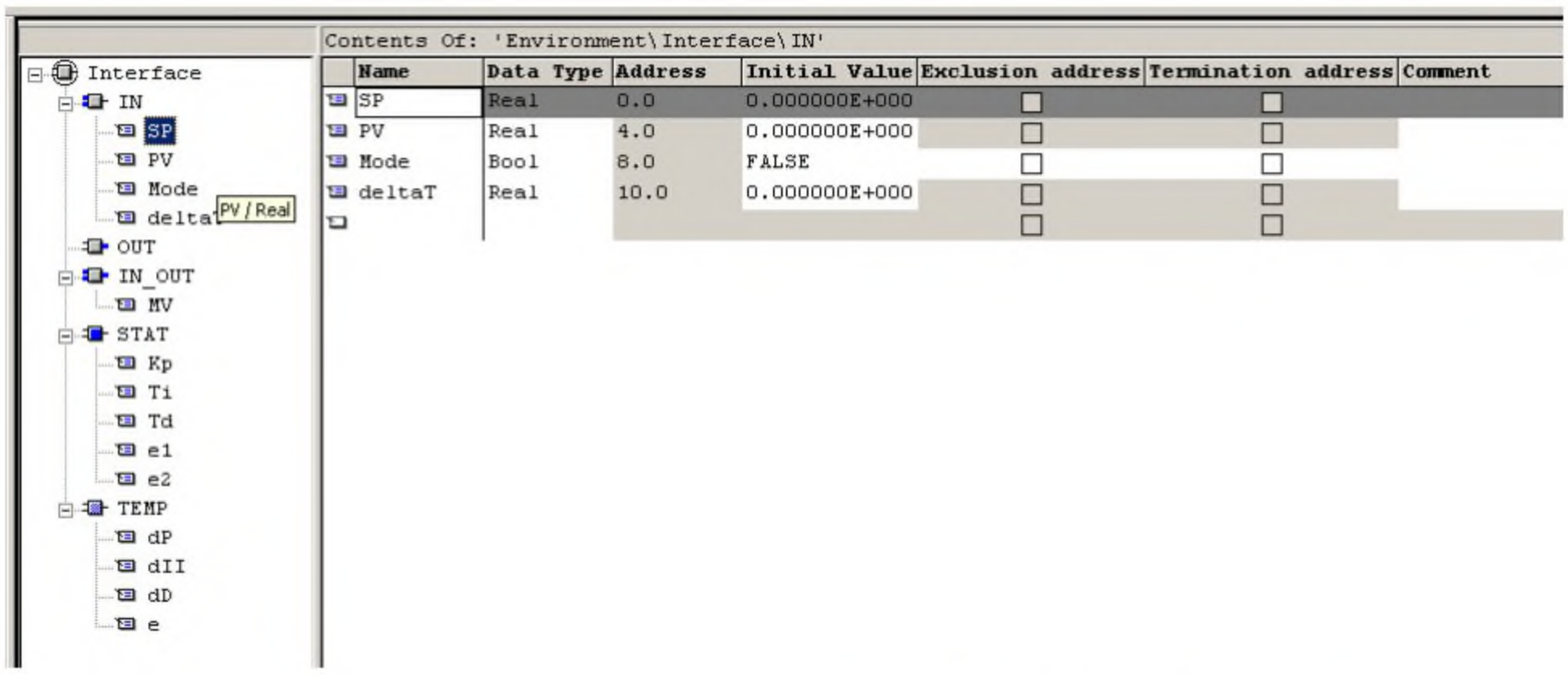


Рис. 4.10. Определение переменных в ФБ ПИД-регулятора

На рис. 4.11 отображен исходный код функционального блока ПИД-регулятора.

```
FB1 : Title:
Comment:
Network 1: Title:
Comment:
      A     #Mode                       //если ручной режим Mode = 0 безусловный переход на метку m1
//если автоматический режим Mode = 1 переход на метку m1 не происходин
      JCN   m1                          //безусловный переход на метку m1

      L     #SP                         //загружаем в ACCU1 SP
      L     #PV                         //загружаем в ACCU1 PV, в ACCU2 SP
      -R                                //ACCU2-ACCU1
      T     #e                          //выгружаем рассогласование e
      L     #e1                         //загружаем рассогласование на пред. такте e1
      -R                                //e-e1
      L     #Kp                         //загружаем Kp
      *R                                //(e-e1)*Kp
      T     #dP                         //выгружаем пропорциональную составляющую

      L     #e                          //загружаем в ACCU1 e
      L     #e1                         //загружаем в ACCU1 e1, в ACCU2 e
      +R                                    //e+e1
      L      2.000000e+000                  //загружаем 2.00
      /R                                    //(e+e1)/2.00
      L      #deltat                        //загружаем deltat
      *R                                    //(e+e1)/2.00
      L      #Kp                            //загружаем Kp
      *R                                    //Kp*(e+e1)/2.00
      L      #Ti                            //загружаем Ti
      /R                                    //(Kp*(e+e1))/(2.00*Ti)
      T      #dII                           //выгружаем интегральную составляющую

      L      #e1                            //загружаем в ACCU1 e1
      L      -2.000000e+000                 //загружаем в ACCU1 -2.00, в ACCU2 e1
      *R                                    //-2.00*e1
      L      #e                             //загружаем e
      +R                                    //-2.00*e1+e
      L      #e2                            //загружаем e2
      +R                                    //-2.00*e1+e+e2
      L      #deltat                        //загружаем deltat
      /R                                    //(-2.00*e1+e+e2)/deltat
      L      #Kp                            //загружаем Kp
      *R                                    //kp*(-2.00*e1+e+e2)/deltat
      L      #Td                            //загружаем Td
      *R                                    //Td*kp*(-2.00*e1+e+e2)/deltat
      T      #dD                            //выгружаем дифференциальную составляющую

      L     #dP                         //загружаем в ACCU1 dP
      L     #dII                        //загружаем в ACCU1 dII, в ACCU2 dP
      +R                                //dP+dII
      L     #dD                         //загружаем в аккумулятор dD
      +R                                //dP+dII+dD
      L     #MV                         //загружаем в аккумулятор MV
      +R                                //dP+dII+dD+MV
      T     #MV                         //выгружаем управляющее воздействие на насос(в %)

m1:   L     #MV                         //загружаем в ACCU1 MV
      L     1.000000e+002               //загружаем в ACCU1 100.00, в ACCU2 MV
      >R                                //если MV>100, RLO=1 вывод MV, если MV<100,RLO=0 переход на метку m2
      JCN   m2                          //безусловный переход на метку m2
      T     #MV                         //вывод MV, если RLO=1
m2:   L     #MV                         //загружаем в ACCU1 MV
      L     0.000000e+000               //загружаем в ACCU1 0.00, в ACCU2 MV
      <R                                //если MV<0.00, RLO=1 вывод MV, если MV>0,RLO=0 переход на метку m3
      JCN   m3                          //безусловный переход на метку m2
      T     #MV                         //вывод MV, если RLO=1
m3:   L     #e1                         //загружаем в аккумулятор e1
      T     #e2                         //присваеваем e1 = e2
      L     #e                          //загружаем в аккумулятор e
      T     #e1                         //присваеваем e = e1
```

Рис. 4.11. Исходный код ПИД-регулятора

Организация вызова функционального блока ПИД-регулятора из блока OB1 изображена на рис. 4.12. При этом, так как указан несуществующий экземплярный блок DB1, будет выдано предложение на его создание (рис. 4.13). При положительном ответе данный блок создается, и его содержимое приведено на рис. 4.14 и, как видно из рисунка, соответствует интерфейсной части блока (без переменных типа TEMP).

```
Network 2: Title:
Comment:

      L     #OB1_PREV_CYCLE
      ITD
      DTR
      L     1.000000e+003
      /R
      T     "dt"                          MD33
      CALL  FB     1 , DB1
       SP    :=1.000000e+002
       PV    :=MD0
       Mode  :=Q124.2
       deltat:="dt"                       MD33
       MV    :=MD10
```

Рис. 4.12. Вызов функционального блока ПИД-регулятора

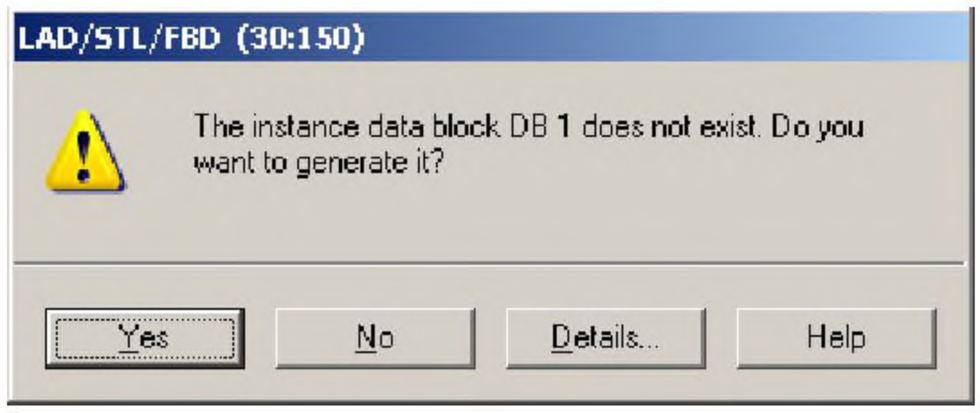


Рис. 4.13. Запрос на создание экземплярного блока данных DB1

DB Param - DB1

Data block Edit PLC Debug View Window Help

DB1 -- PID\SIMATIC 300(1)\CPU 313C

	Address	Declaration	Name	Type	Initial value	Actual value	Comment
1	0.0	in	SP	REAL	0.000000e+...	0.000000e+000	
2	4.0	in	PV	REAL	0.000000e+...	0.000000e+000	
3	8.0	in	Mode	BOOL	FALSE	FALSE	
4	10.0	in	deltat	REAL	0.000000e+...	0.000000e+000	
5	14.0	in_out	MV	REAL	0.000000e+...	0.000000e+000	
6	18.0	stat	Kp	REAL	5.000000e+...	5.000000e+000	
7	22.0	stat	Ti	REAL	2.000000e+...	2.000000e+001	
8	26.0	stat	e1	REAL	0.000000e+...	0.000000e+000	
9	30.0	stat	e2	REAL	0.000000e+...	0.000000e+000	
10	34.0	stat	Td	REAL	0.000000e+...	0.000000e+000	

Рис. 4.14. Содержимое созданного автоматически экземплярного блок данных DB1

На рис. 4.15 изображено перемасштабирование выходного сигнала, поступающего на насос.

Network 3: Title:

Масштабирование выходного сигнала

```
L     MD    10              //загружаем в ACCU1 MD10=MV(%)
L     1.000000e+002         //загружаем в ACCU1 100.00 в ACCU2 MD10
/R                          //MD10/100
L     2.621400e+004         //загружаем в аккумулятор максимальный сигна, который может выдавать насос
*R                          //26214.00*(MD10/100)
RND                         //округляем целого
ITD                         //переводим целое в двойное целое
T     PQW  752              //управляющее воздействие на насос
```

Рис. 4.15. Управляющее воздействие на насос

4.3. Контрольные задания

Составить блок-схему, временную диаграмму и программу на языке LAD, выполняющую, согласно рис. 4.16, следующие действия:

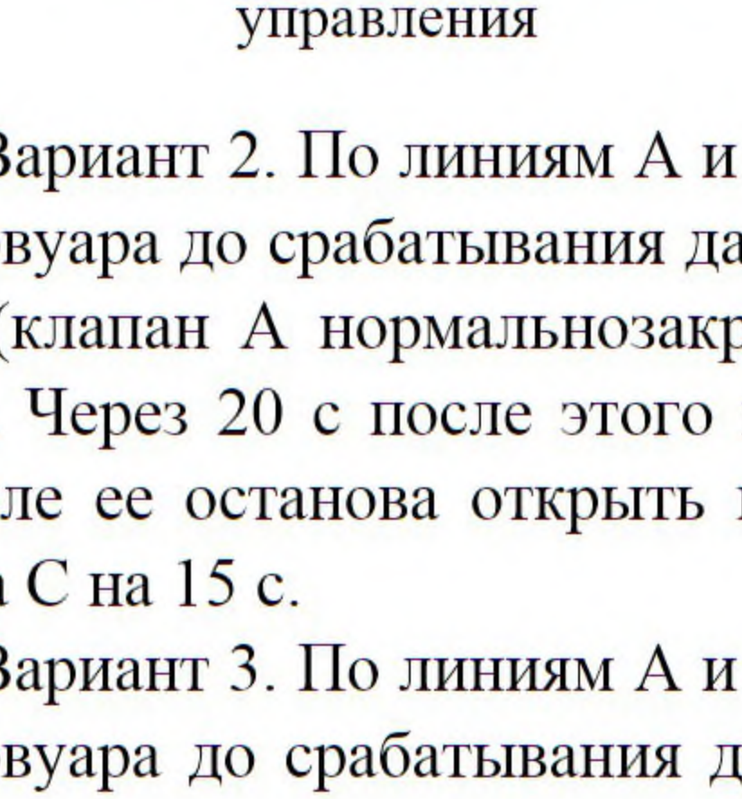


Рис. 4.16. Схема объекта управления

Вариант 1. По линиям А и В текут жидкости. После заполнения резервуара до срабатывания датчика Д закрыть клапаны в линиях А и В (клапан А нормальнооткрытый, клапан В – нормальнозакрытый). Через 10 с после этого включить мешалку М на время 40 с и после ее останова открыть нормальнозакрытый клапан в линии слива С.

Вариант 2. По линиям А и В текут жидкости. После заполнения резервуара до срабатывания датчика Д закрыть клапаны в линиях А и В (клапан А нормальнозакрытый, клапан В – нормальнооткрытый). Через 20 с после этого включить мешалку М на время 30 с и после ее останова открыть нормальнооткрытый клапан в линии слива С на 15 с.

Вариант 3. По линиям А и В текут жидкости. После заполнения резервуара до срабатывания датчика Д закрыть клапан в линии А

и через 10 с в линии В (клапаны А и В нормальнозакрытые). Через 5 с после этого включить мешалку М на время 25 с, и через 10 с после ее останова открыть нормальноотрытый клапан в линии слива С.

Вариант 4. По линиям А и В текут жидкости. После заполнения резервуара до срабатывания датчика Д закрыть клапан в линиях А, и через 5 с в линии В (клапан А нормальнозакрытый, клапан В – нормальнооткрытый). Через 10 с после этого включить мешалку М на время 40 с, и после ее останова открыть нормальнооткрытый клапан в линии слива С на 55 с. После этого заполнить резервуар жидкостями из линий А и В до срабатывания датчика Д.

Глава 5. ОТЛАДКА РАЗРАБОТАННОЙ ПРОГРАММЫ ПОЛЬЗОВАТЕЛЯ

Для отладки программы пользователя существует широкий набор средств. В первую очередь это режим Monitor On/Off (рис. 5.1), для его включения необходимо кликнуть мышью на значок Monitor On/Off.

Рис. 5.1. Вызов режима включения/отключения режима мониторинга

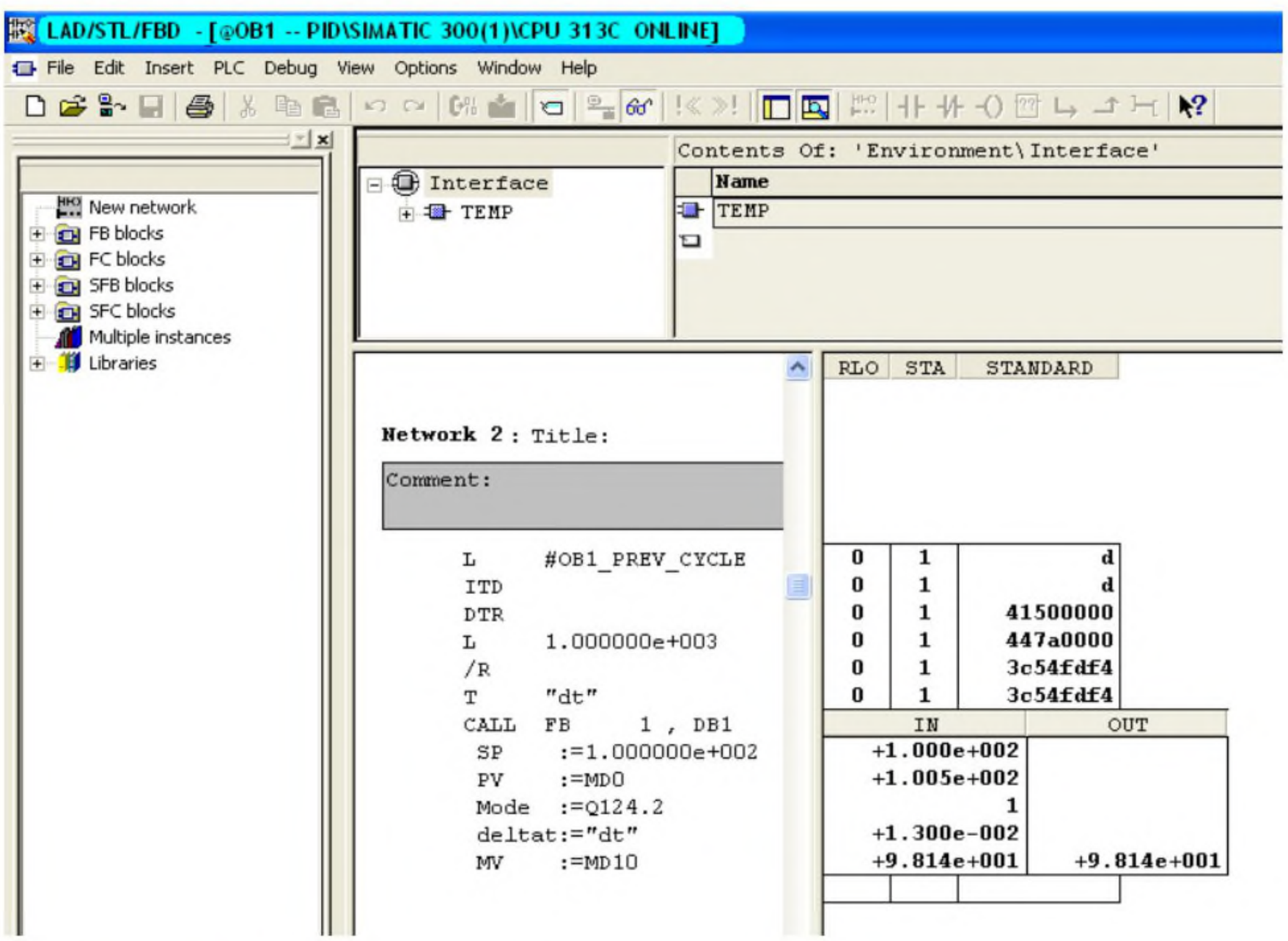


Рис. 5.2. Отладочная информация в режиме мониторинга

В режиме мониторинга (рис. 5.2) слева от кода программы отображается текущее состояние переменных. Выбор формата отображения данных (WORD, INTEGER, REAL и т.д.) можно произвести, кликнув правой кнопкой мыши по этим данным, далее в открывшемся контекстном меню выбрать нужный формат.

Также возможно отслеживание состояния памяти и внесение в нее изменений через инструментарий Monitor/Modify Variables. Внешний вид такого инструментария представлен на рис. 5.3. Для того чтобы принудительно подать на выход платы какой-либо сигнал, необходимо написать его значение в поле Modify value (для дискретных сигналов это 0 или 1, для аналоговых, например, число в формате WORD W#16#AAAA), кликнуть по нему правой кнопкой мыши и выбрать в открывшемся контекстном меню пункт Modify value.

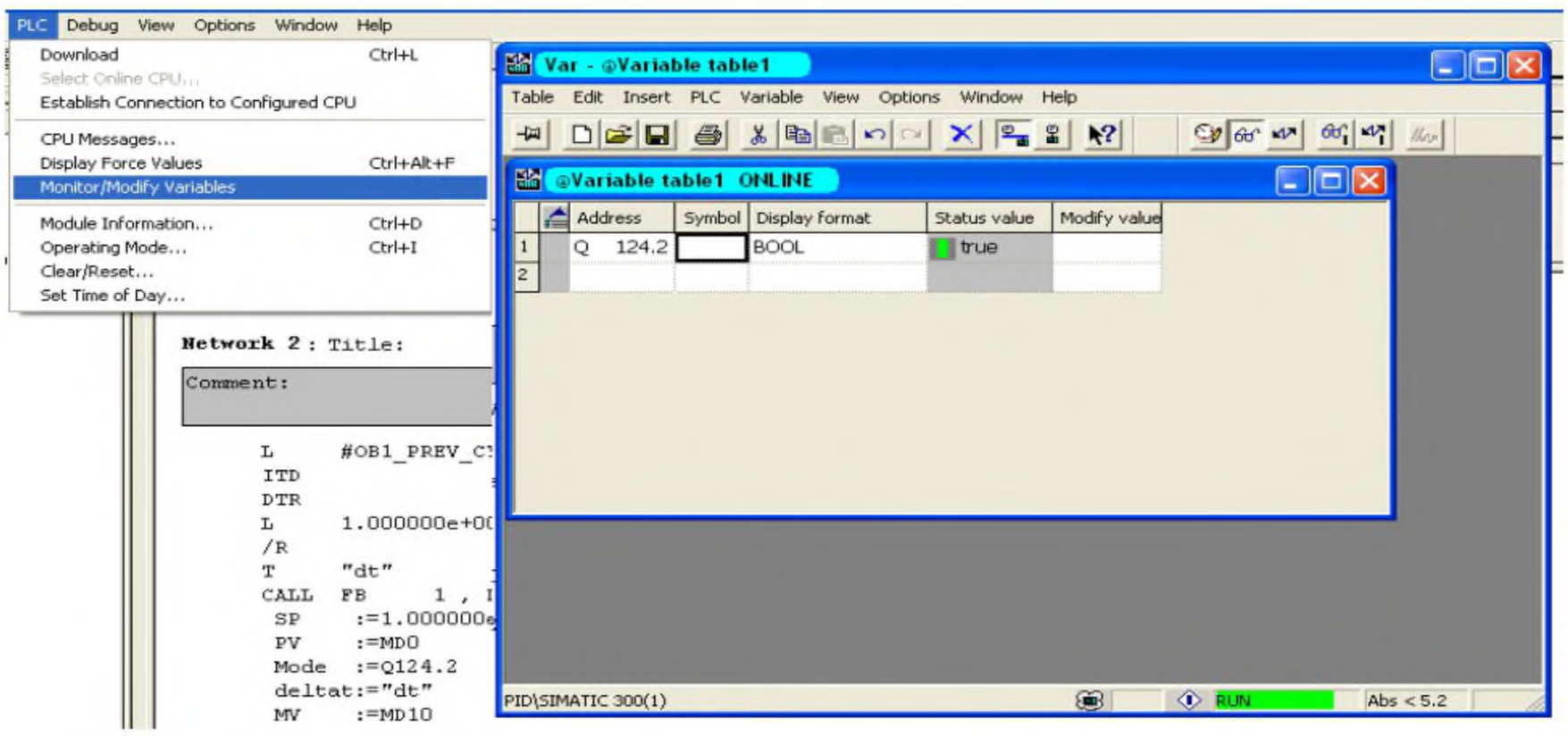


Рис. 5.3. Инструментарий Monitor/Modify Variables

Управление работой программы пользователя реализуется через панель эмулятора контроллера, откуда возможно внесение изменений в память контроллера и мониторинг ее состояния. Для запуска эмулятора необходимо кликнуть на значек  . На рис. 5.4 представлена панель эмулятора контроллера. На рис. 5.5 представлена панель эмулятора контроллера.

Кликнув левой кнопкой мыши на вкладке Insert (рис. 5.6), можно выбрать Input Variable (входные переменные), Output Variable (выходные переменные), переменные, которые находятся в меркерной памяти контроллера (Bit Memory).

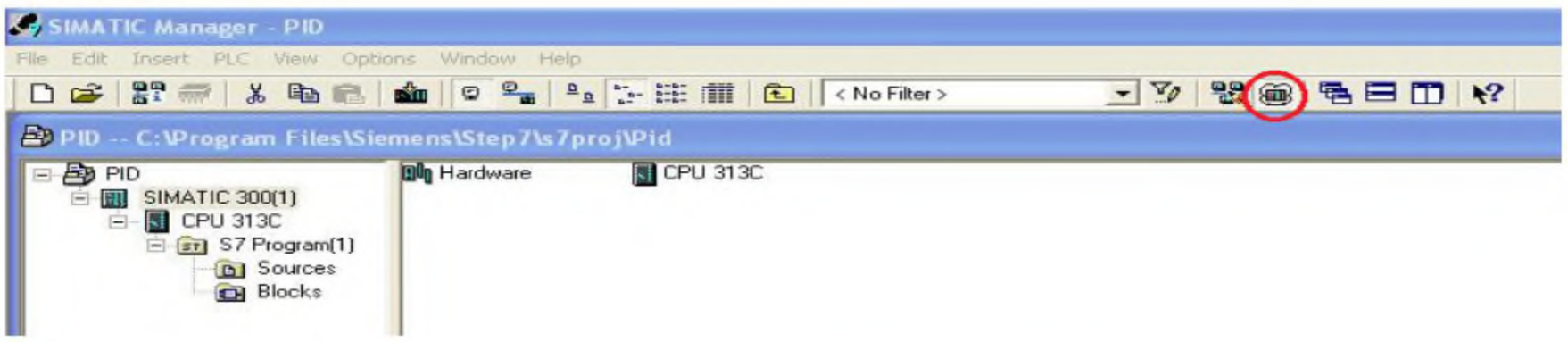


Рис. 5.4. Запуск эмулятора контроллера

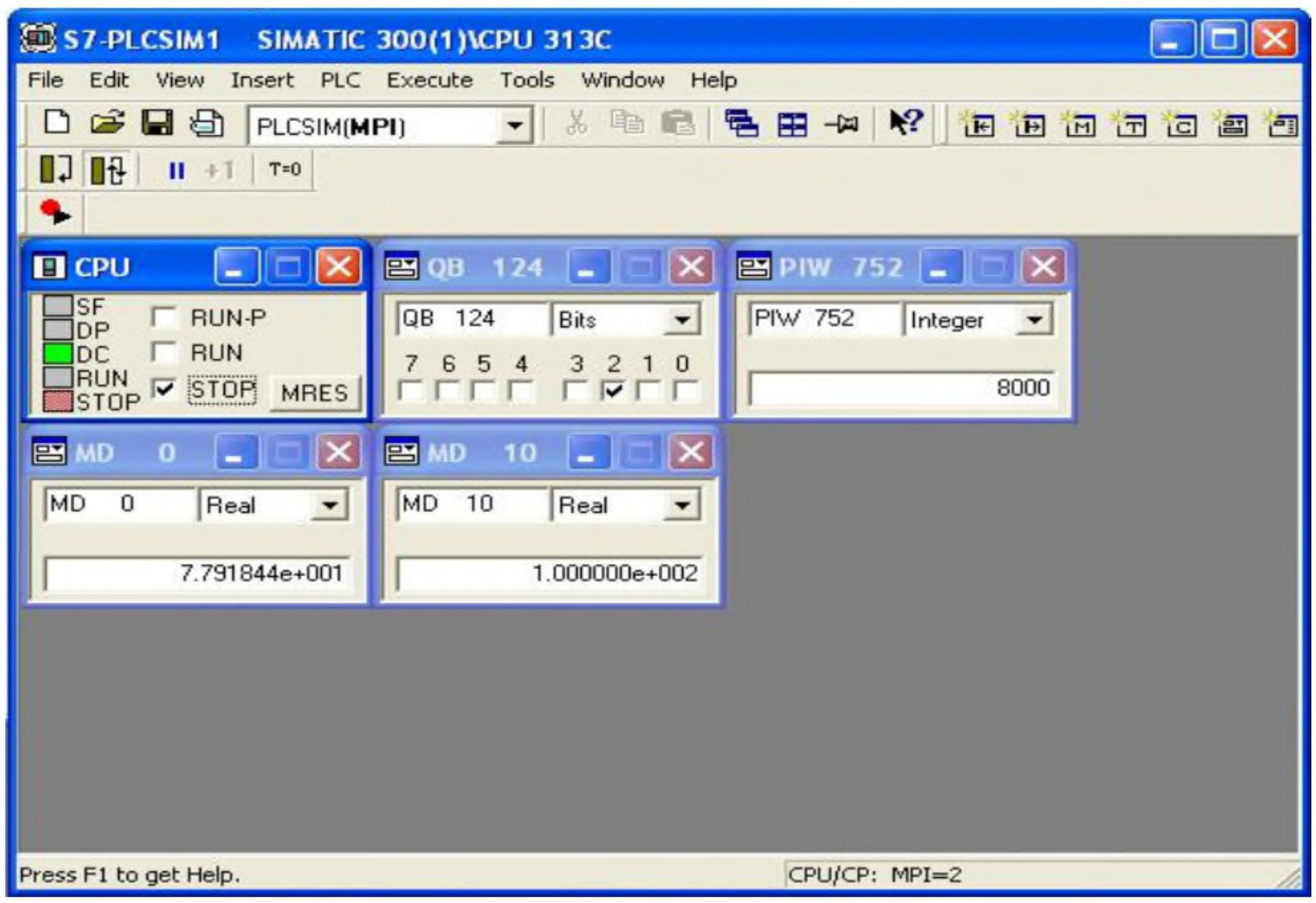


Рис. 5.5. Общий вид эмулятора

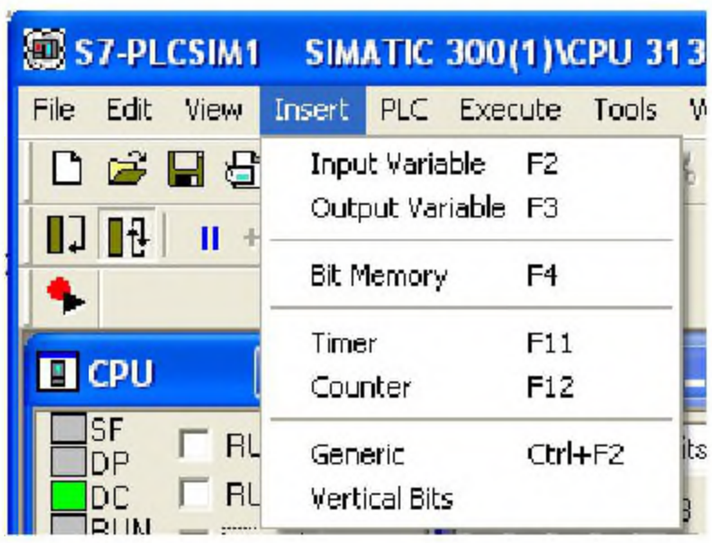


Рис. 5.6. Выбор переменных в эмуляторе

Рис. 5.7. Запуск эмулятора контроллера

Для запуска эмулятора необходимо установить галочку в режиме RUN (рис. 5.7).

Кнопка MRES осуществляет сброс памяти контроллера.

Глава 6. КОНФИГУРИРОВАНИЕ SCADA-СИСТЕМЫ WINCC

6.1. Общие сведения

При первом запуске WinCC на рабочем столе отображается WinCCExplorer (рис. 6.1), из которого происходит разработка всего проекта. Основные редакторы, которые используются при разработке проекта, – Tag Management и Graphics Designer.

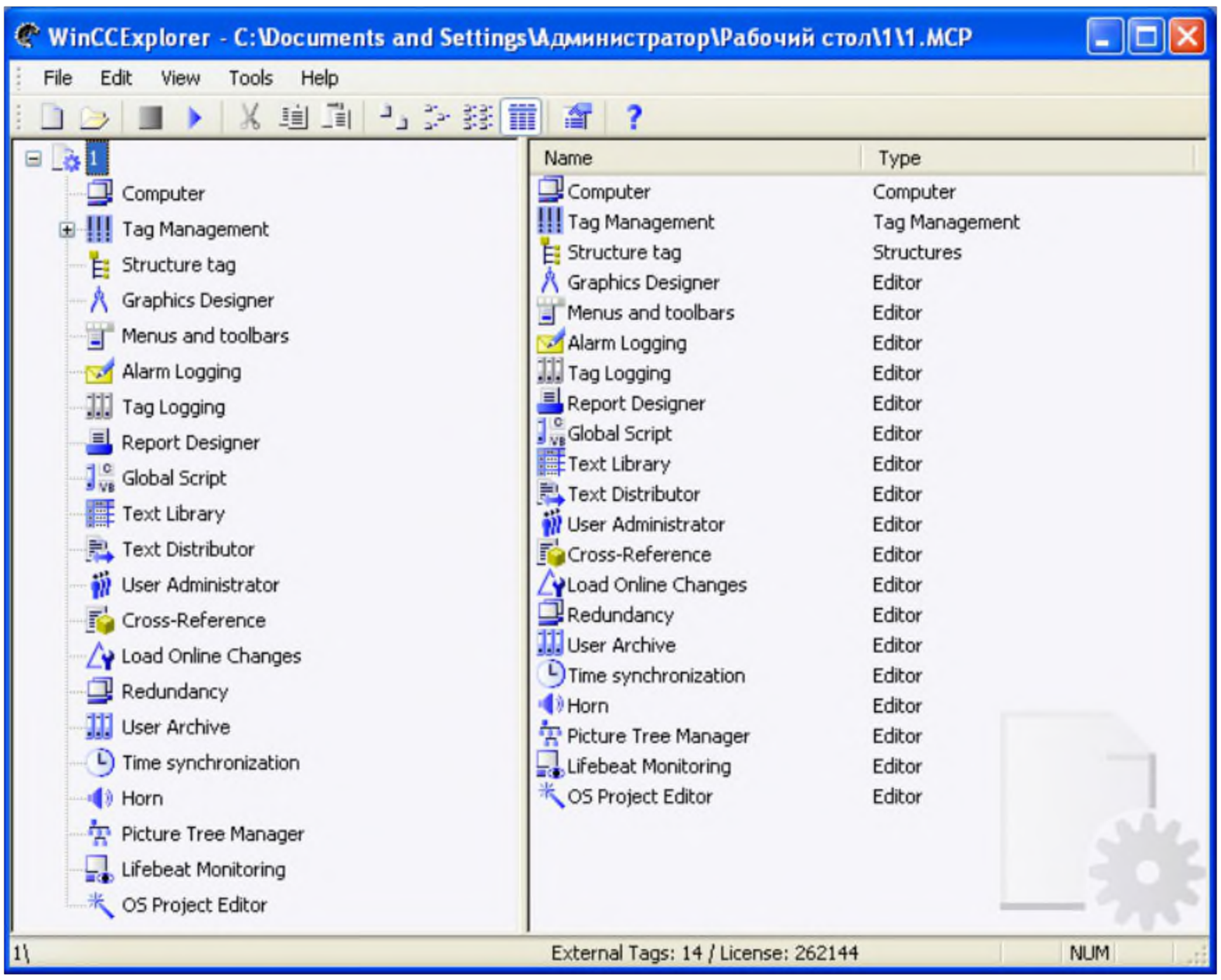


Рис. 6.1. Основное окно проекта

Создание нового проекта начинается с создания подключения к контроллеру. Во вкладке Tag Management выбираем SIMATIC S7 PROTOCOL SUITE, далее выбираем один из интерфейсов подключения контроллера к ПК, в данном случае это MPI. Выбираем из контекстного меню MPI функцию New Driver Connection (рис 6.2).

Следующим этапом создания проекта является составление списка тэгов (Tag). Тэги – это переменные, связывающие WinCC с контроллером. Для создания тэга вызываем контекстное меню подключения, созданного в MPI, и выбираем «New Tag…» (рис. 6.3).

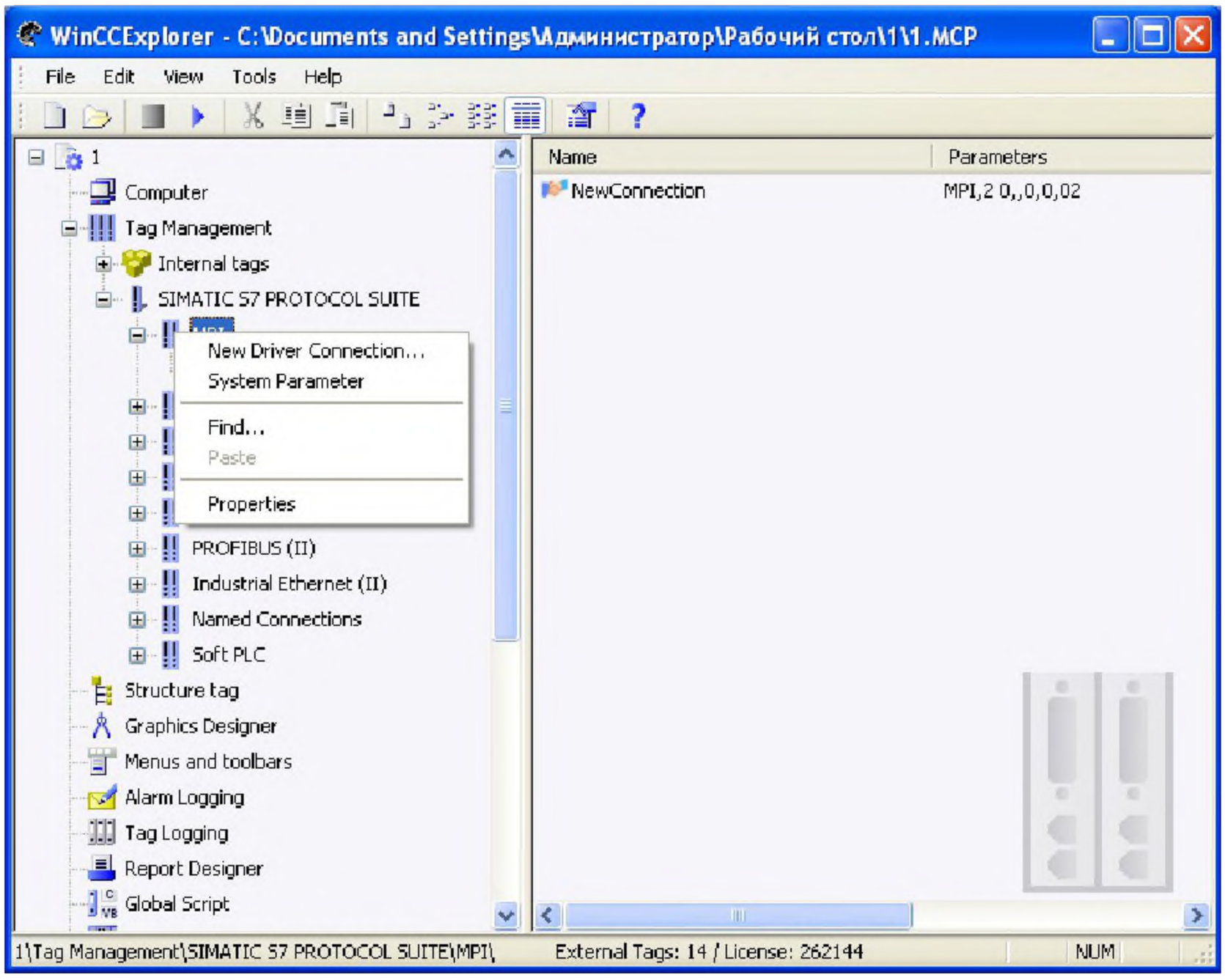


Рис. 6.2. Выбор типа подключения

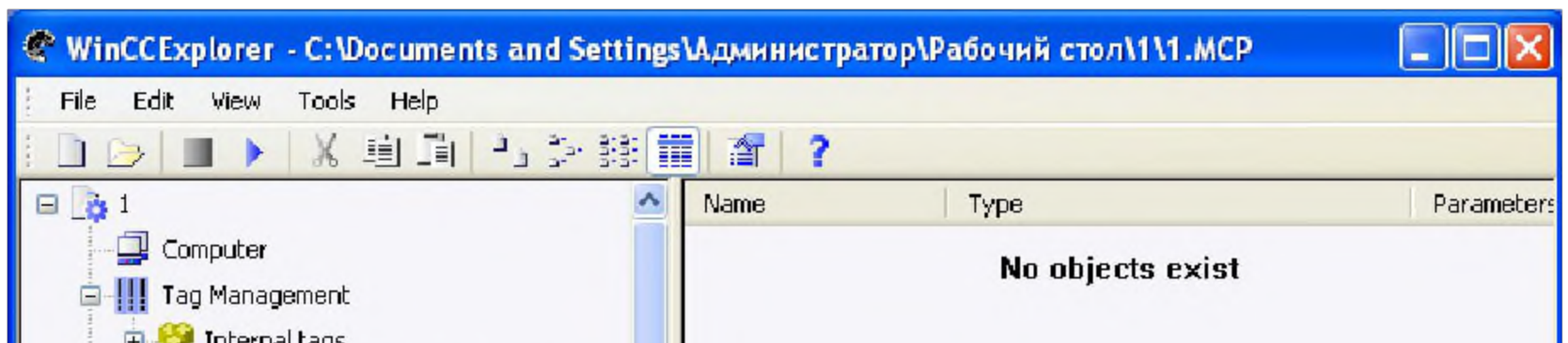


Рис. 6.3. Создание тэга

В открывшемся окне прописываем имя тэга в поле Name, выбираем тип данных DataType (рис. 6.4). Нажимаем кнопку select, выбираем тип данных и адрес переменной в контролере (рис. 6.5).

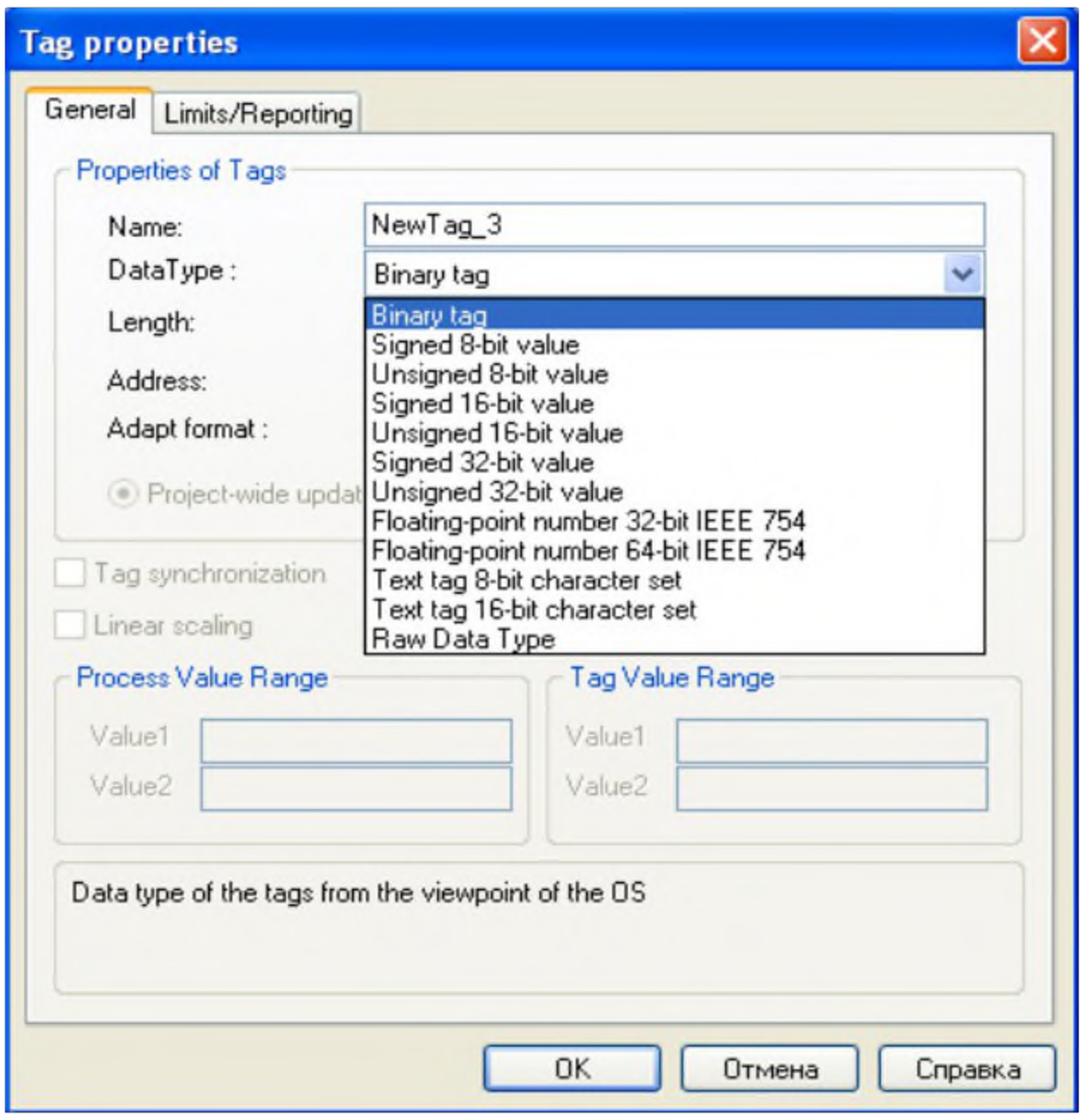


Рис. 6.4. Создание тэга

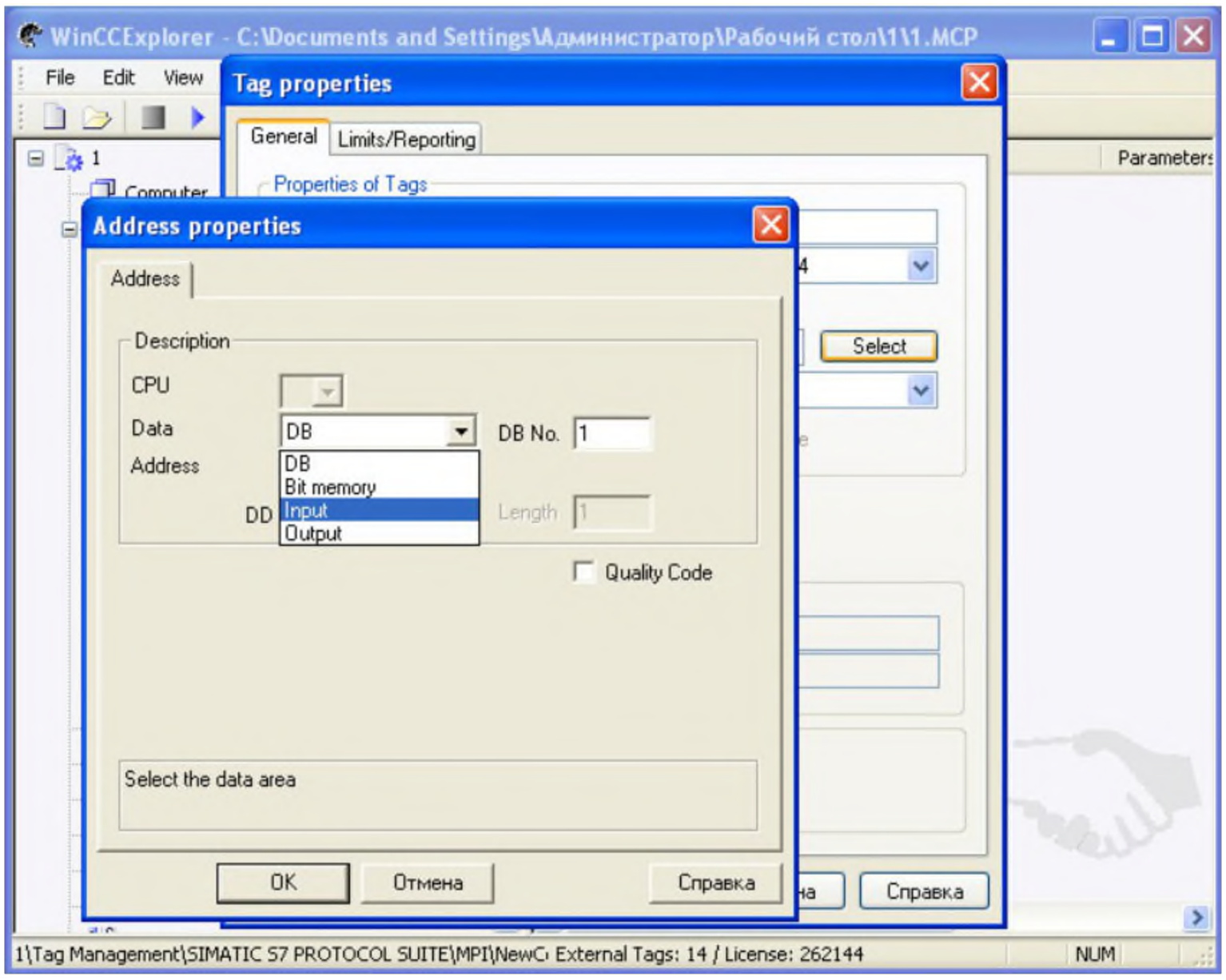


Рис. 6.5. Создание тэга

В данной работе используются дискретные и аналоговые сигналы входов и выходов, поэтому таблица тэгов будет иметь вид, представленный в табл. 6.1. Таблица тэгов необходима для того, чтобы объявить все используемые блоки, входы, выходы и т.д. и присвоить им соответствующие адреса и типы данных.

Таблица 6.1

Таблица тэгов

Имя	Тип данных	Адрес
pc_start	Binary Tag	M10.1
pc_stop	Binary Tag	M10.2
zadanie	Floating-point number 32-bit IEEE 754	MD34
start_Pregul	Binary Tag	M38.0
rasxod	Floating-point number 32-bit IEEE 756	MD20
valve	Floating-point number 32-bit IEEE 757	MD42
zadanie_valve	Floating-point number 32-bit IEEE 758	MD48
Kp	Floating-point number 32-bit IEEE 759	DB1,DD18
Ti	Floating-point number 32-bit IEEE 760	DB1,DD22
Td	Floating-point number 32-bit IEEE 761	DB1,DD26

6.2. Создание пользовательского интерфейса в Graphics Designer

6.2.1. Описание компонентов Graphics Designer

После создания всех тэгов приступаем к созданию интерфейса пользователя. Запускаем Graphics Designer (рис. 6.6) из WinCCExplorer (см. рис. 6.1).

Все основные объекты для создания внешнего вида интерфейса находятся в Object Palette, они делятся на следующие:

1) Standart objects – различные геометрические объекты;

2) Smart Objects – динамические объекты (объекты ввода/вывода информации);

3) Windows Objects – кнопки, переключатели;

4) Tube Objects – инструмент для рисования трубопроводов.

Создадим мнемосхему контура управления расходом, для этого используются следующие компоненты из Object Palette:

1) Siemens HMI Symbol Library (условное обозначение насоса и резервуара);

2) I/O Field (отображение или передача числовой информации);
3) Static Text (создание статичных надписей);
4) Button (создание кнопок);
5) Slider Object (создание слайдера);
6) WinCC Online TrendControl (компонент для отображения тренда);
7) Polygon tube.

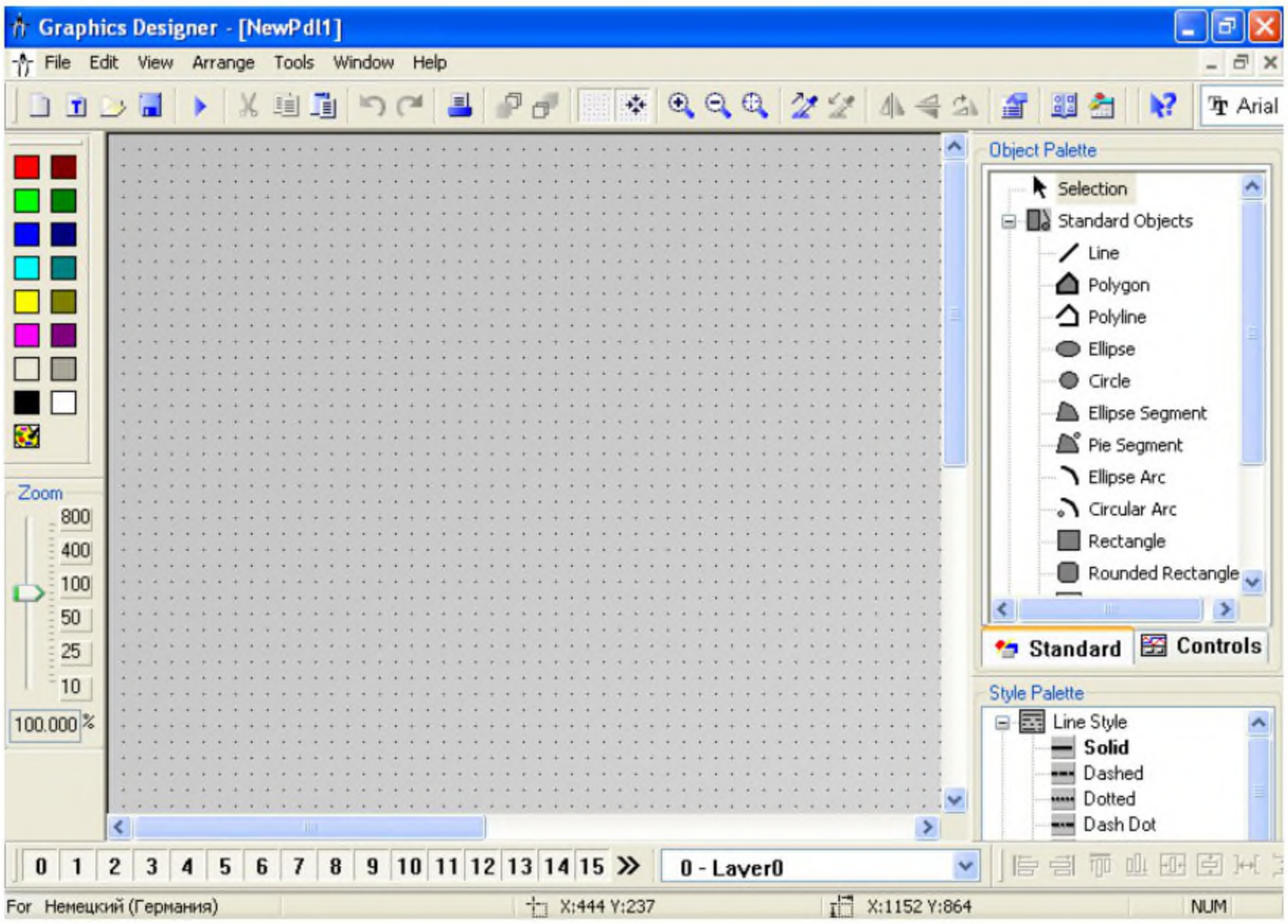


Рис. 6.6. Основное окно Graphics Designer

Результат создания мнемосхемы представлен на рис. 6.7.

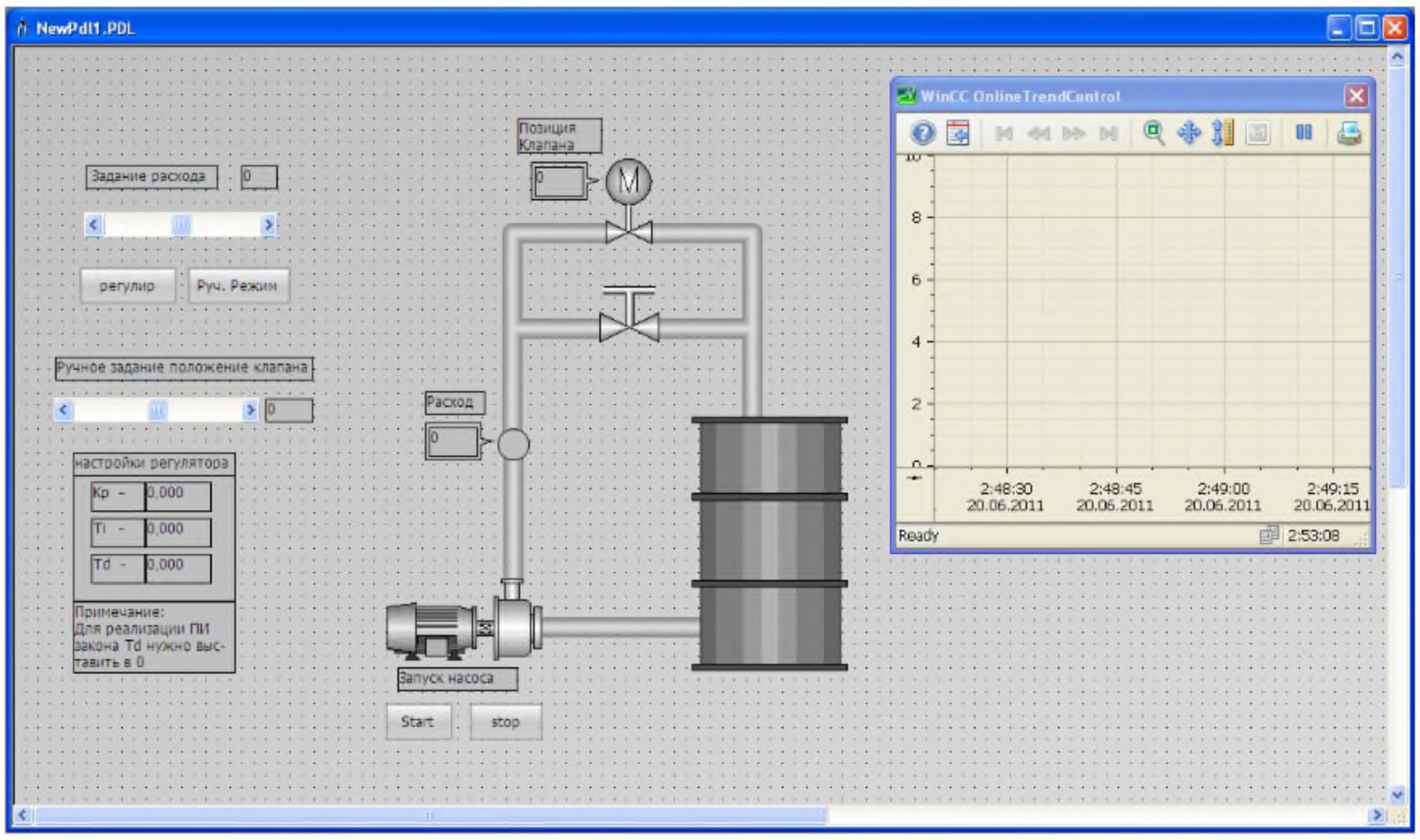


Рис. 6.7. Мнемосхема контура управления

6.2.2. Конфигурирование объектов в Graphics Designer

Для конфигурирования I/O-Field в контекстном меню объекта выбираем Configuration Dialog. После выполнения данной операции появляется окно, представленное на рис. 6.7. Поле Tag служит для выбора тэга, Update – время обновления данного тэга, Type – тип тэга (Output – только отображение, Input – только передача информации в контроллер, I/O Field – и отображение, и передача). Выбор самого тэга происходит в окне Tags-Project (рис. 6.8).

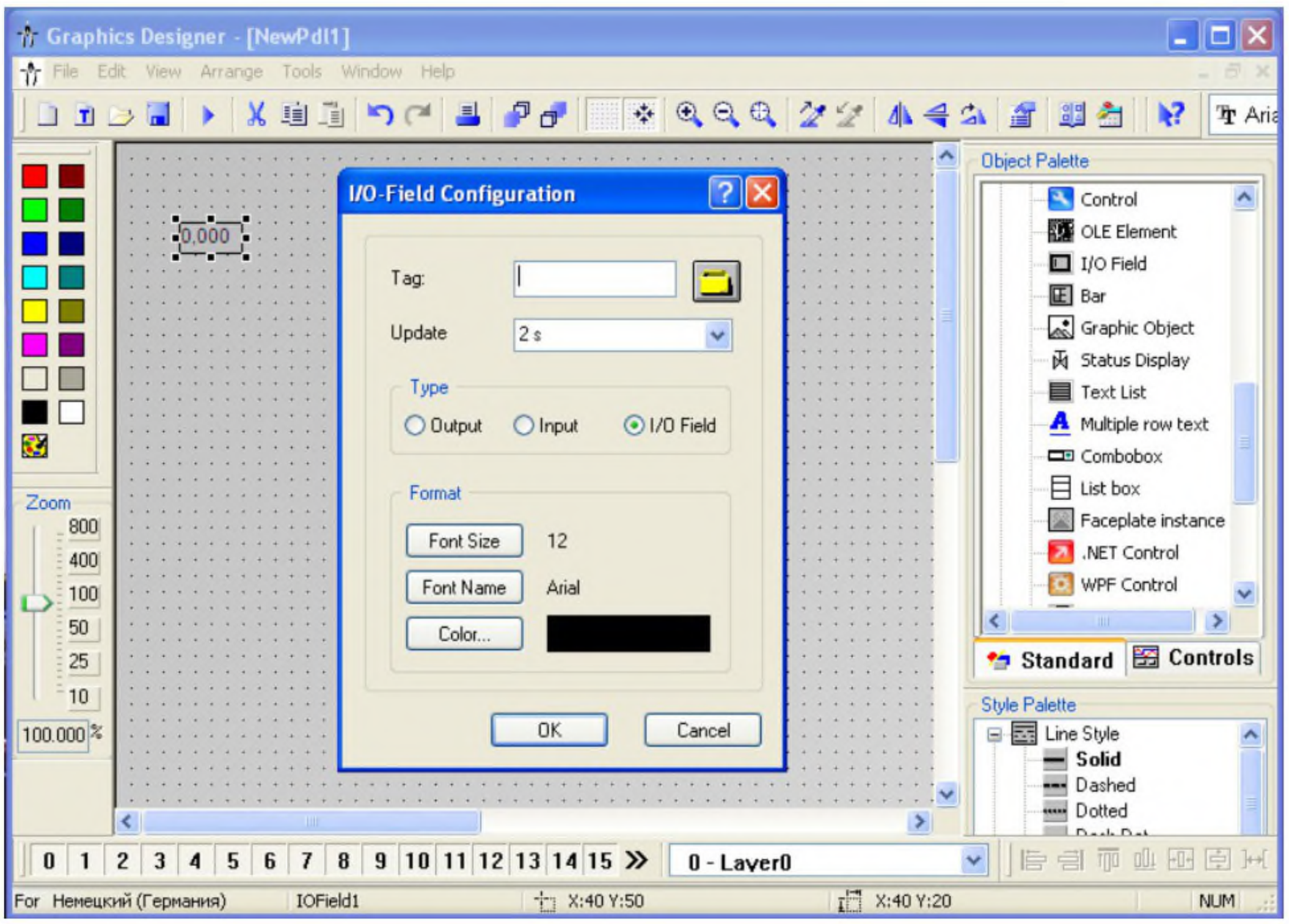


Рис. 6.8. Конфигурационное меню объекта

Для редактирования объекта Static Text необходимо двойным щелчком мыши нажать на него, в результате чего появится возможность редактировать текст данного объекта.

Для конфигурирования компонента Button необходимо двойным щелчком мыши нажать на объект, в появившемся окне переходим во вкладку Events, выбираем пункт Mouse. Для имитации нажатия кнопки выполним следующие действия. Двойным щелчком выбираем пункт Press left. В открывшемся дополнительном окне в области Source указываем пункт Constant и устанавливаем значение

«1». В области Target выбираем пункт Tag (рис 6.10). Выбор соответствующего тэга аналогичен вышеописанному пункту. То же самое необходимо сделать и в пункте Release left, установив при этом значение Constant в «0».

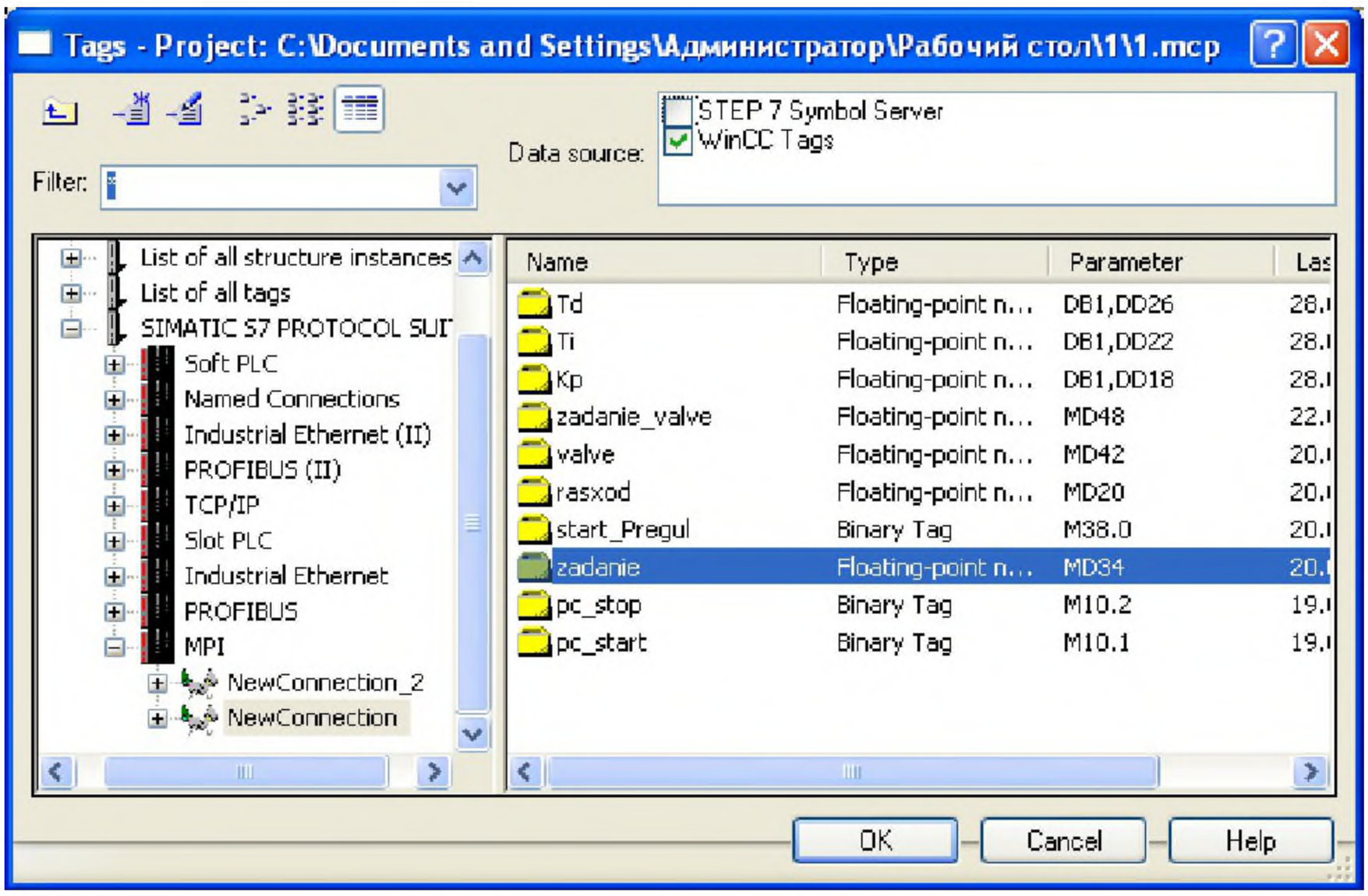


Рис. 6.9. Выбор тэга для объекта

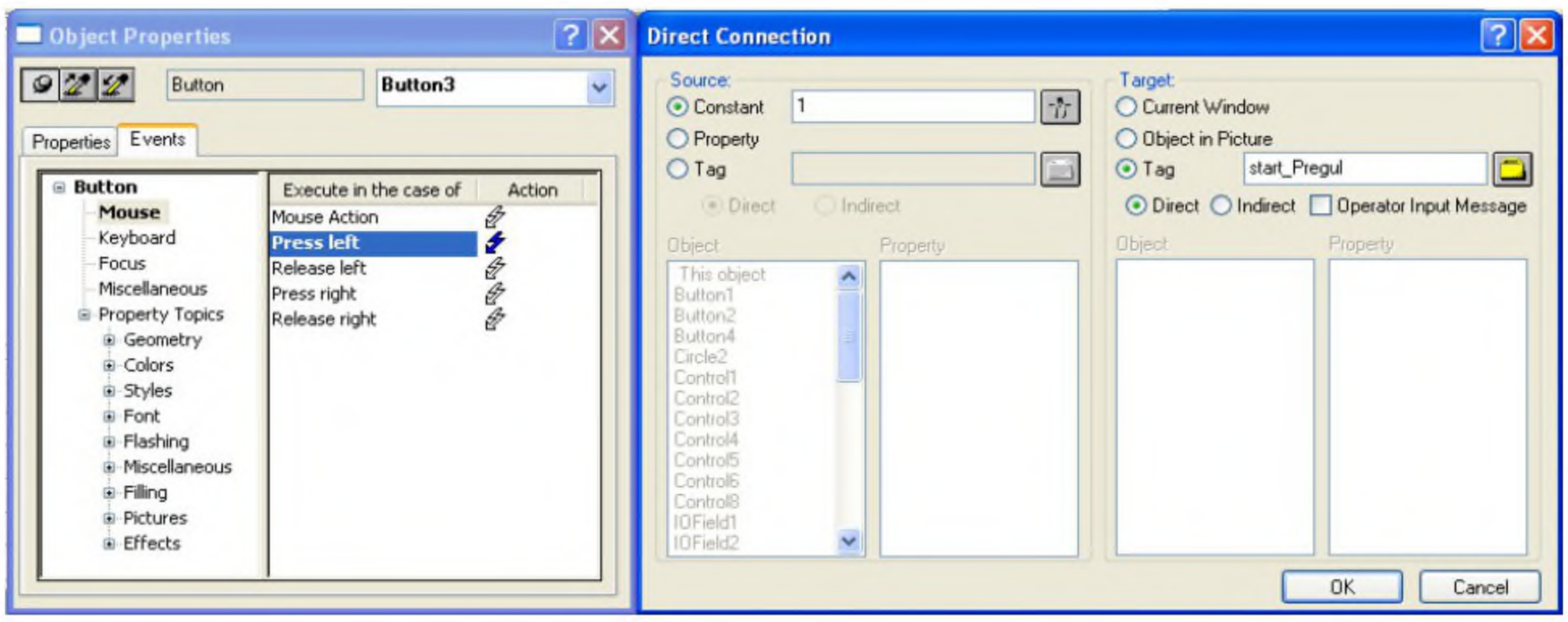


Рис. 6.10. Конфигурирование компонента Button

Для конфигурирования Slider Object в контекстном меню объекта выбираем Configuration Dialog. Поле Tag служит для выбора тэга, Update – время обновления данного тэга, в области Limits задаются максимальное и минимальное значение слайдера (Max. Val-

ue, Min. Value), а также шаг отображения (Step). В области Orientation выбирается положение слайдера – горизонтальное или вертикальное (рис 6.11). Выбор соответствующего тэга аналогичен.

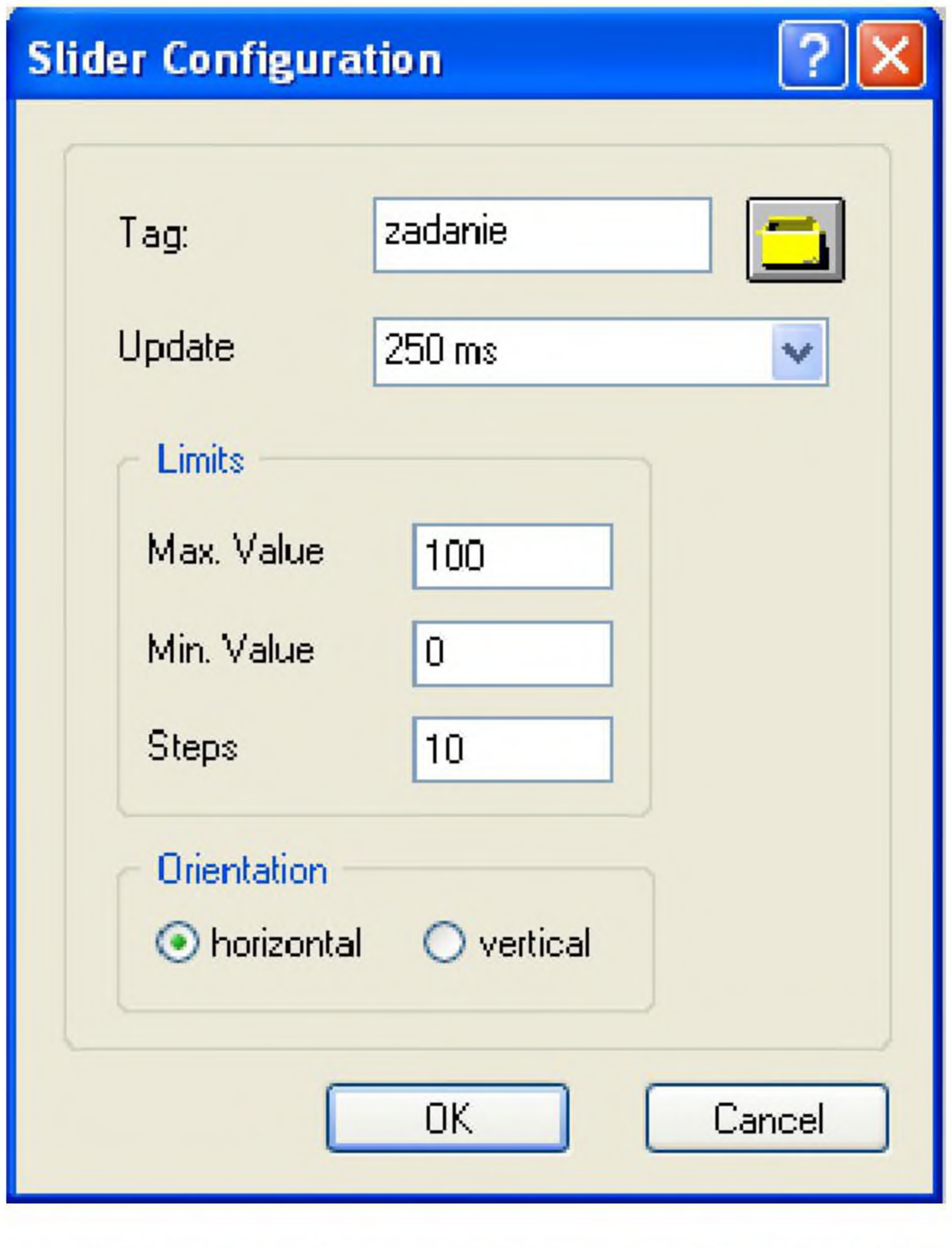


Рис. 6.11. Конфигурирование Slider Object

Для конфигурирования WinCC Online TrendControl в контекстном меню объекта выбираем Configuration Dialog. Переходим на вкладку Trends, в области Data Connection выставляем Data Source в положение Online tags, а в Tag Name нажимаем кнопку . В появившемся окне Tage Configuration выбираем необходимый тэг, как описано выше, а также время его обновления Cycle Time (рис. 6.12).

Для конфигурирования Siemens HMI Symbol Library необходимо двойным щелчком мыши нажать на объект, в появившемся окне выбрать категорию символа и необходимый символ из списка (рис. 6.13).

Для запуска проекта используем кнопку Run .

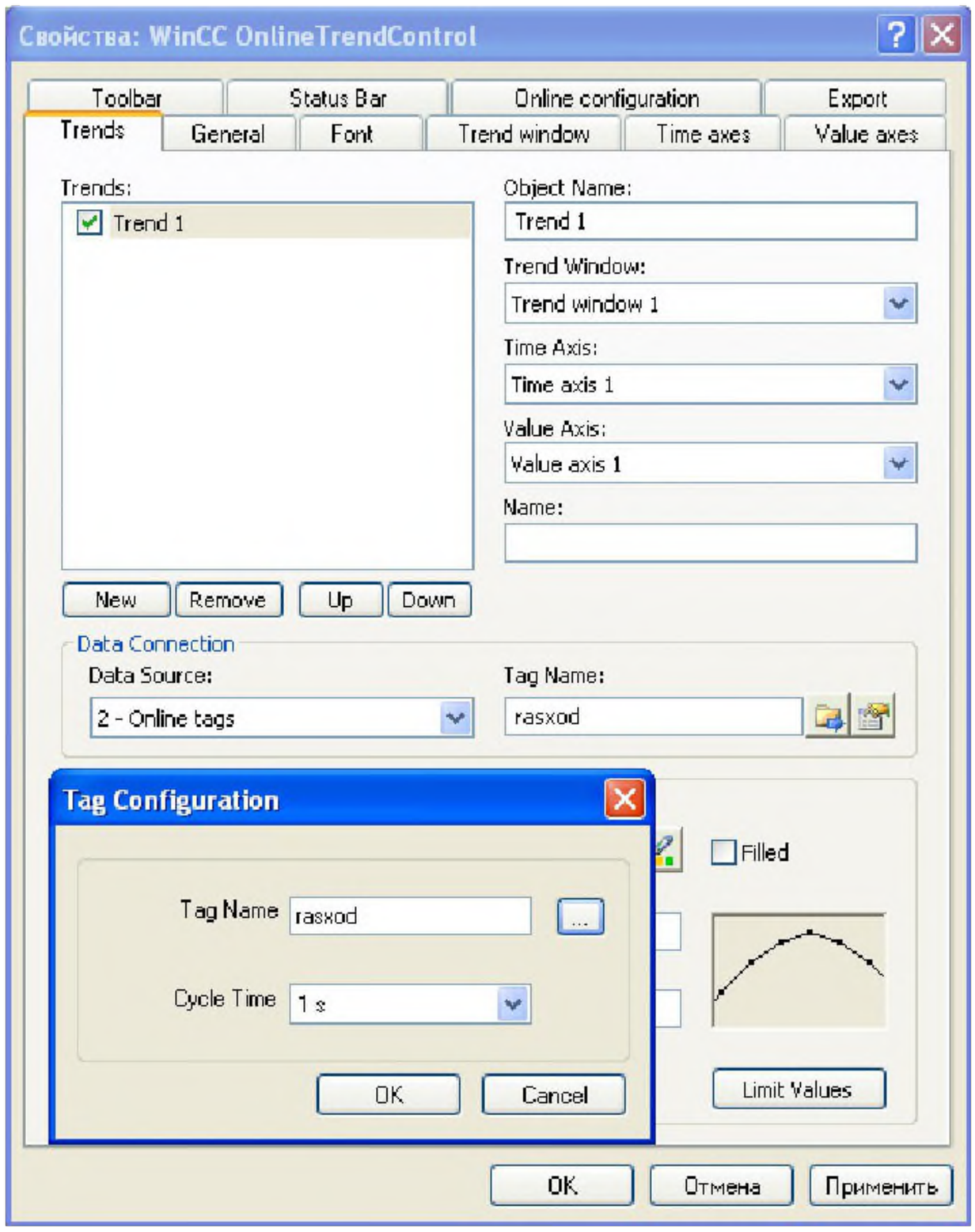


Рис. 6.12. Конфигурирование WinCC Online TrendControl

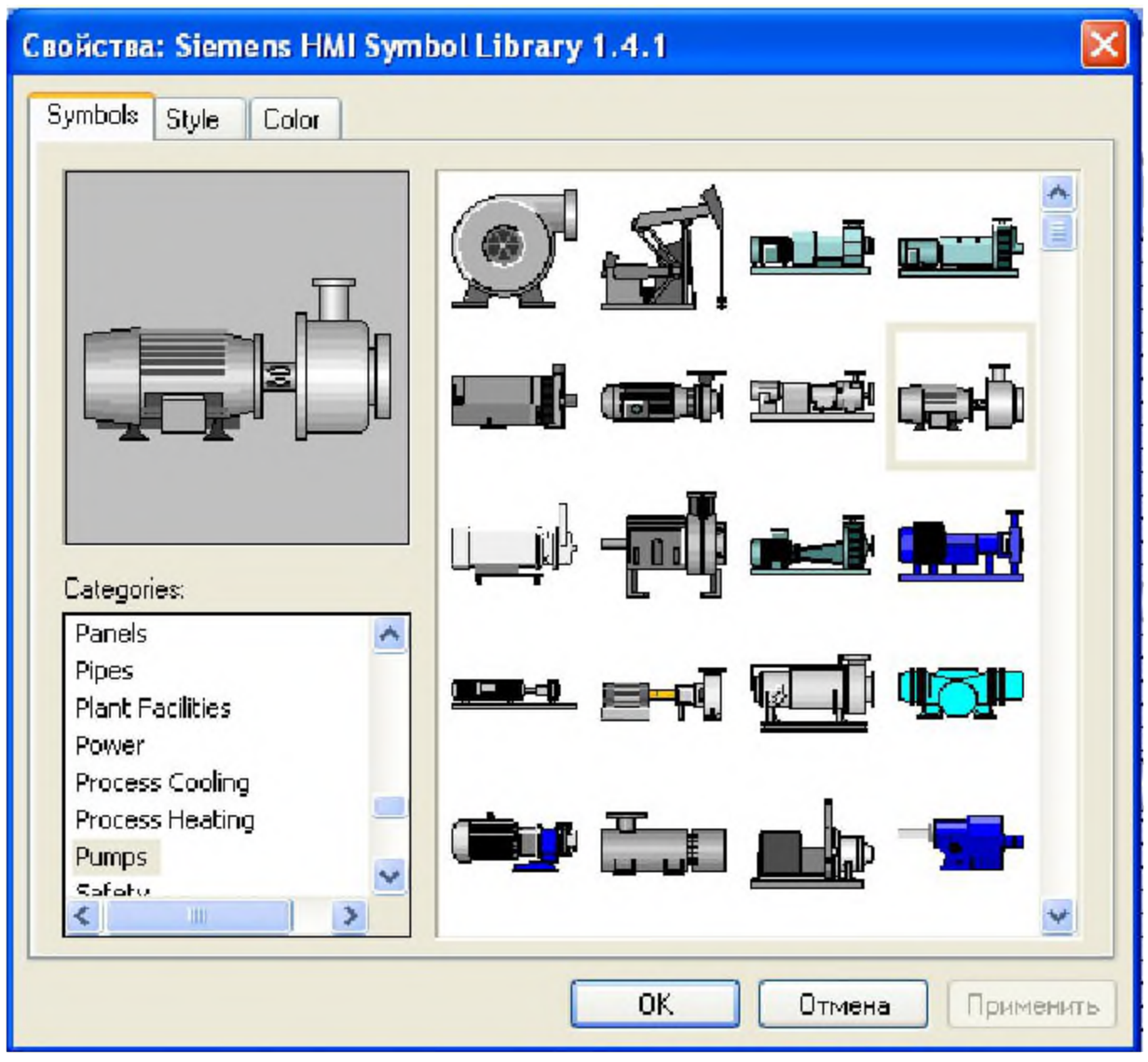


Рис. 6.13. Конфигурирование Siemens HMI Symbol Library

Глава 7. АЛГОРИТМИЗАЦИЯ СИСТЕМЫ ПРОГРАММНО-ЛОГИЧЕСКОГО УПРАВЛЕНИЯ НА БАЗЕ HONEYWELL EXPERION PKS

7.1. Агоритмизация системы управления клапаном-отсекателем

Процесс алгоритмизации системы управления клапаном-отсекателем можно представить следующим перечнем задач:

1) создание схемы управления клапаном-отсекателем;

2) загрузка схемы в эмулятор контроллера и активирование эмулятора;

3) создание мнемосхемы управления клапаном-отсекателем;

4) создание динамического элемента для отображения состояния клапана на мнемосхеме;

5) осуществление коррекции схемы управления и элементов на мнемосхеме.

7.1.1. Создание схемы управления

Запуск Configuration Studio из панели «Пуск» показан на рис. 7.1.

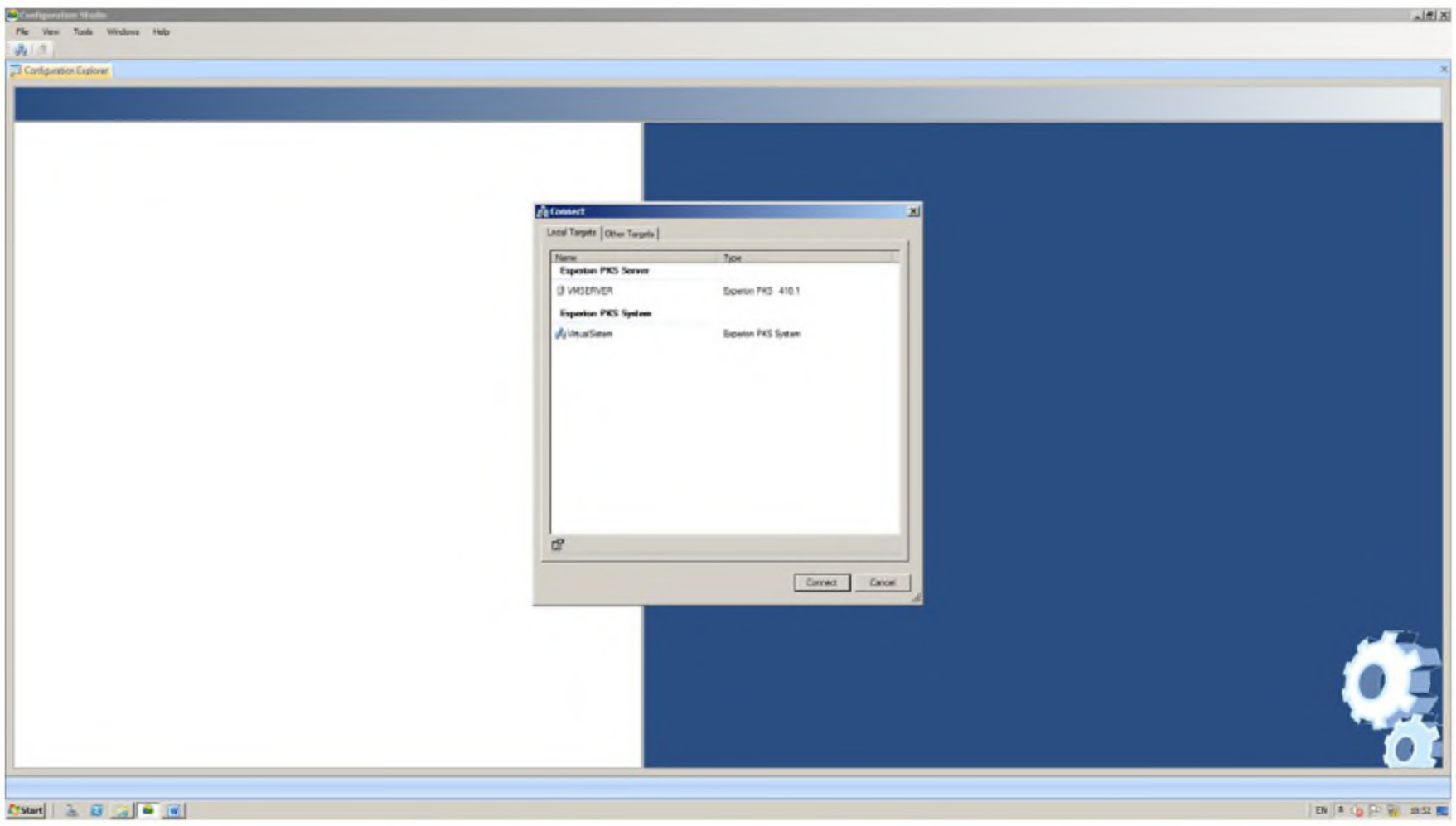


Рис. 7.1. Configuration Studio

Необходимо настроить подключение к VMSERVER, для этого нужно нажать Connect. После этого появится окно, представленное на рис. 7.2.

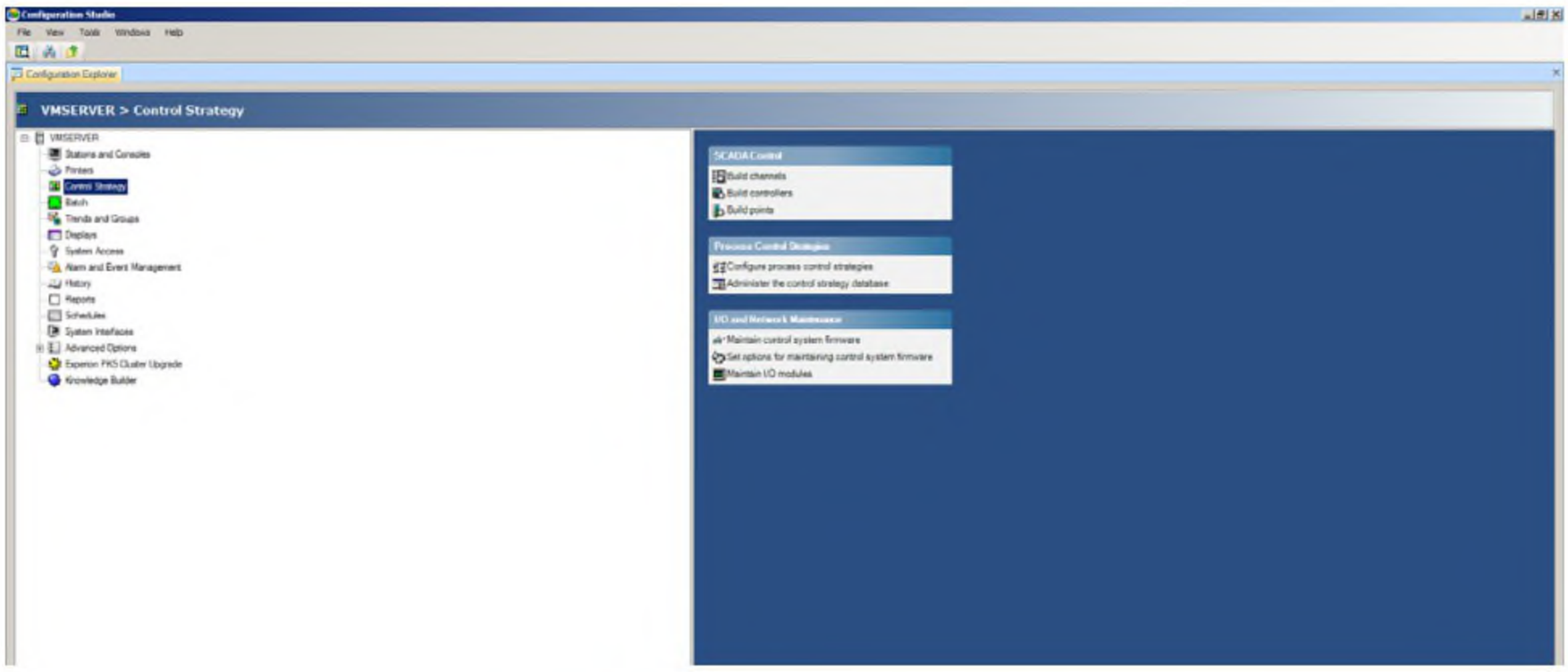


Рис. 7.2. Configuration Studio

Выбрать Control Strategy / Configure process control strategies. Появиться окно, представленное на рис. 7.3.

Рис. 7.3. Control Bulder

В главном меню выбрать File → New → Control modyle, как показано на рис. 7.4.

Появится рабочее окно, представленное на рис. 7.5.

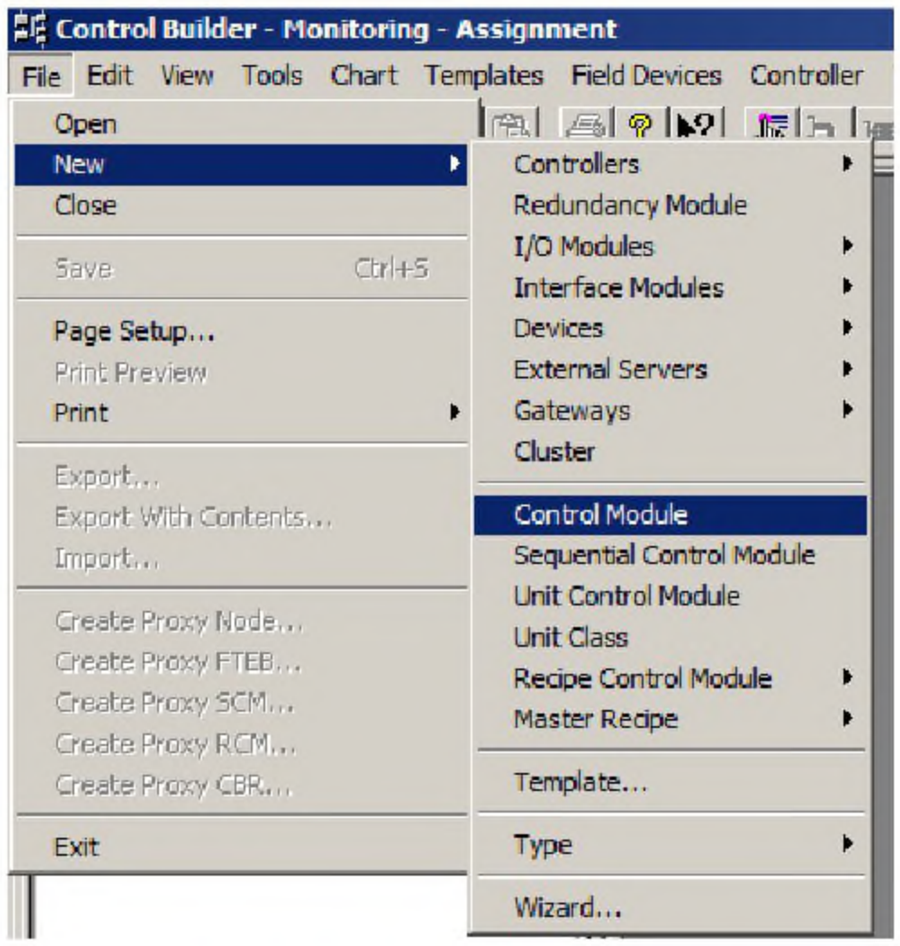


Рис. 7.4. Создание Control modyle

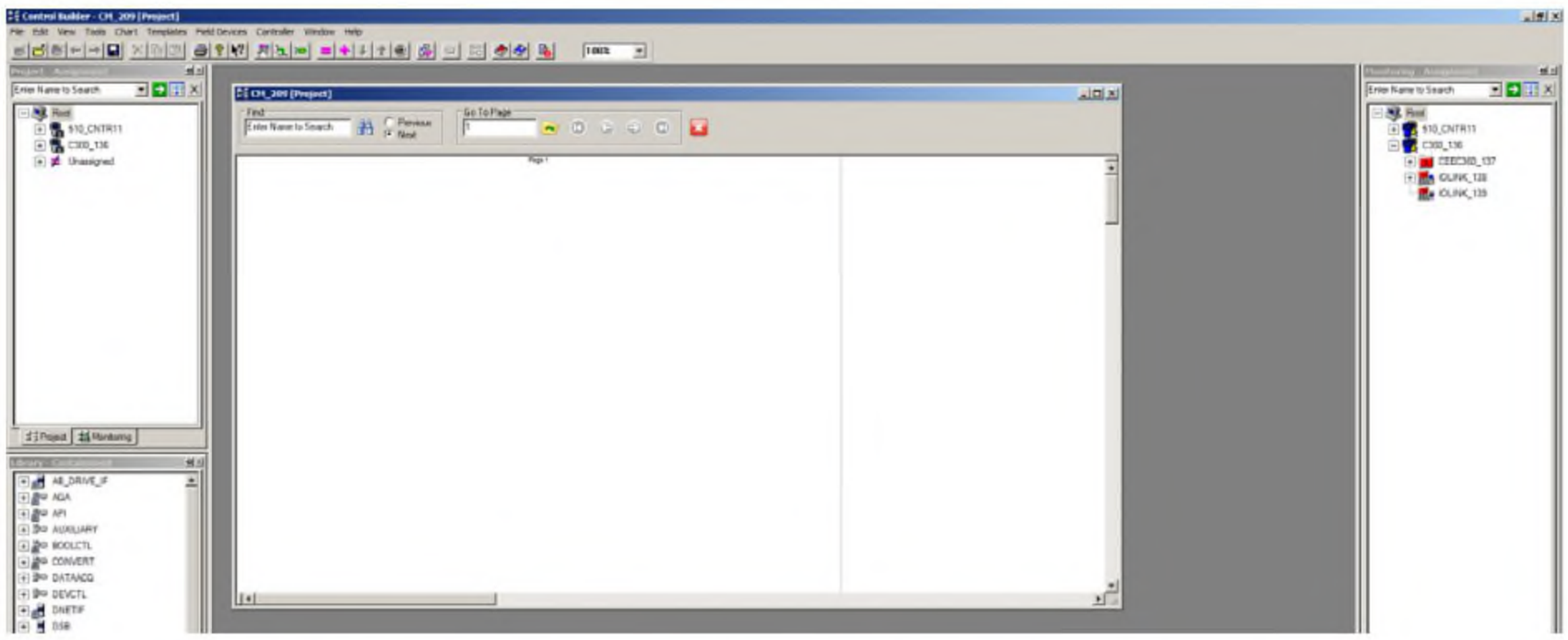


Рис. 7.5. Окно созданного Control modyle

Структурная схема управления клапаном-отсекателем представлена на рис. 7.6. В данной схеме используются блоки, описание которых представлено в табл. 7.1.

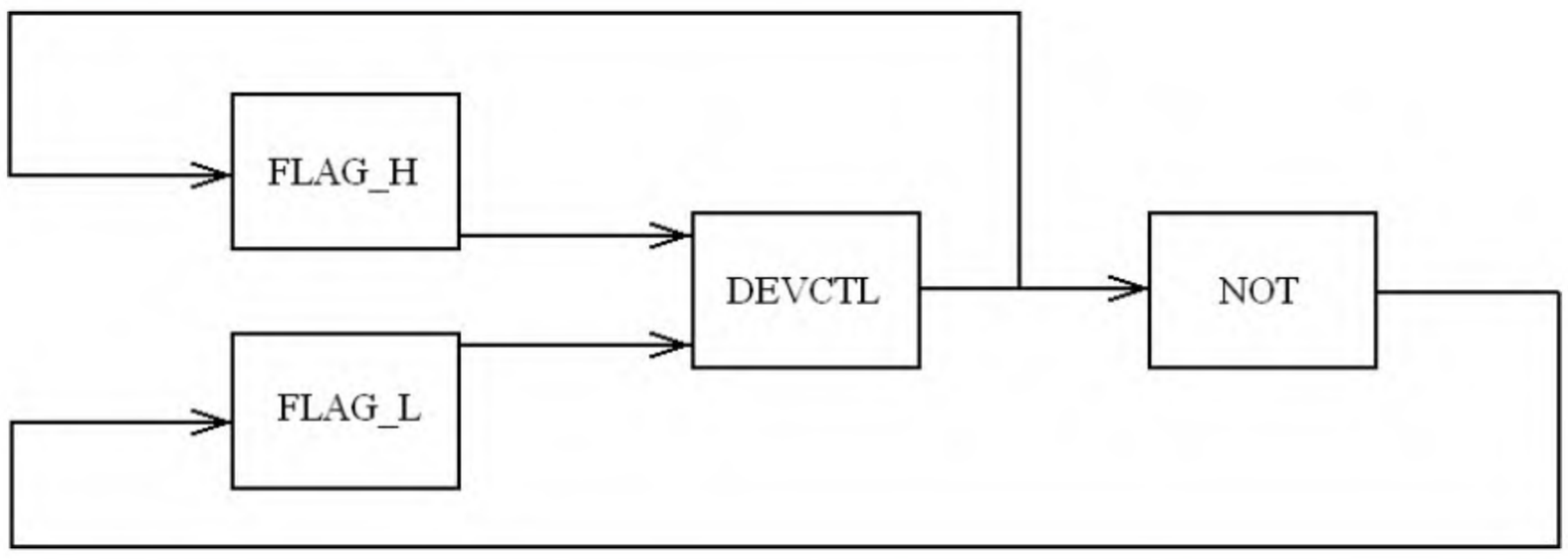


Рис. 7.6. Структурная схема управления клапаном-отсекателем

Таблица 7.1

Описание функциональных блоков

Название блока	Описание блока	Функции блока
Блок DEVCTL (управления устройствами)	Обеспечивает возможность выполнения функции нескольких входов/выходов для взаимодействия с устройствами дискретного действия, такими как двигатели, насосы, соленоидные клапаны и клапаны с приводами. Блок управления устройствами содержит встроенные структуры для обработки блокировок и поддерживает дисплей групп условий блокировки, а также фрагментарный и графический дисплеи	1. Позволяет управлять заданиями дискретных выходных значений и интерпретирует соответствующие сигналы обратной связи дискретных входных значений, представленные параметром состояния PV (текущее состояние обратной связи). 2. Функционирование включает в себя передачу команд, представленных параметром состояния OP (заданное командой выходное состояние), контроль PV и генерацию сигналов тревоги на основе различных заранее определенных условий, например, в случае, если PV не достигнет состояния, заданного в OP. 3. Обеспечивает выполнение защитных блокировок, блокировок, связанных с отдельными состояниями, набора команд инициализации, сбора статистических данных, связанных с эксплуатацией устройств, а также функций запуска периодического процесса уровня 1
Блок FLAG (флаг)	Обеспечивает сохранение одного параметра, имеющего два устойчивых значения, к которому возможно обращение как к простому логическому значению (ON или OFF) с помощью параметра PVFL или как	Обеспечивает функцию индикации события

Окончание табл. 7.1

Название блока	Описание блока	Функции блока
	к одному из двух сконфигурированных пользователем значений состояния (например, «Работа» и «Останов») через параметр PV	
Блок NOT (инверсия)	Предоставляет алгоритм NOT, обеспечивающий выполнение функции инвертирования	Инвертирует состояние дискретного входа (IN) таким образом, что выходной параметр OUT является дополнением к входному параметру

Далее необходимо в дереве элементов выбрать Utility → Flag (рис. 7.7).

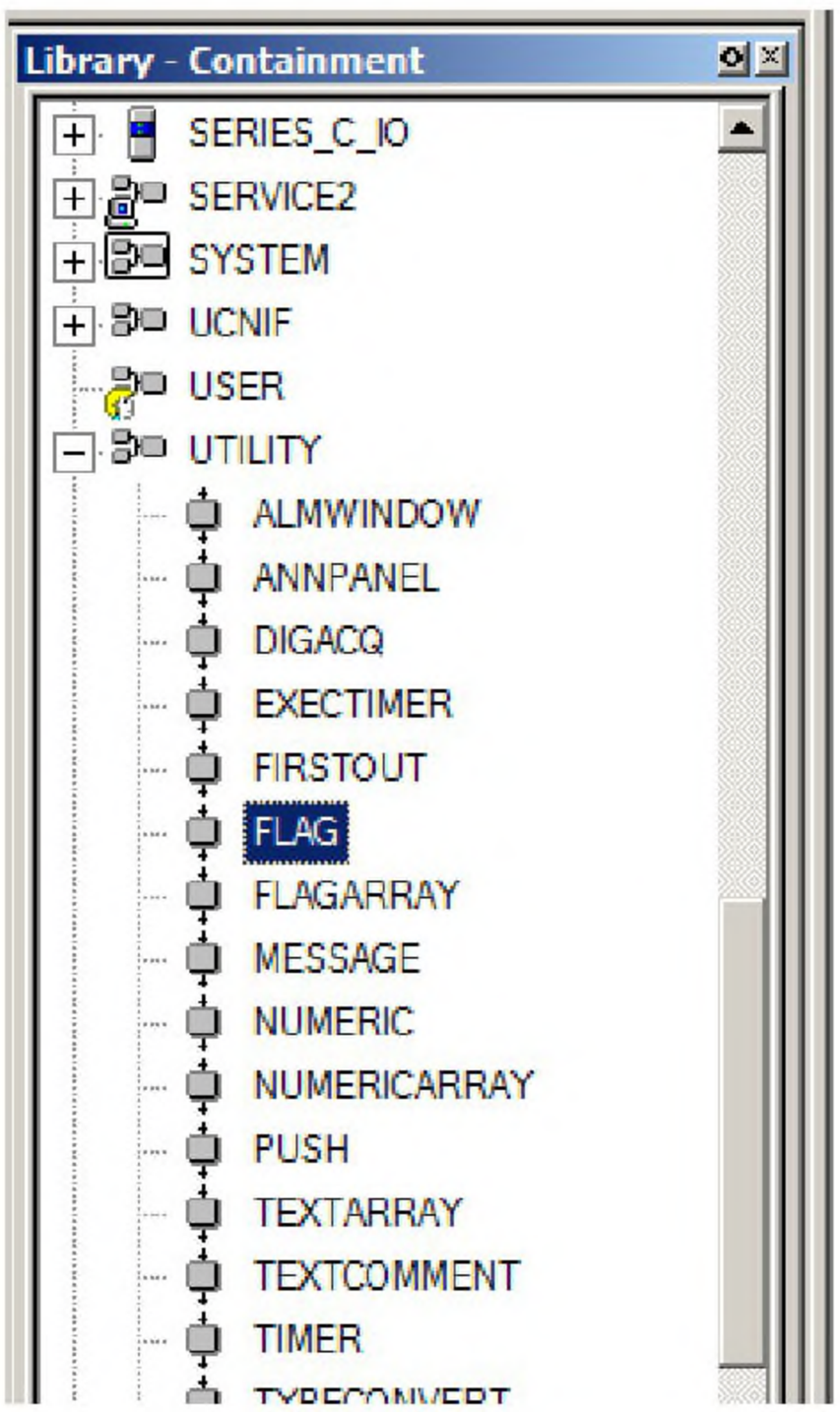


Рис. 7.7. Дерево функциональных блоков

Названия элементов (флагов), имитирующих открытое и закрытое состояния клапана, будут «FL_H» и «FL_L» соответственно. Количество входных и выходных сигналов настраивается в соответствии с рис. 7.8.

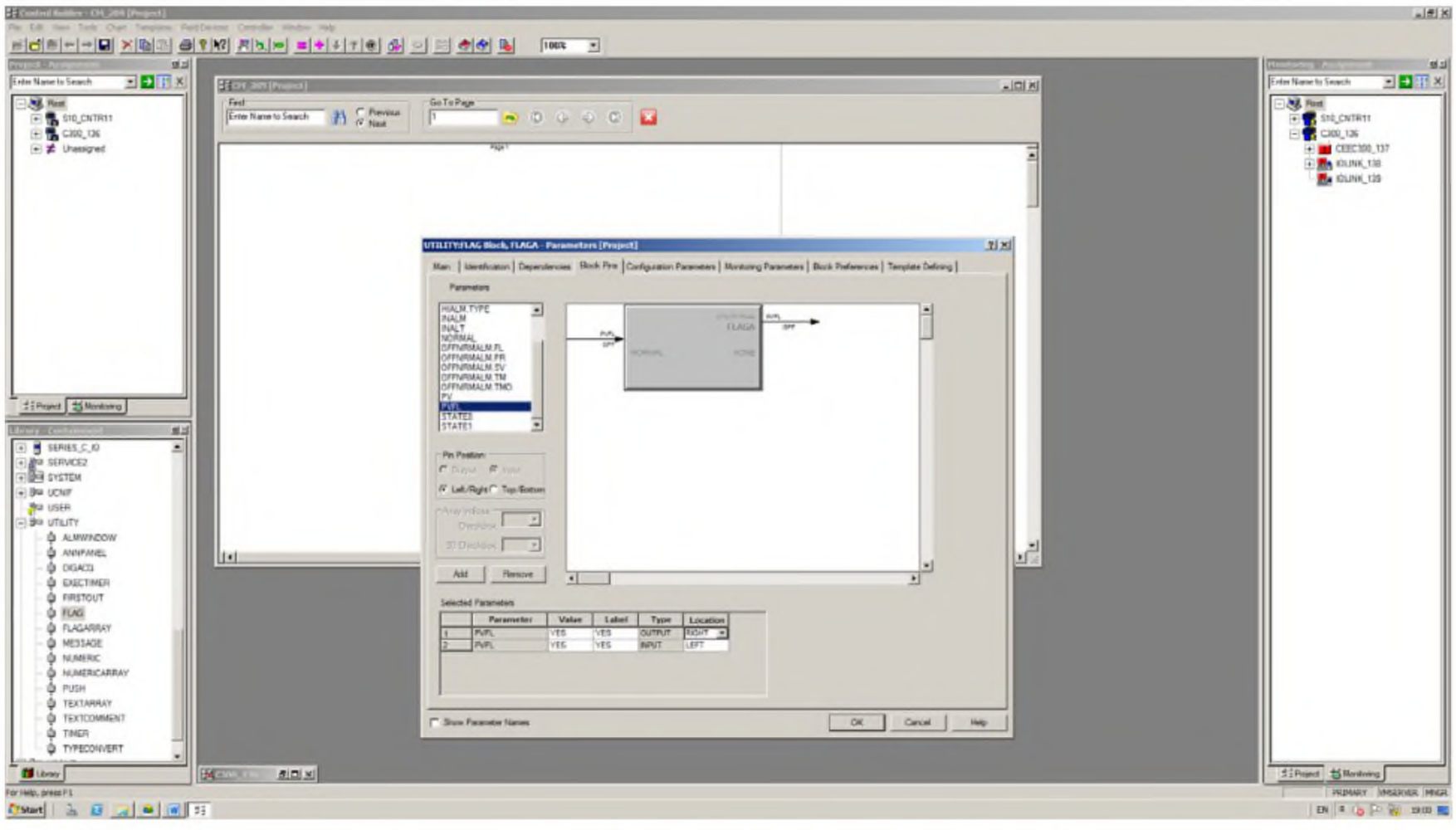


Рис. 7.8. Свойства блока FLAG

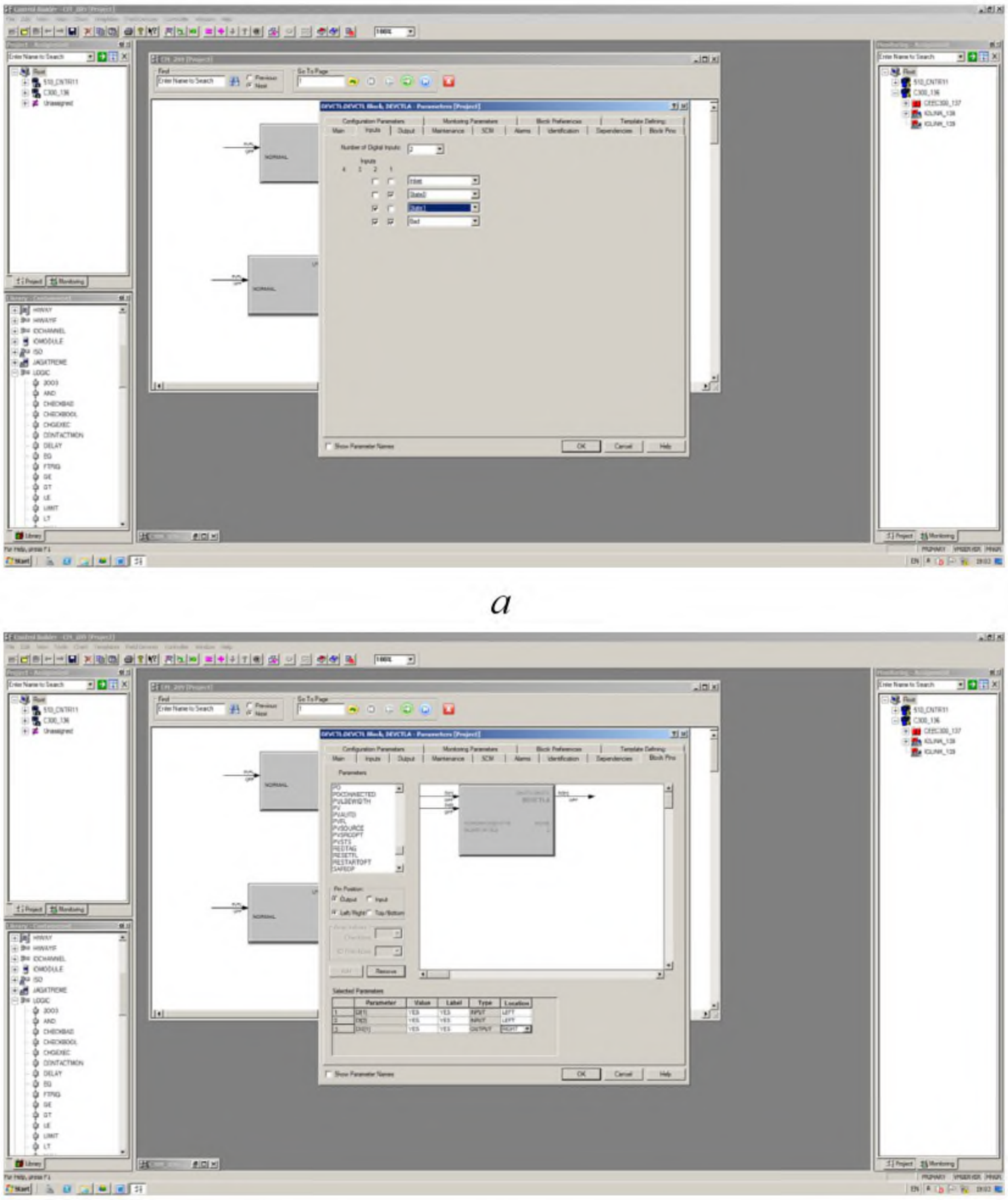
Далее в дереве элементов необходимо выбрать DEVCTL → DEVCTL и настроить этот элемент в соответствии с рис. 7.9, *а* и *б*. Во вкладке Main также необходимо задать имена состояний, как показано в табл. 7.2.

Далее в дереве элементов необходимо выбрать LOGIC → NOT и составить схему, приведенную на рис. 7.10.

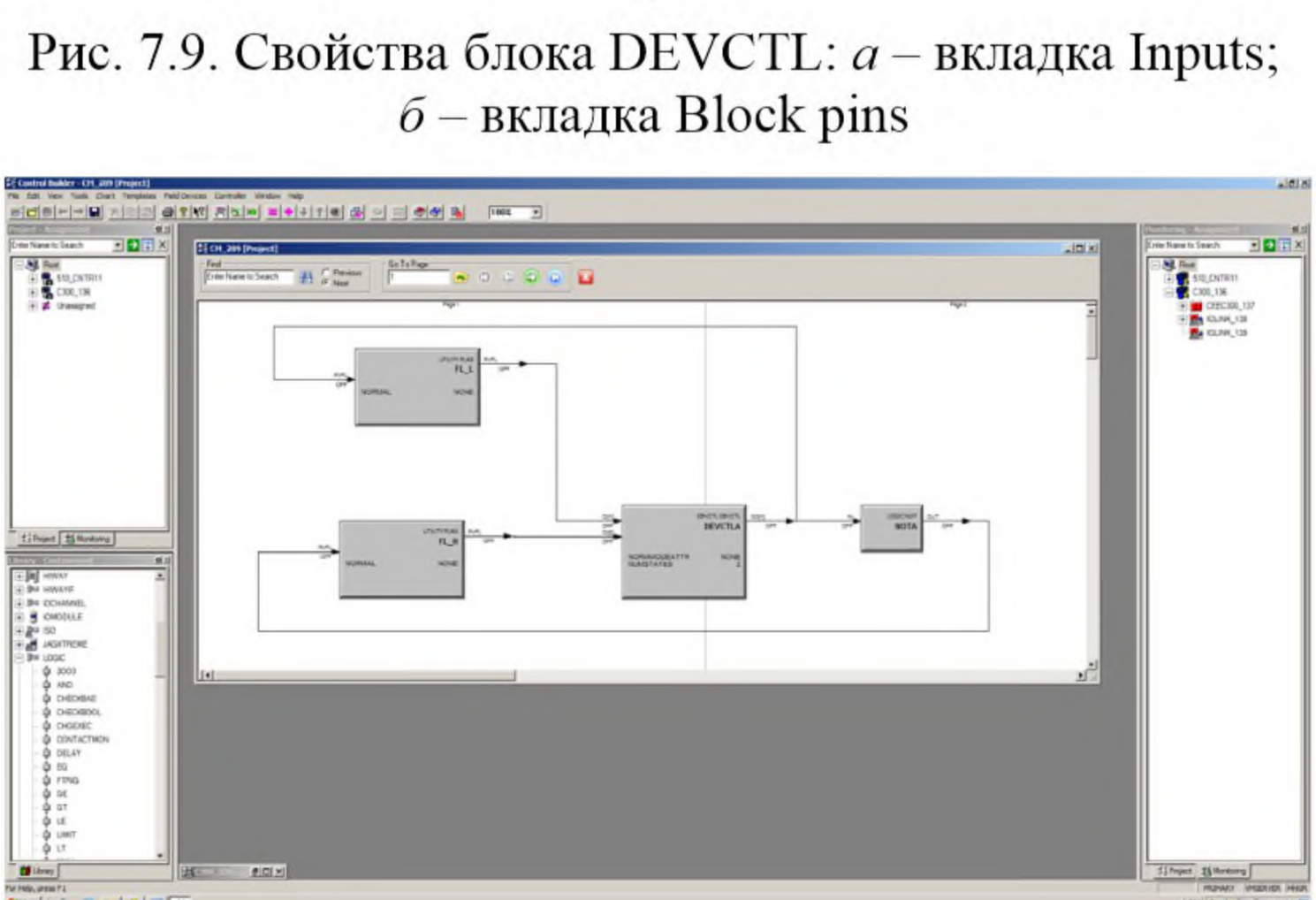
Таблица 7.2

Состояния концевых контактов клапана

FLAG_H	FLAG_L	Состояние
0	0	Переход из одного состояния в другое
0	1	Клапан закрыт
1	0	Клапан открыт
1	1	Ошибка



а



б

Рис. 7.9. Свойства блока DEVCTL: *а* – вкладка Inputs; *б* – вкладка Block pins

Рис. 7.10. Схема, созданная в Control modyle

7.1.2. Активация эмулятора контроллера

Закрыв рабочее окно с созданной схемой, необходимо добавить созданный проект в эмулятор контроллера C300_136. Для этого используют клавиши «=» и «+». Данную операцию производят в соответствии с рис. 7.11, выбрав созданный проект в меню Available modules и CEEC300_137 в Assigment information при помощи кнопки Assign.

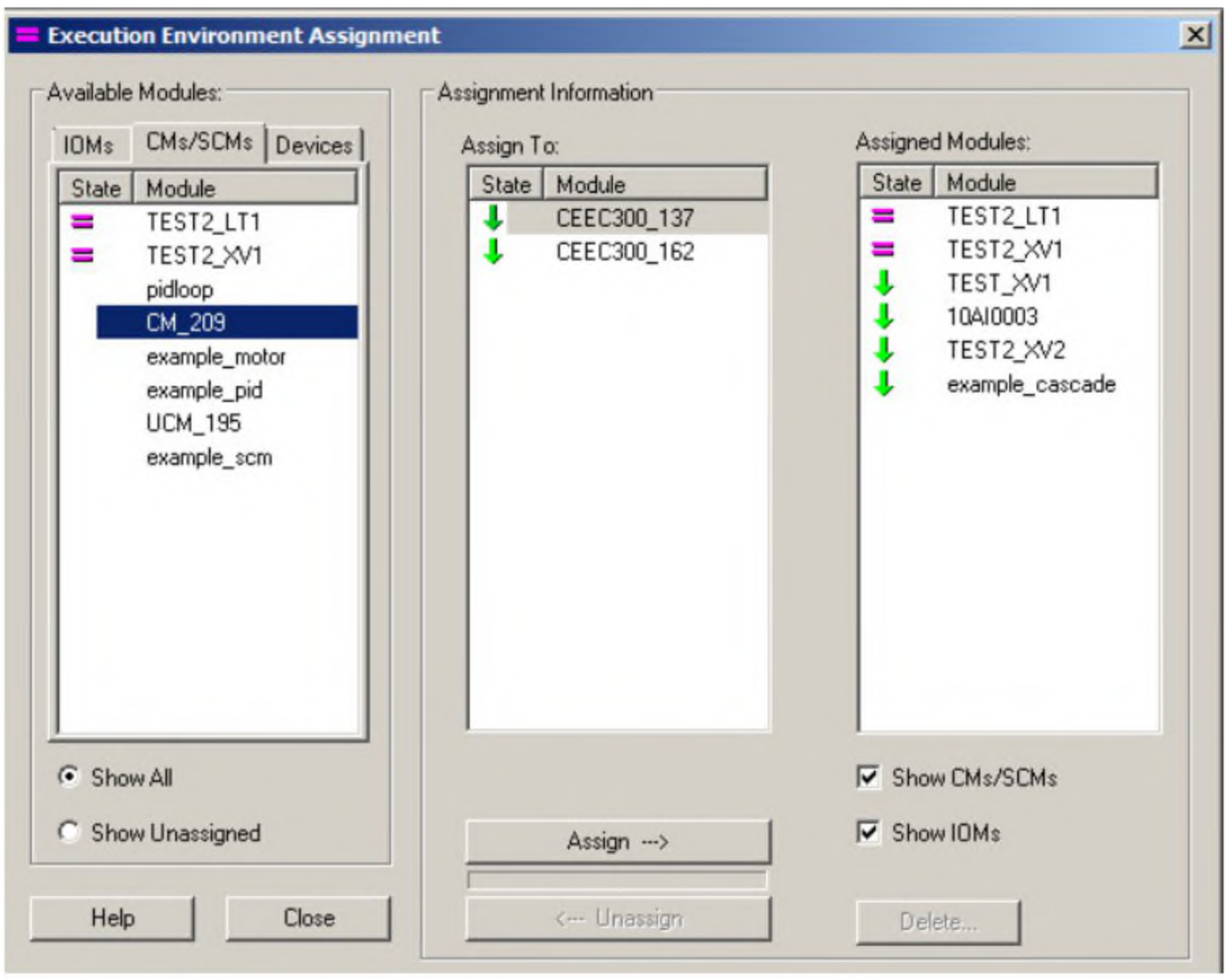


Рис. 7.11. Добавление проекта в эмулятор

Данный проект появится в дереве Monitoring. При этом все изменения производятся только в дереве Project, например, если нужно изменить в блоке DEVCTL имена состояний (вкладка Main).

Перед активацией эмулятора контроллера C300_136 необходимо в окне его настроек, вызываемом двойным щелчком мыши, открыть вкладку «Main» и убедиться, что у настройки «Load to simulation Enviroment» поставлена соответствующая галочка.

Далее в окне настройки модуля CEEC300_137 необходимо осуществить настройку в соответствии с рис. 7.12.

Далее нужно выбрать контроллер C300_136, нажать на панели инструментов клавишу «стрелка вниз», после чего появится окно, представленное на рис. 7.13.

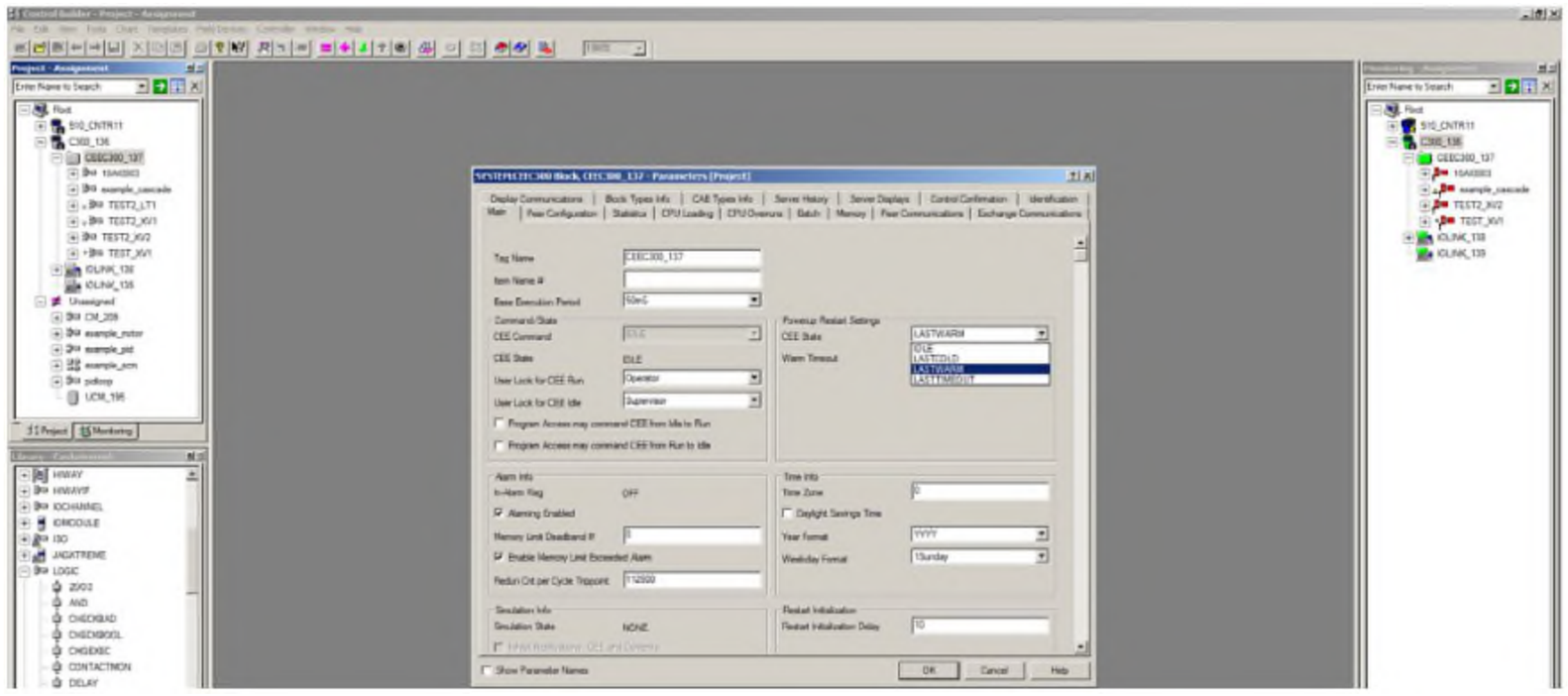


Рис. 7.12. Настройки модуля CEEC300_137

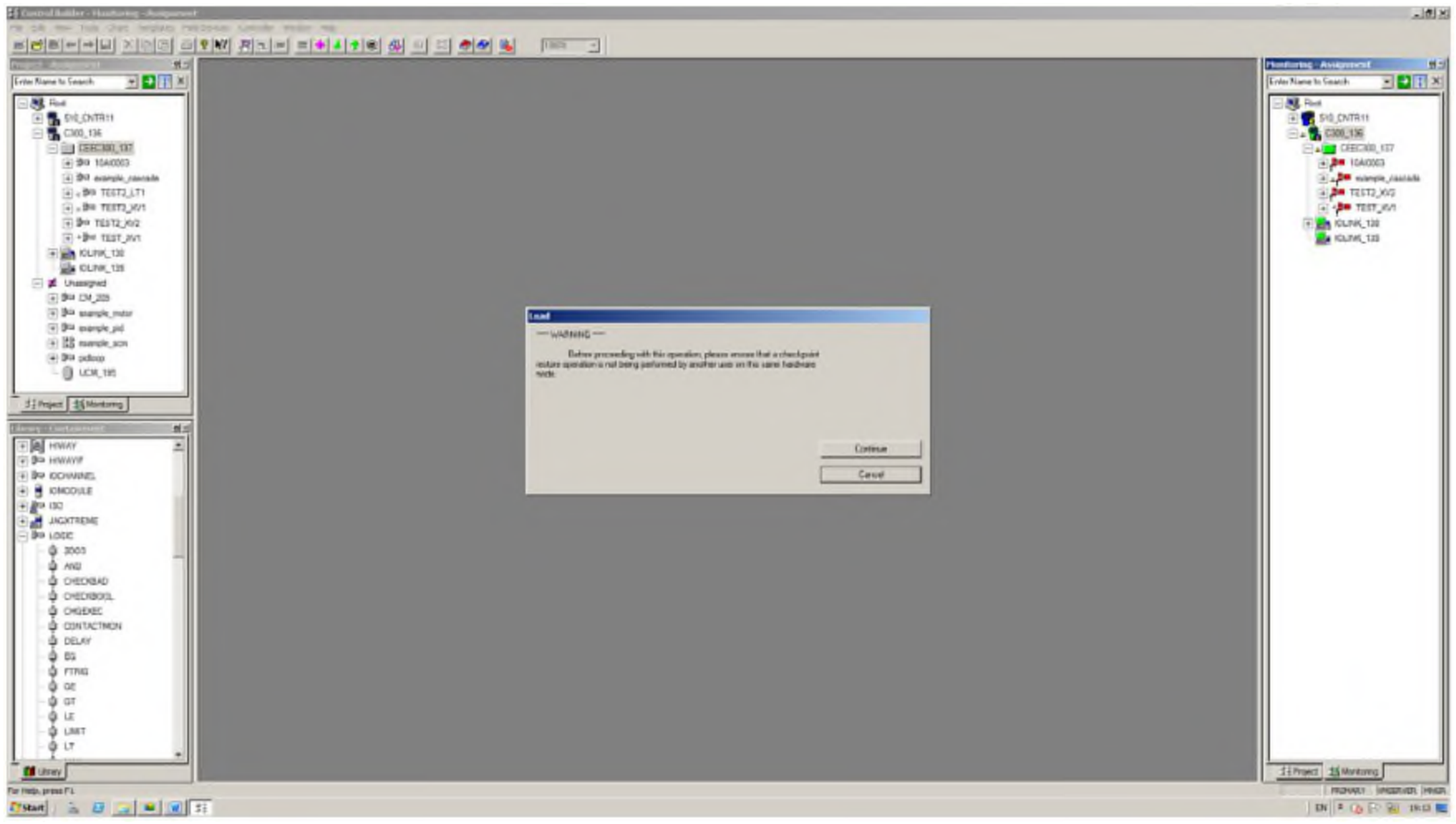


Рис. 7.13. Окно Load

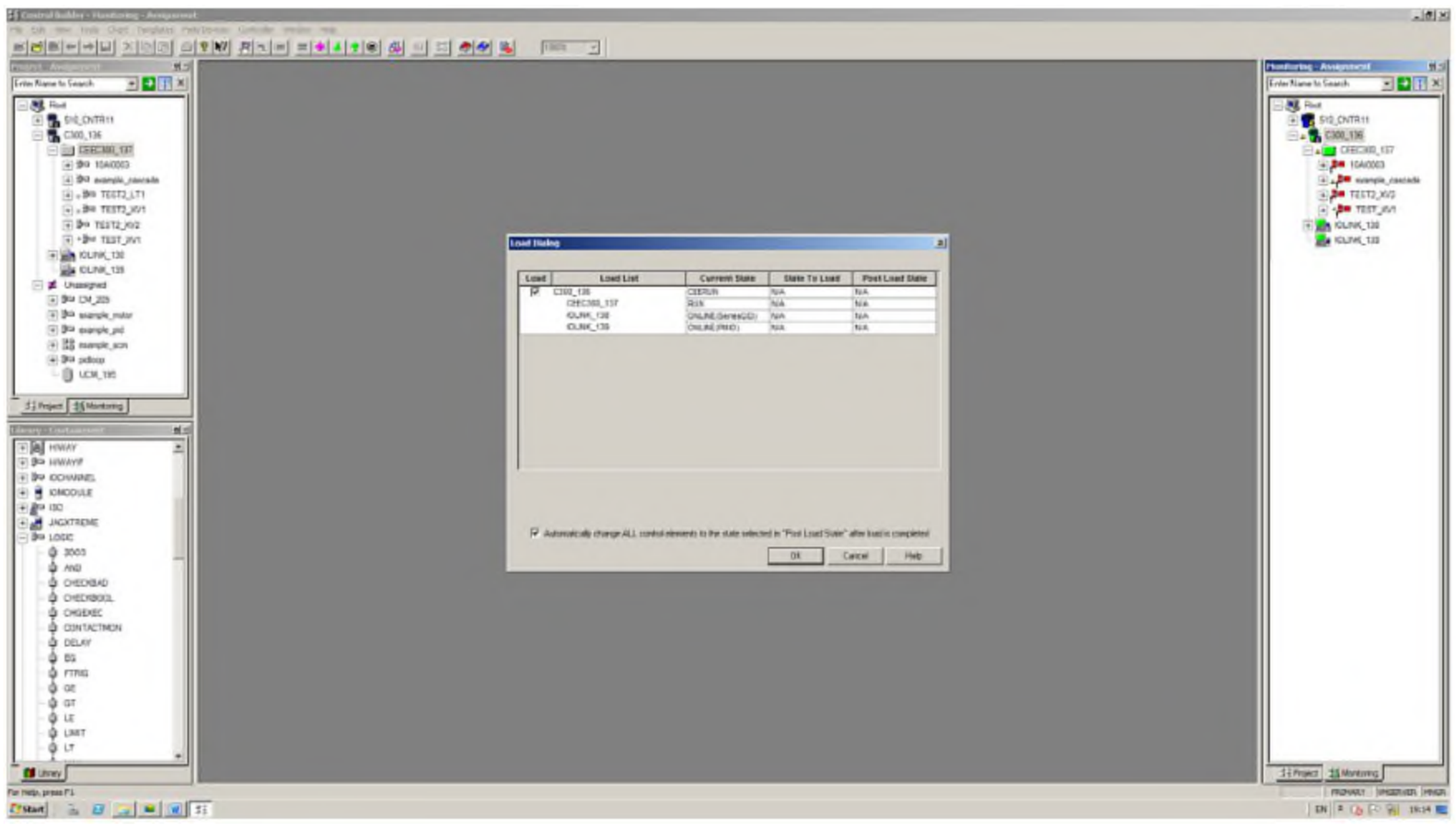


Рис. 7.14. Окно Load Dialog

Далее необходимо нажать Continue. В появившемся окне (рис. 7.14) необходимо поставить соответствующие галочки (рис. 7.15) и нажать Ok.

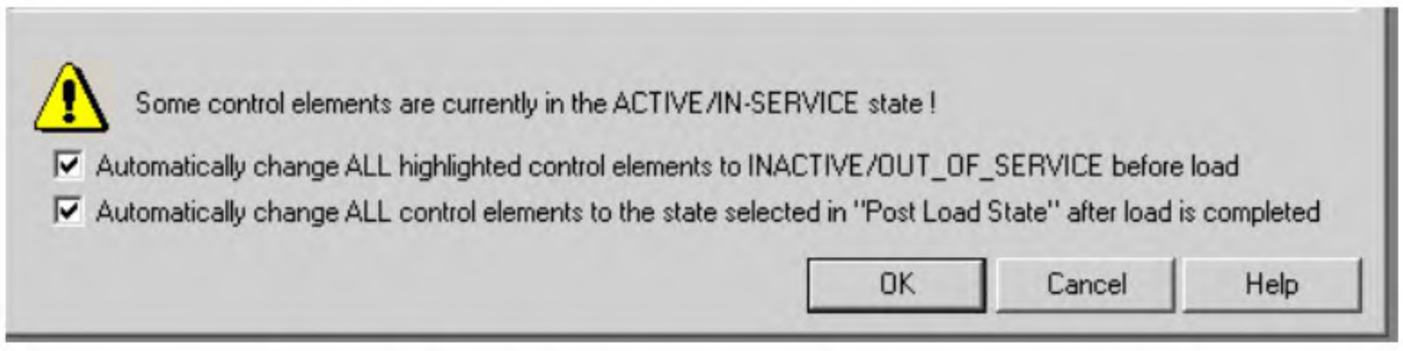


Рис. 7.15. Окно-предупреждение

Также для запуска эмулятора может потребоваться операция, приведенная на рис. 7.16.

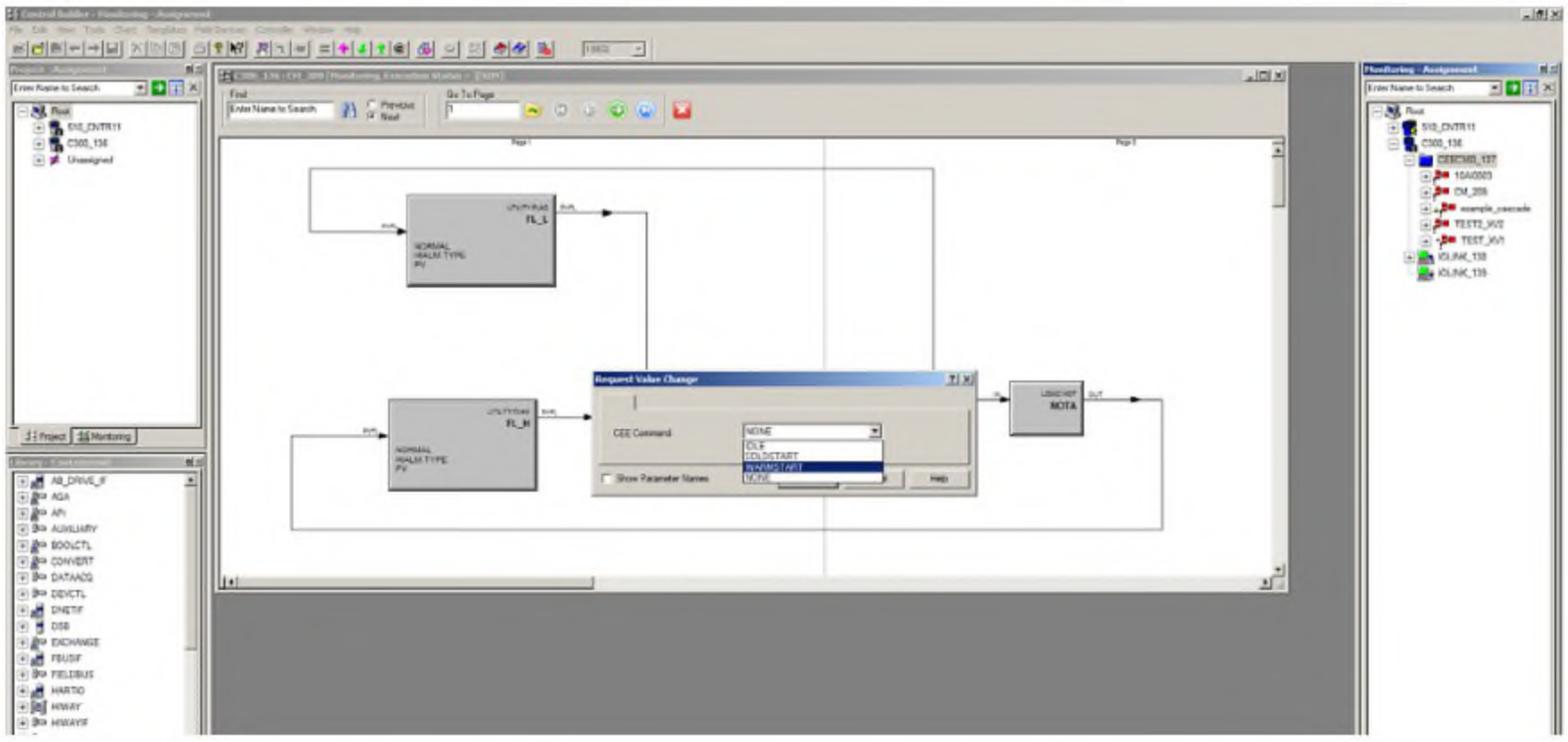


Рис. 7.16. Запуск схемы в работу

Активировать эмулятор можно при помощи команды Change state или Activate в контекстном меню эмулятора (рис. 7.17).

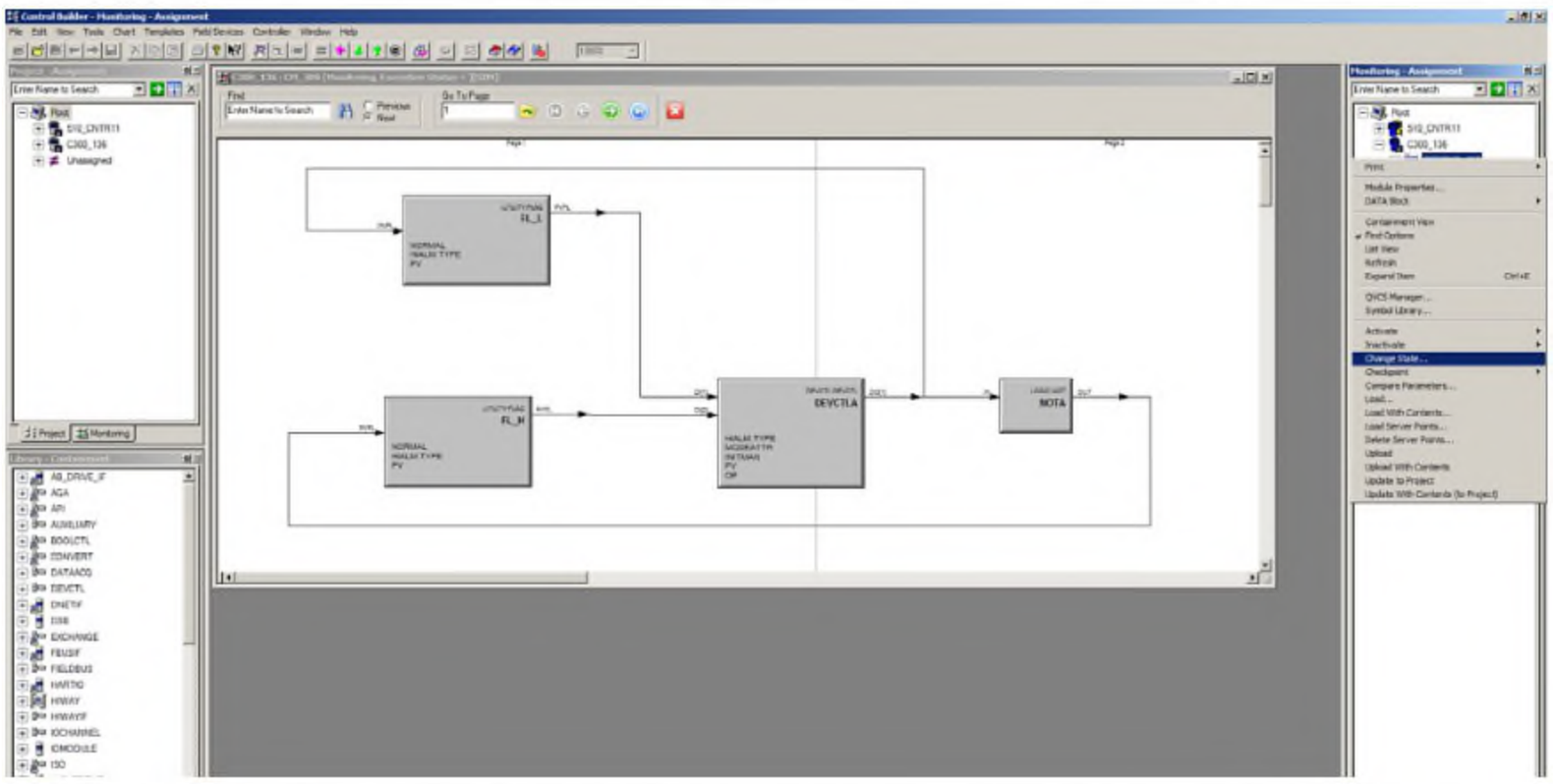


Рис. 7.17. Активирование эмулятора

Во время работы эмулятора можно менять состояние блока DEVCTL вызвав двойным щелчком мыши на его параметре OP окно, представленное на рис. 7.18.

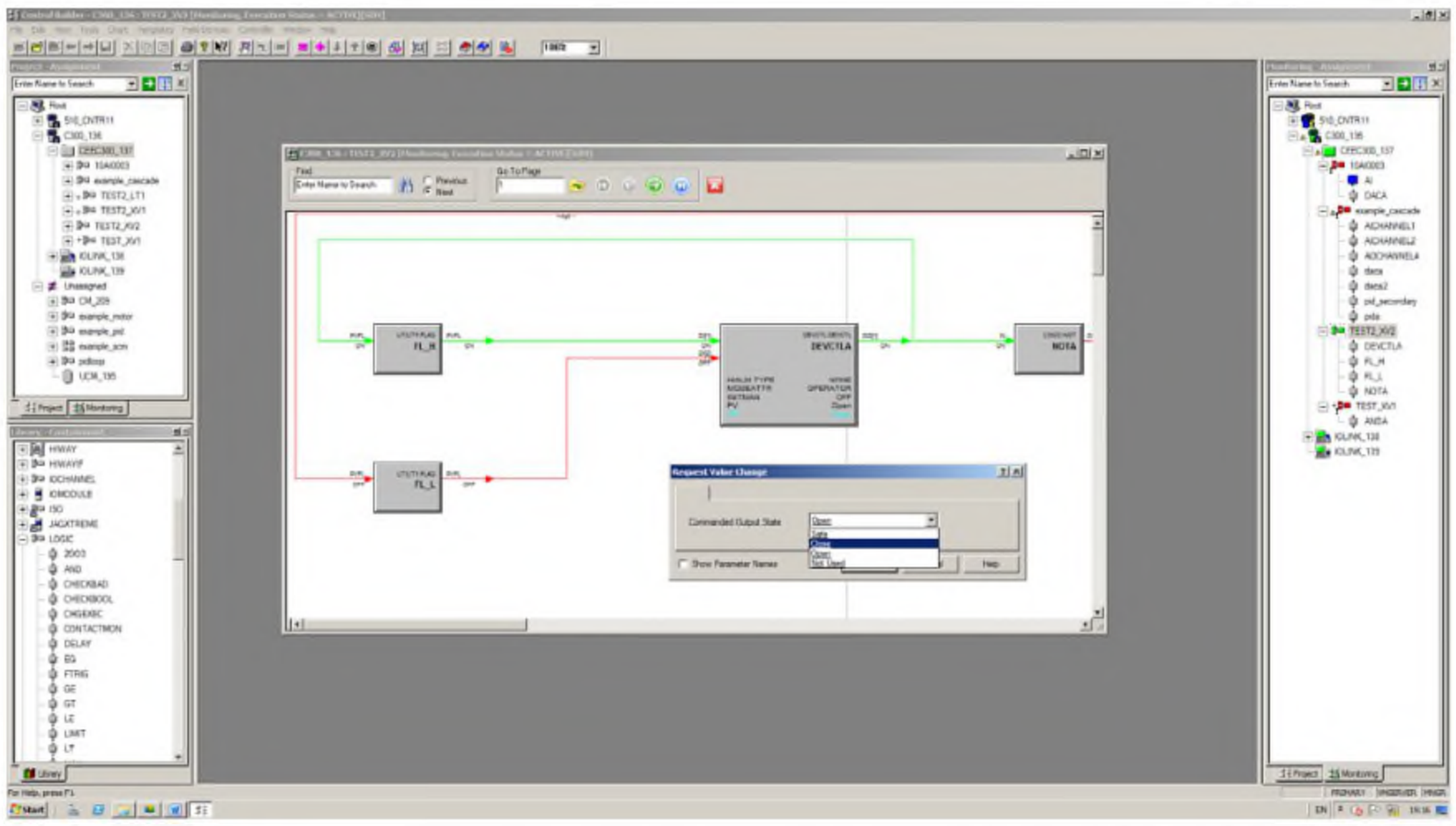


Рис. 7.18. Изменение задания OP в блоке DEVCTL

7.1.3. Разработка мнемосхемы управления клапаном

Для создания мнемосхемы необходимо запустить Configuration Studio → Display → Create new Normal display. Появится поле создания мнемосхем (рис. 7.19).

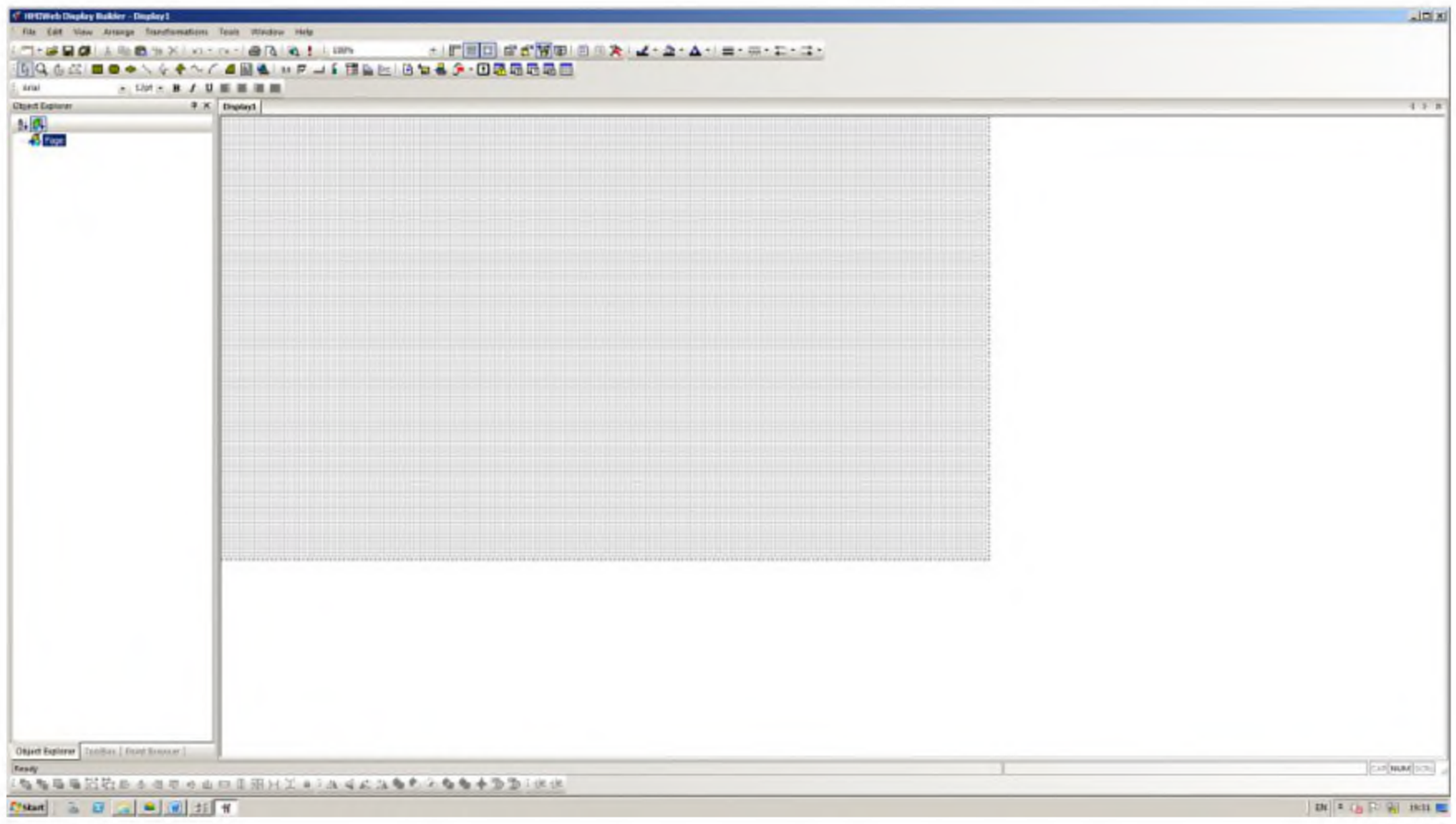


Рис. 7.19. Окно редактора мнемосхем

На панели инструментов выбрать элемент Combobox и произвести его настройку в соответствии с рис. 7.20 и 7.21.

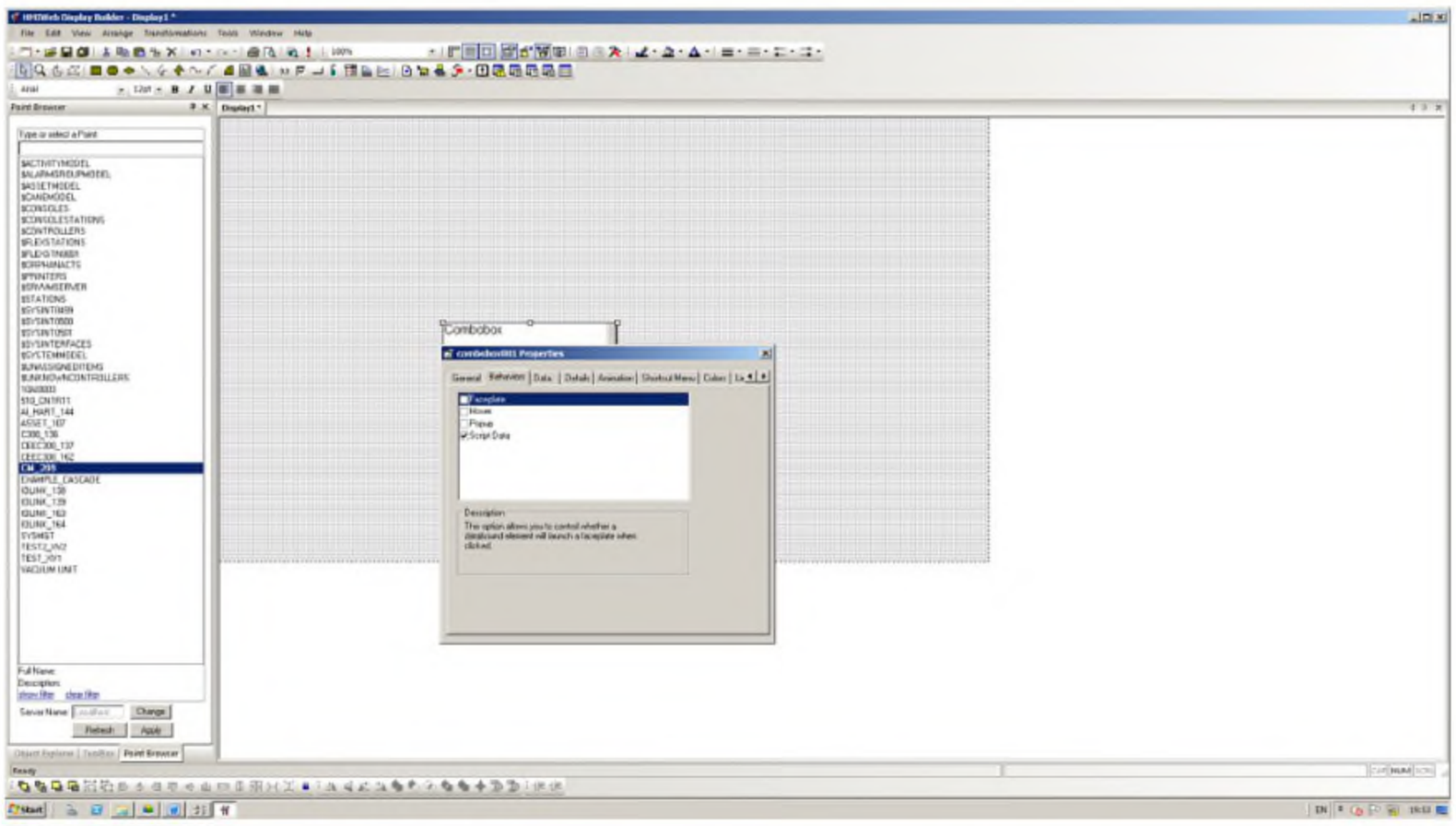


Рис. 7.20. Свойства элемента Combobox

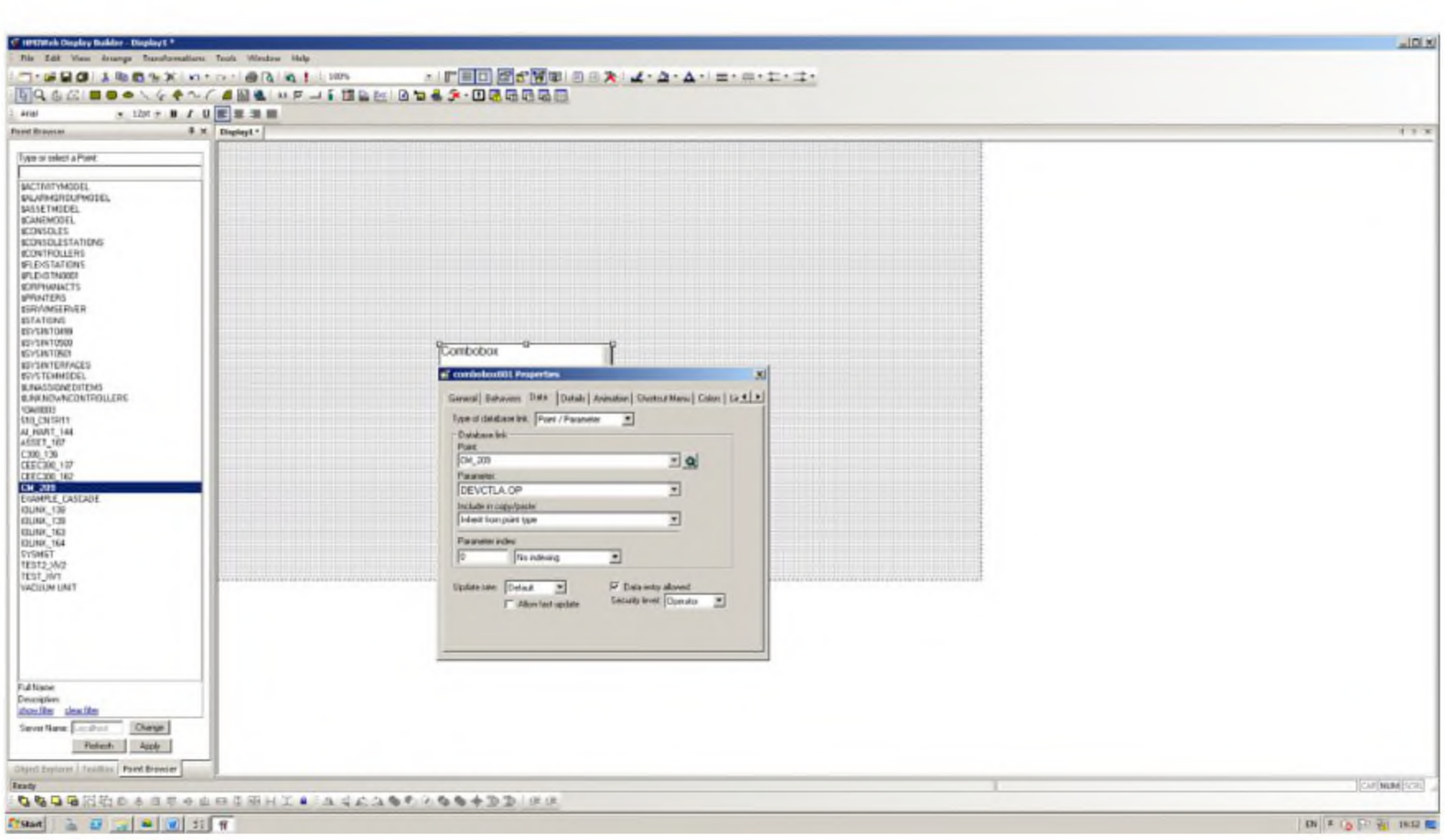


Рис. 7.21. Привязка элемента Combobox

Все созданные мнемосхемы сохраняются в директории, указанной на рис. 7.22.

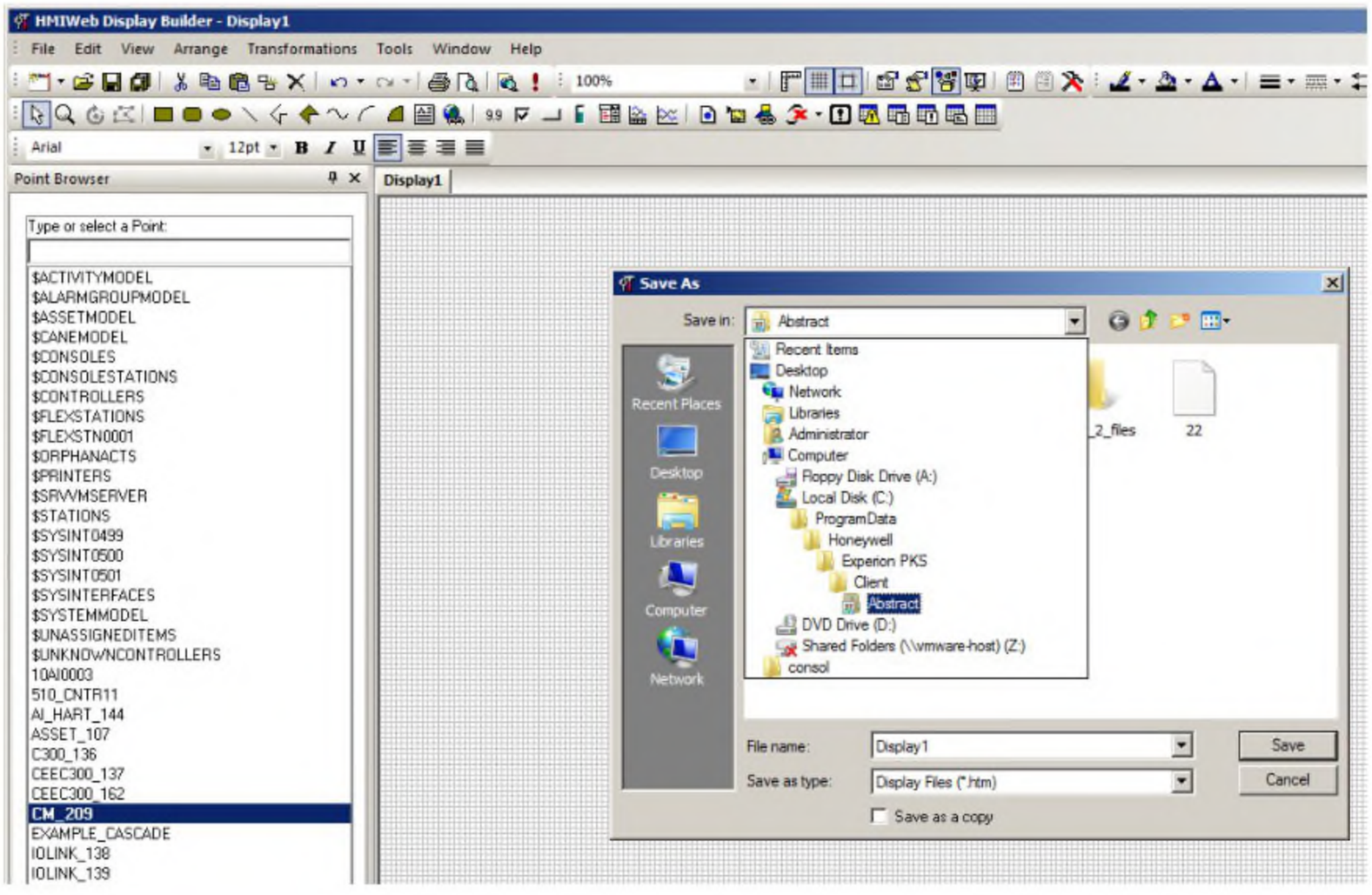


Рис. 7.22. Дерево каталогов для сохранения

7.1.4. Запуск мнемосхемы

Для запуска мнемосхемы необходимо проверить состояние работы сервера (рис. 7.23 и 7.24).

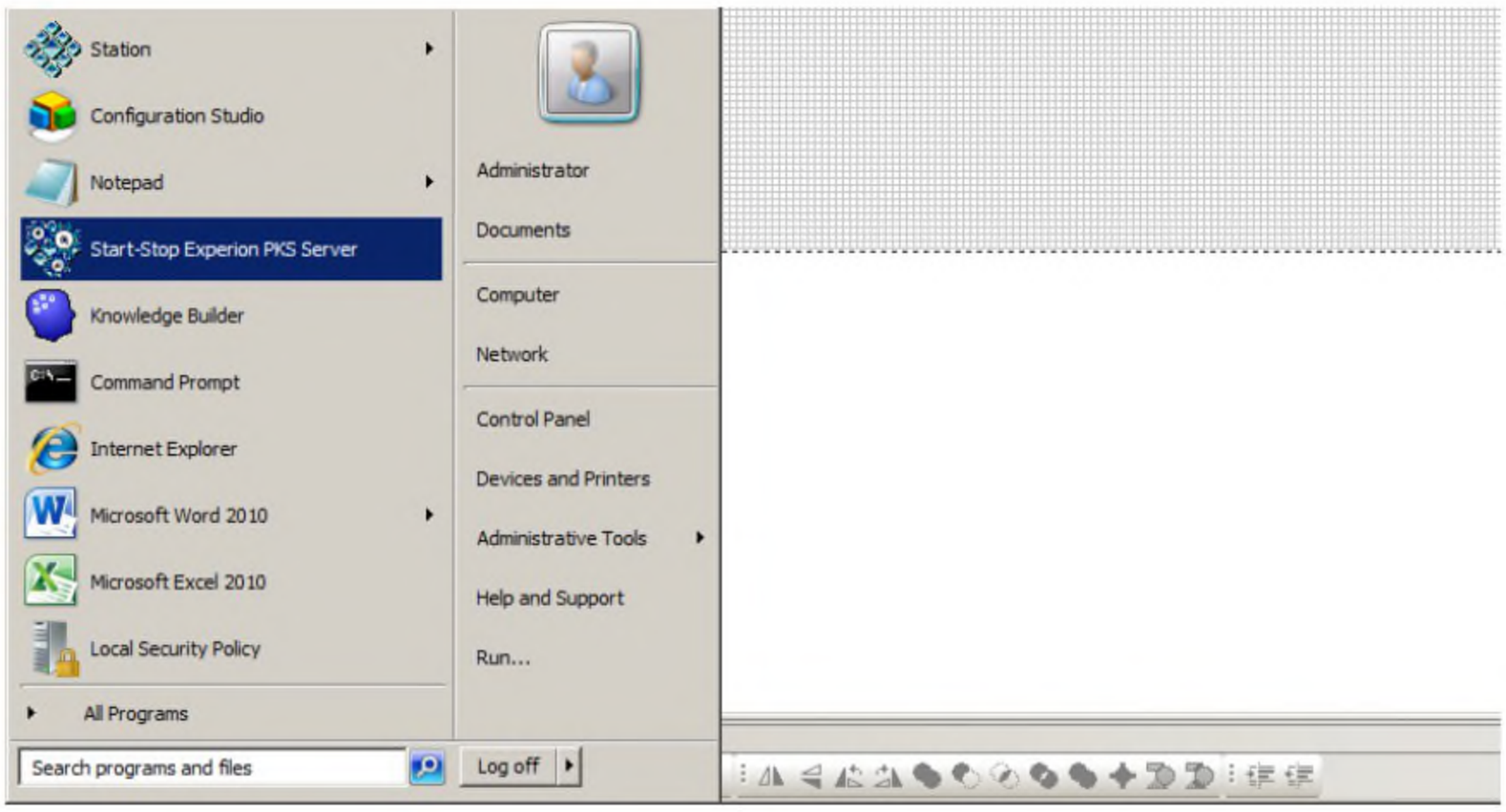


Рис. 7.23. Открытие окна проверки состояния работы сервера

Рис. 7.24. Окно отображения состояния сервера

Далее запускаем Пуск → Station (рис. 7.25).

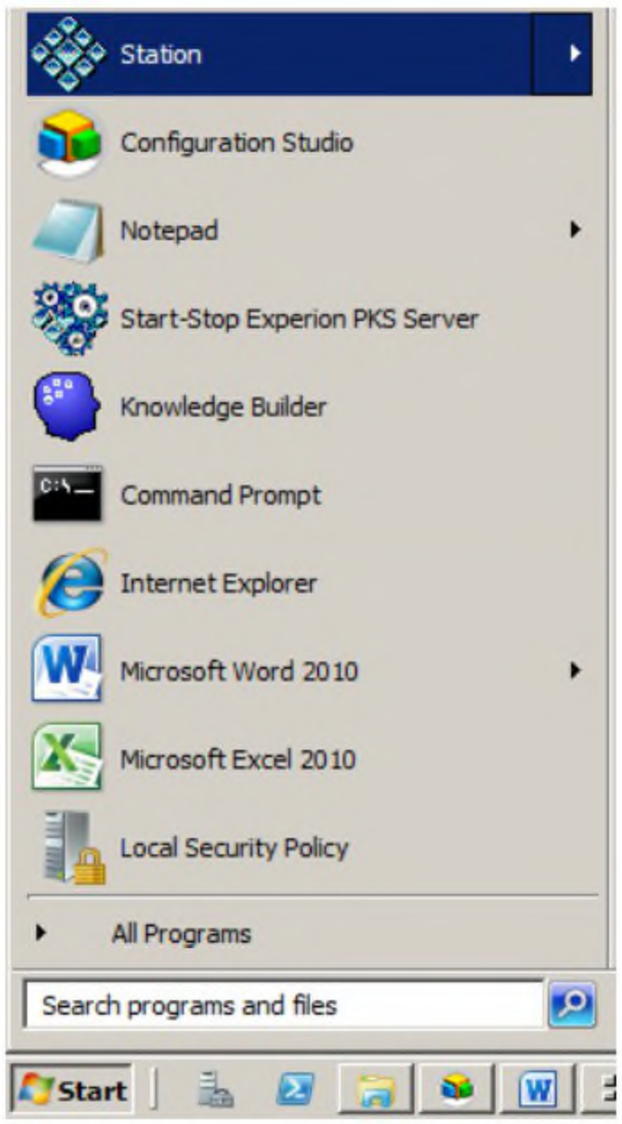


Рис. 7.25. Запуск станции оператора

В появившемся окне необходимо нажать Connect (рис. 7.26 и 7.27).

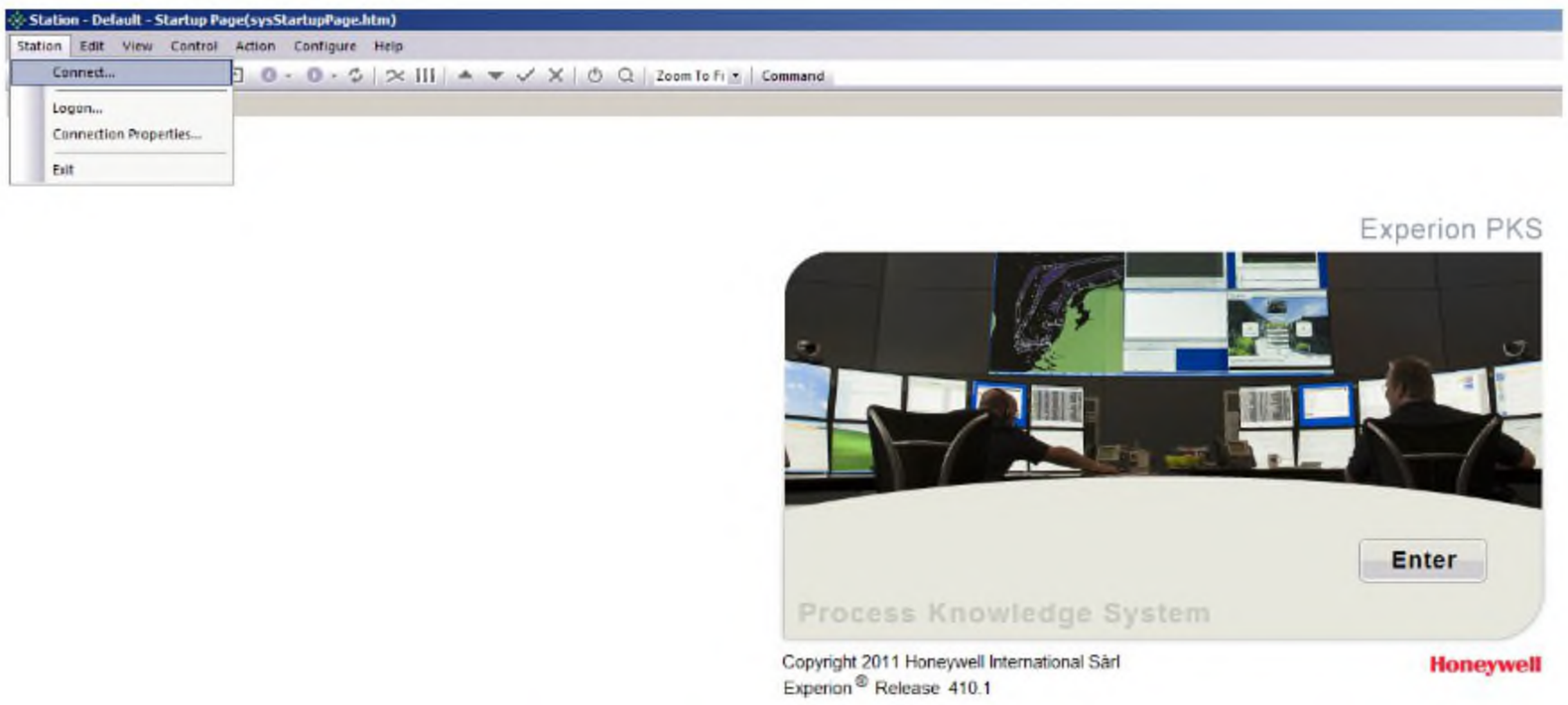


Рис. 7.26. Подключение к серверу

Для запуска мнемосхемы нужно написать ее имя в поле Command.

Примечание: Все динамические элементы необходимо настроить таким образом, чтобы они работали под учетной записью Operator. При этом для работы станции необходимо ввести пароль «mngr».

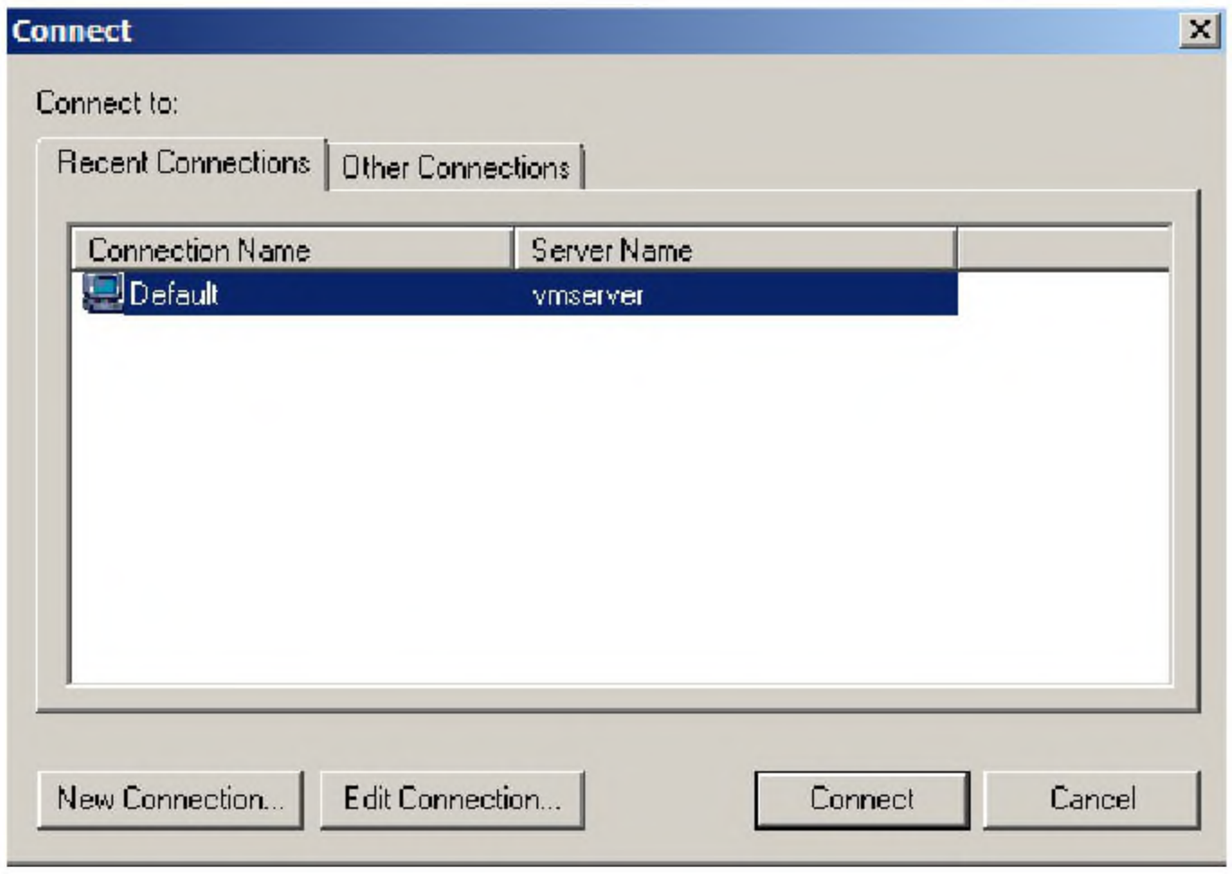


Рис. 7.27. Осно Connect

7.1.5. Создание шейпа

Вызов существующих шейпов (динамических элементов мнемосхемы) осуществляется при помощи клавиши «  », показанной на рис. 7.28. Все шейпы хранятся в определенной директории (рис. 7.29 и 7.30).

Рис. 7.28. Вызов галереи шейпов

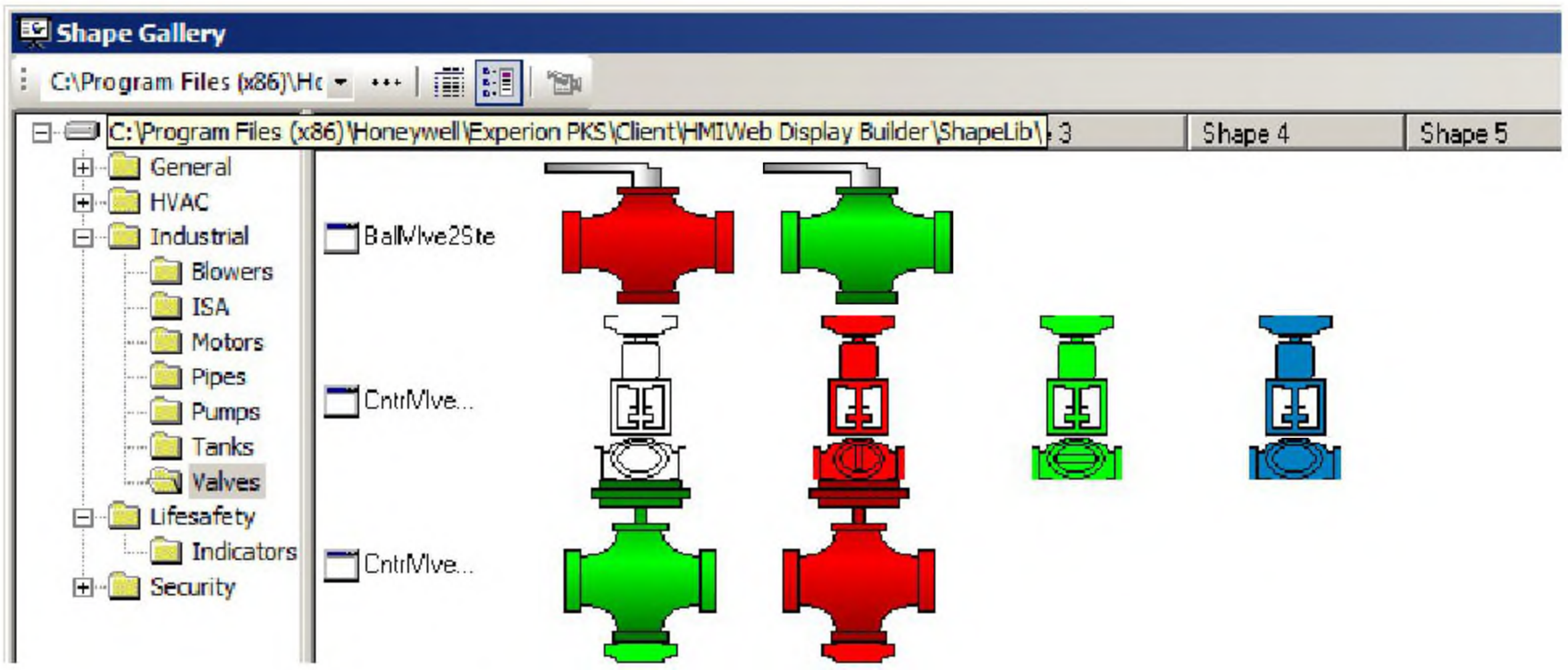


Рис. 7.29. Галерея шейпов

Для создания собственного шейпа необходимо выбрать следующий шейп (см. рис. 30).

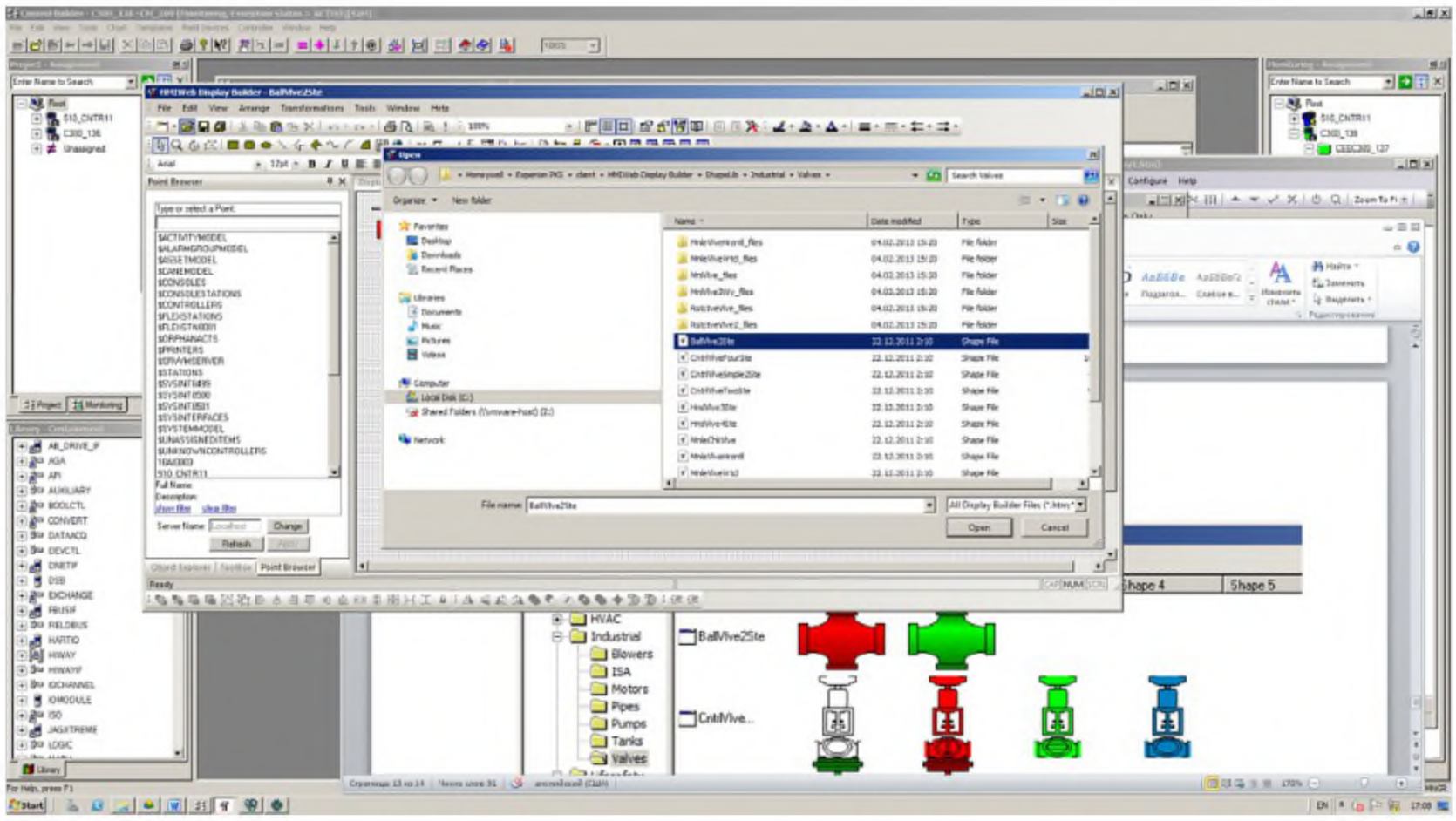


Рис. 7.30. Открытие шейпа

Используя имеющийся шейп, необходимо создать свой, изменив имеющийся следующим образом (рис. 7.31).

Рис. 7.31. Внутренняя развертка шейпа

Сохранить созданный шейп необходимо в папку «1» на рабочем столе, а затем скопировать сохраненный шейп из папки «1» в необходимую директорию, как показано на рис. 7.32.

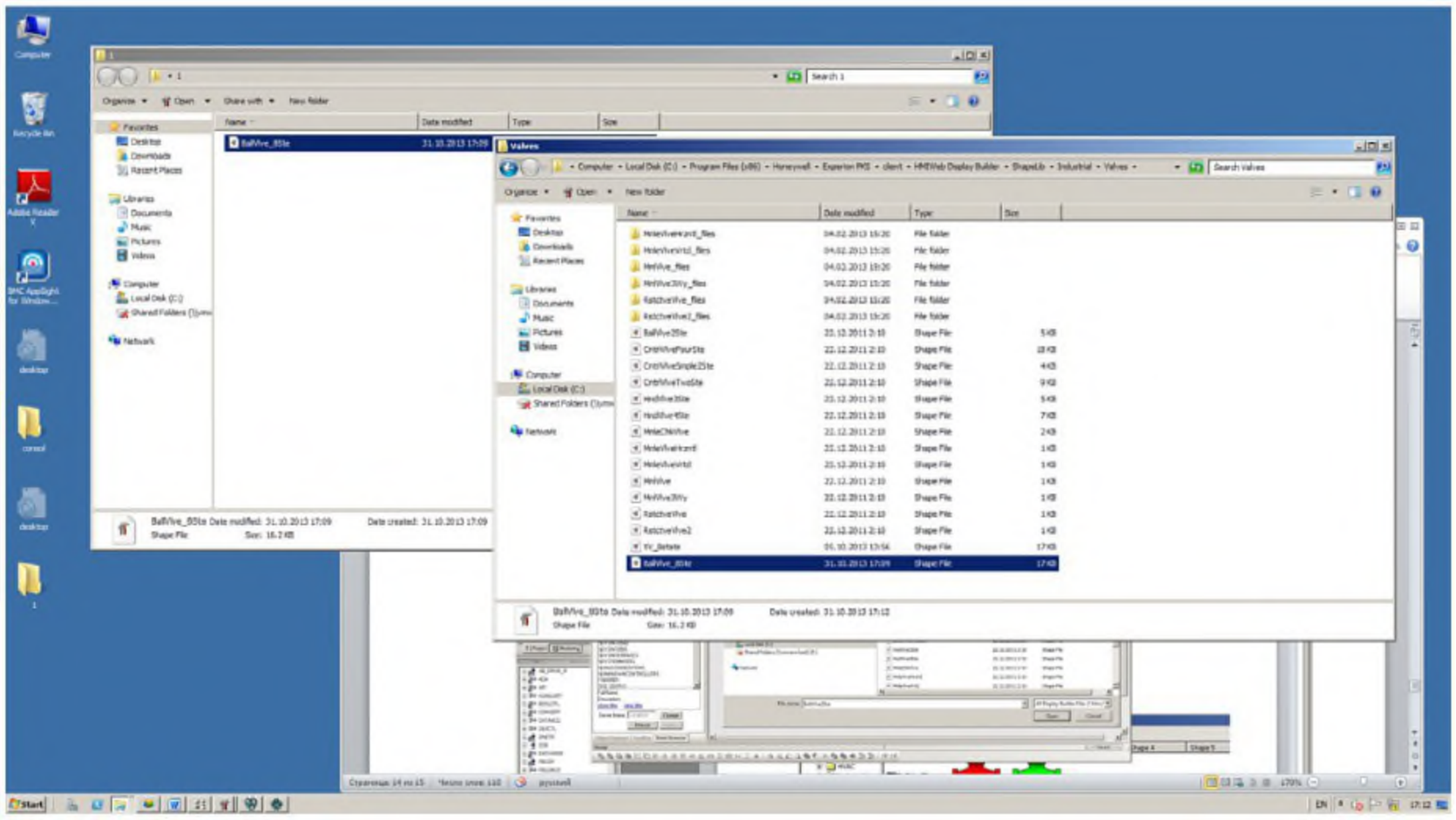


Рис. 7.32. Директория хранения шейпов

Для проверки работы мнемосхемы необходимо выполнить вышеуказанные пункты.

Далее добавляем созданный шейп (рис. 7.33), сохраняем мнемосхему в необходимую директорию.

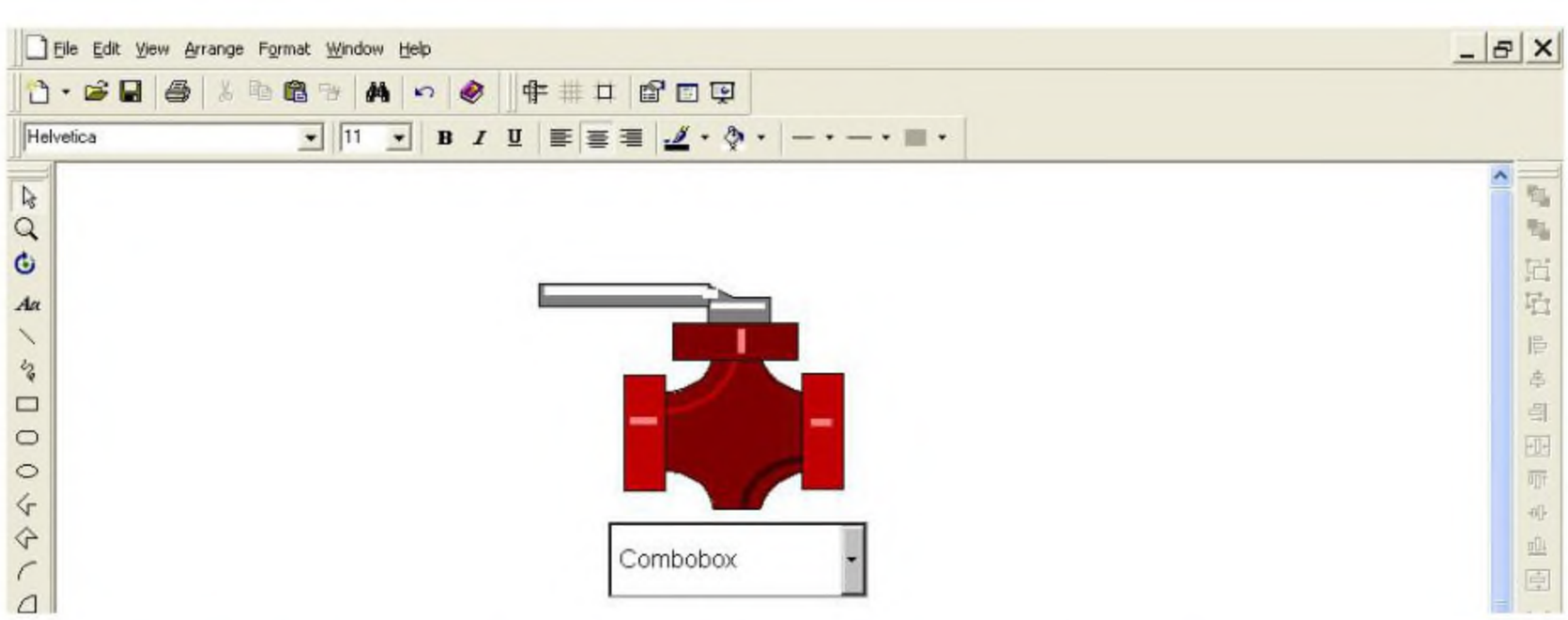


Рис. 7.33. Мнемосхема для управления клапаном-отсекателем

Работа мнемосхемы осуществляется, как описано выше, на станции оператора.

7.2. Алгоритмизация системы управления уровнем жидкости в технологическом резервуаре

На рис. 7.34 представлена подпрограмма TEST2_LT1, отвечающая за реализацию изменения уровня жидкости в резервуаре.

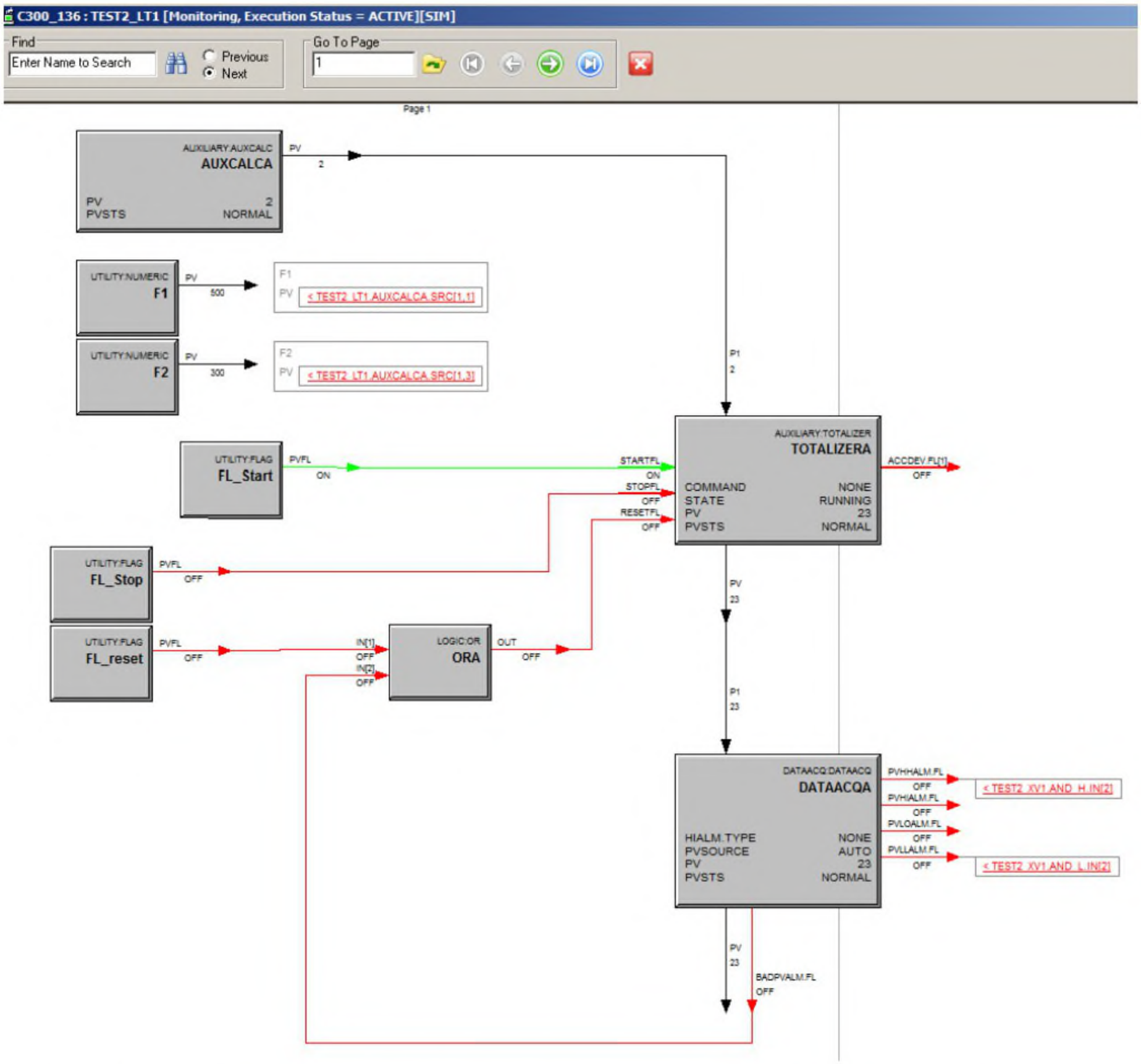


Рис. 7.34. Программа эмуляции изменения уровня жидкости в технологическом резервуаре

Текущее значение уровня жидкости описывается выражением

$$L = \frac{F_1 - F_2}{S},$$

где F_1 – расход жидкости на входе резервуара; F_2 – расход жидкости на выходе из резервуара; S – площадь поперечного сечения резервуара.

Программная реализация данного выражения представлена на рис. 7.35. Настройки блоков, использующихся в программе, представлены на рис. 7.36 и 7.37. Блок Dataacqa отвечает за значения LL, L, H, HH, как показано рис. 7.38.

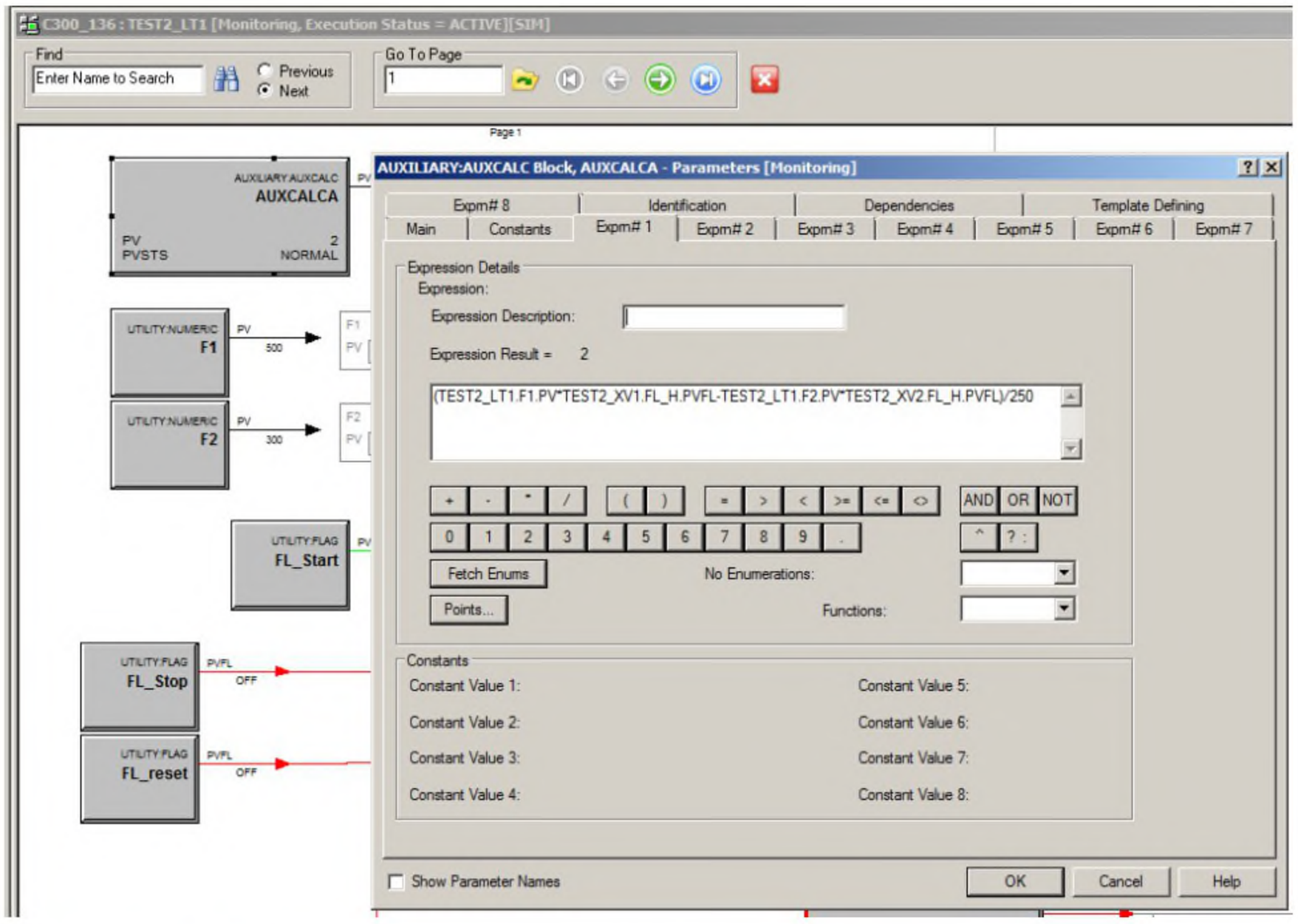


Рис. 7.35. Задание математической модели процесса изменения уровня жидкости в технологическом резервуаре

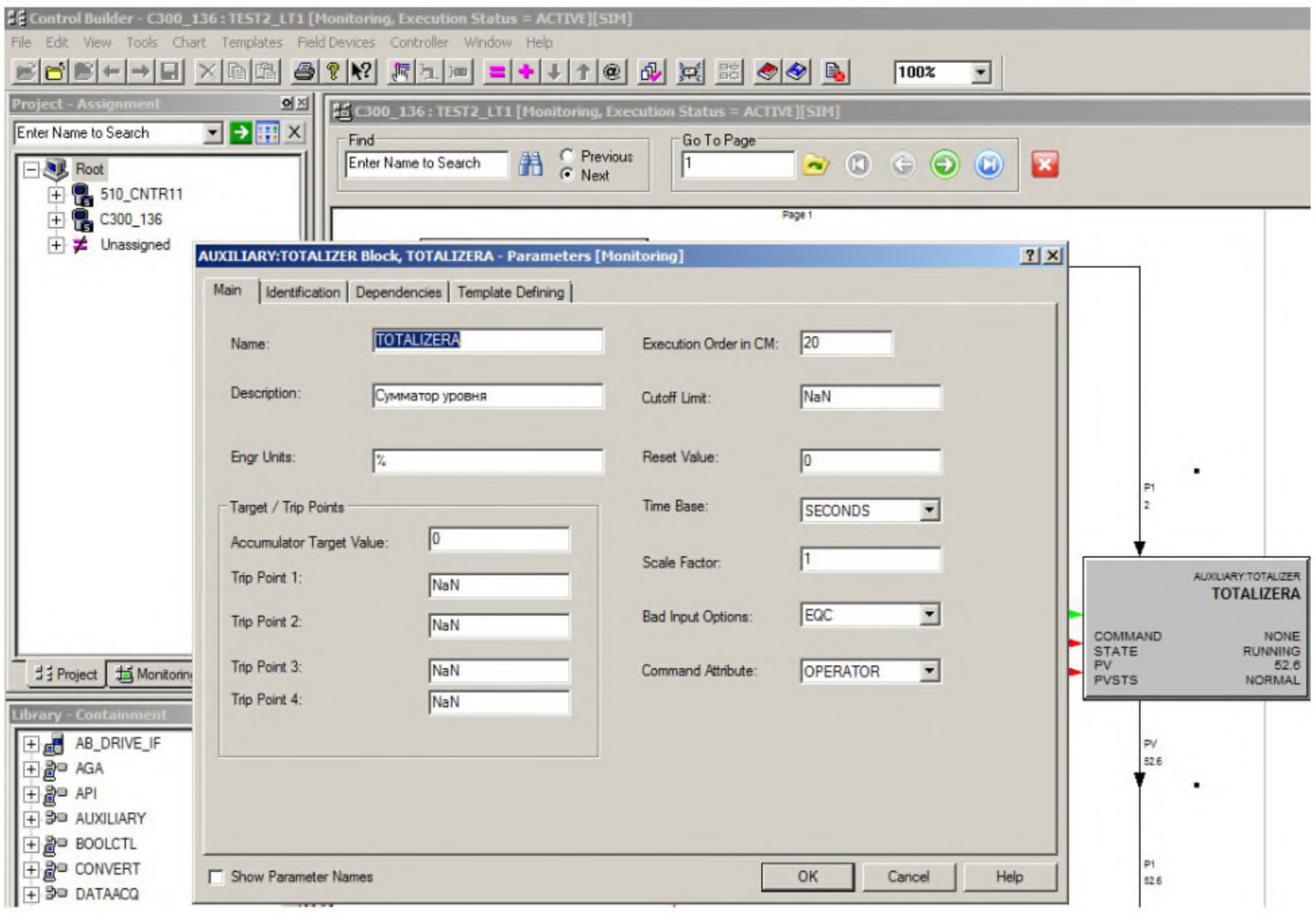


Рис. 7.36. Настройки блока Totalizer

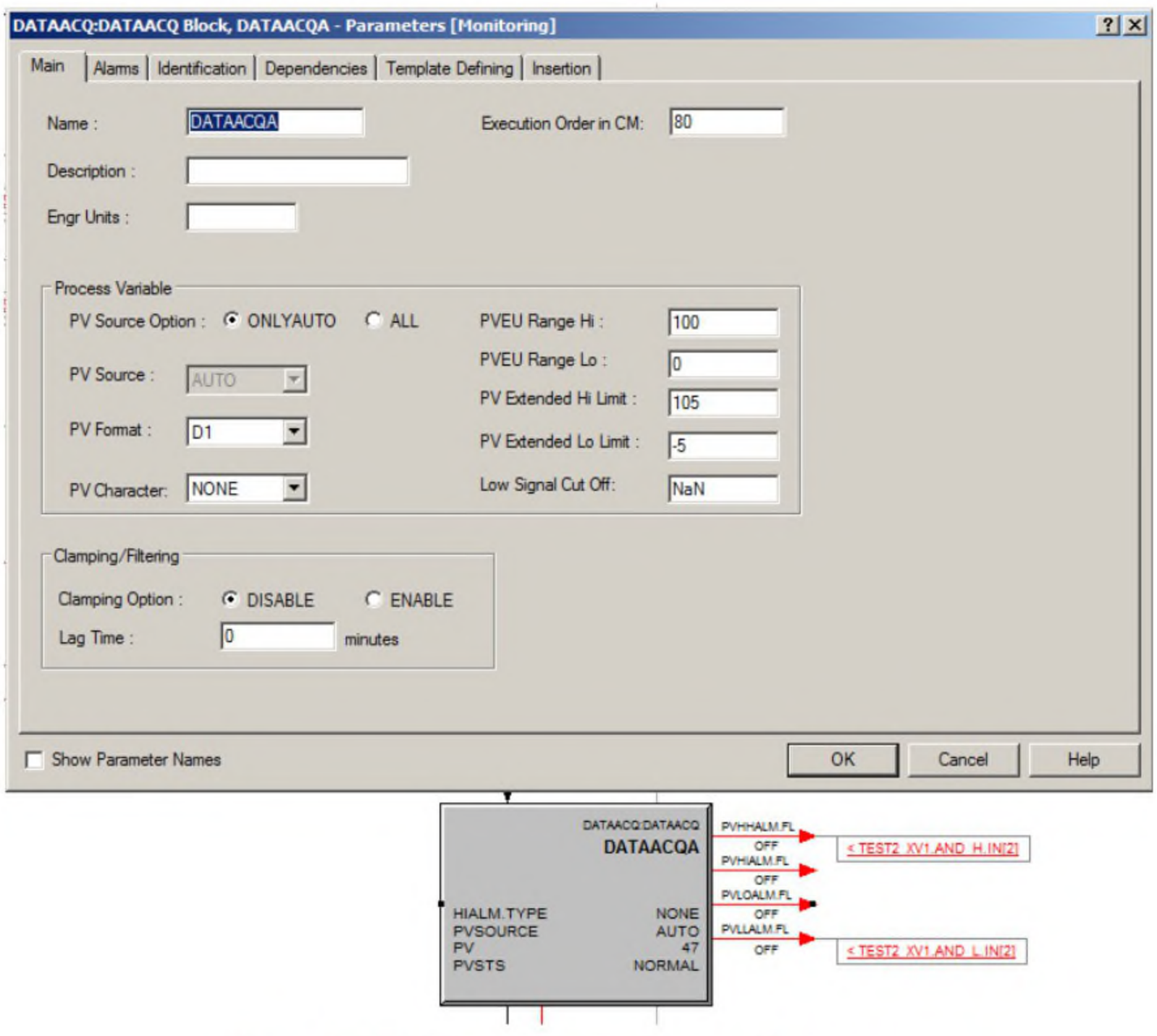


Рис. 7.37. Настройки блока Dataacqa

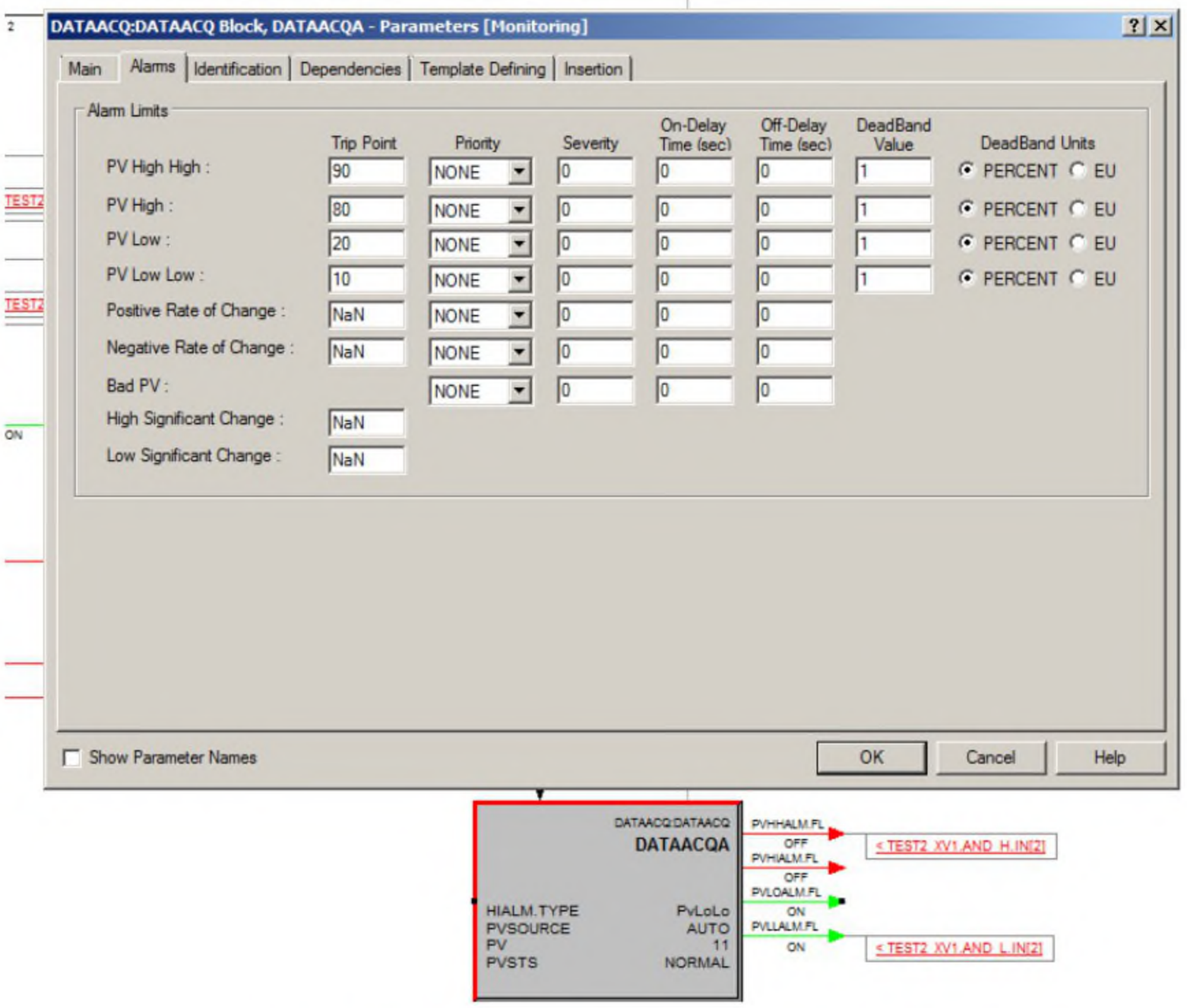


Рис. 7.38. Настройки блока Dataacqa

На рис. 7.39 представлена программа управления клапаном в линии налива TEST2_XV1, на рис. 7.40 – программа управления клапаном в линии слива TEST2_XV2.

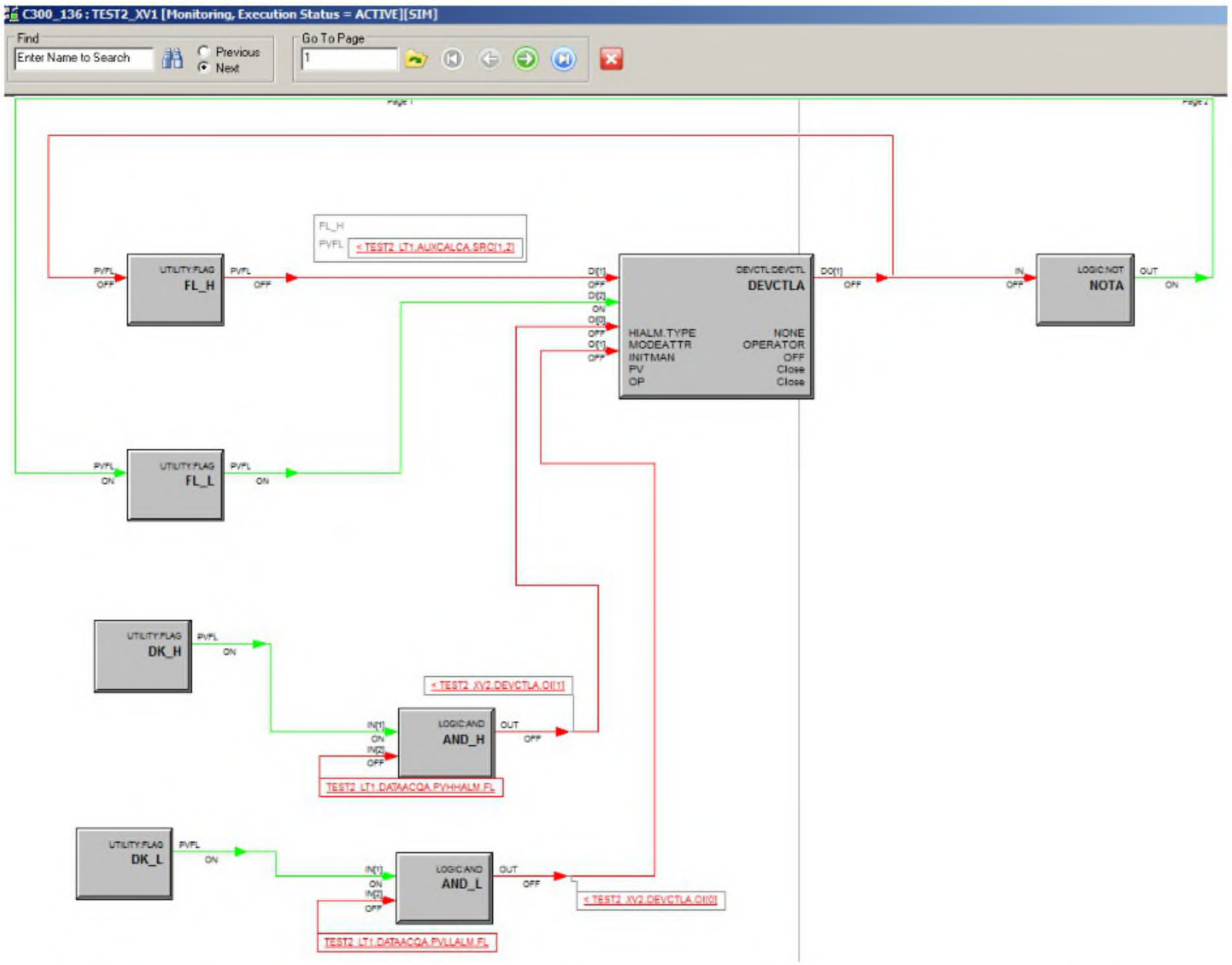


Рис. 7.39. Программа управления клапаном в линии налива

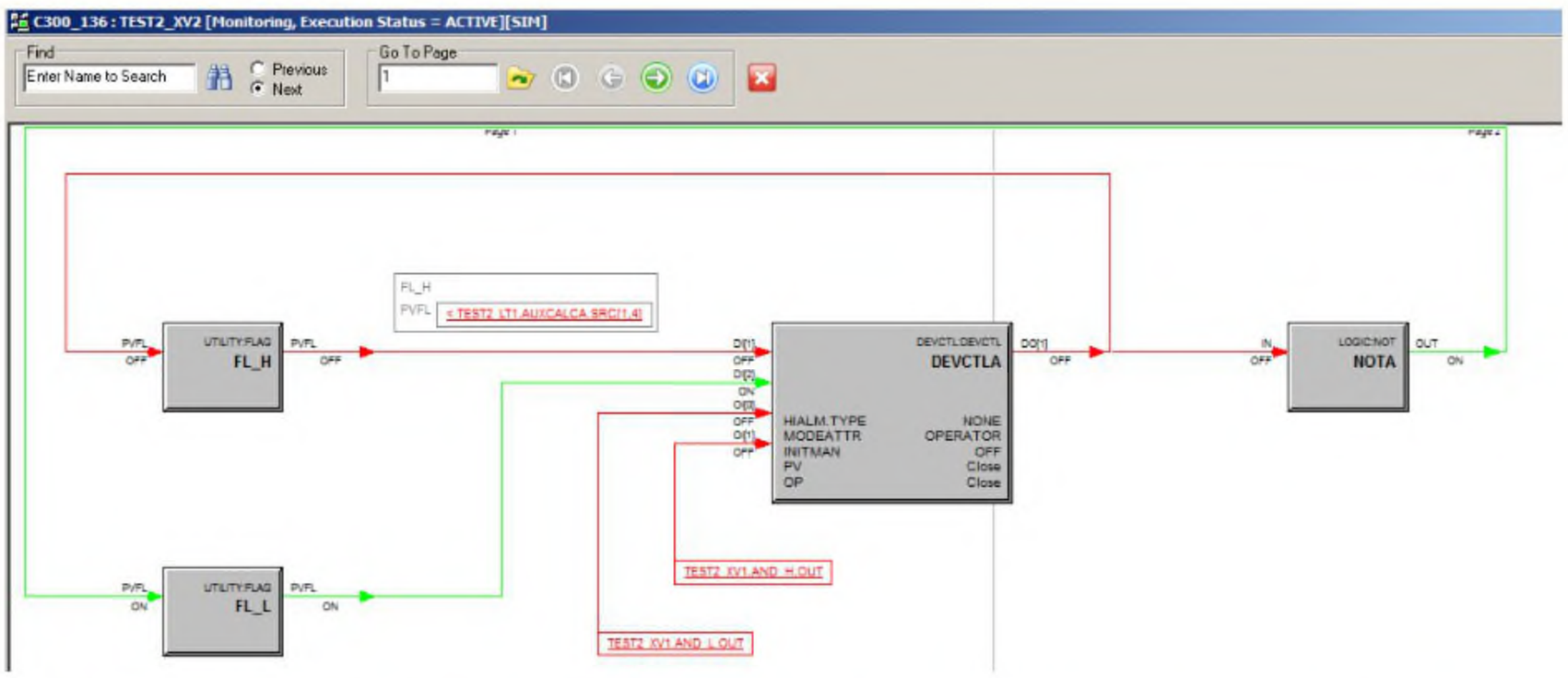


Рис. 7.40. Программа управления клапаном в линии слива

Разработка мнемосхемы выполняется аналогично представленному в п. 7.1 описанию. В данном случае необходимо создать и сохранить шейп «Ключ», показанный на рис. 7.41.

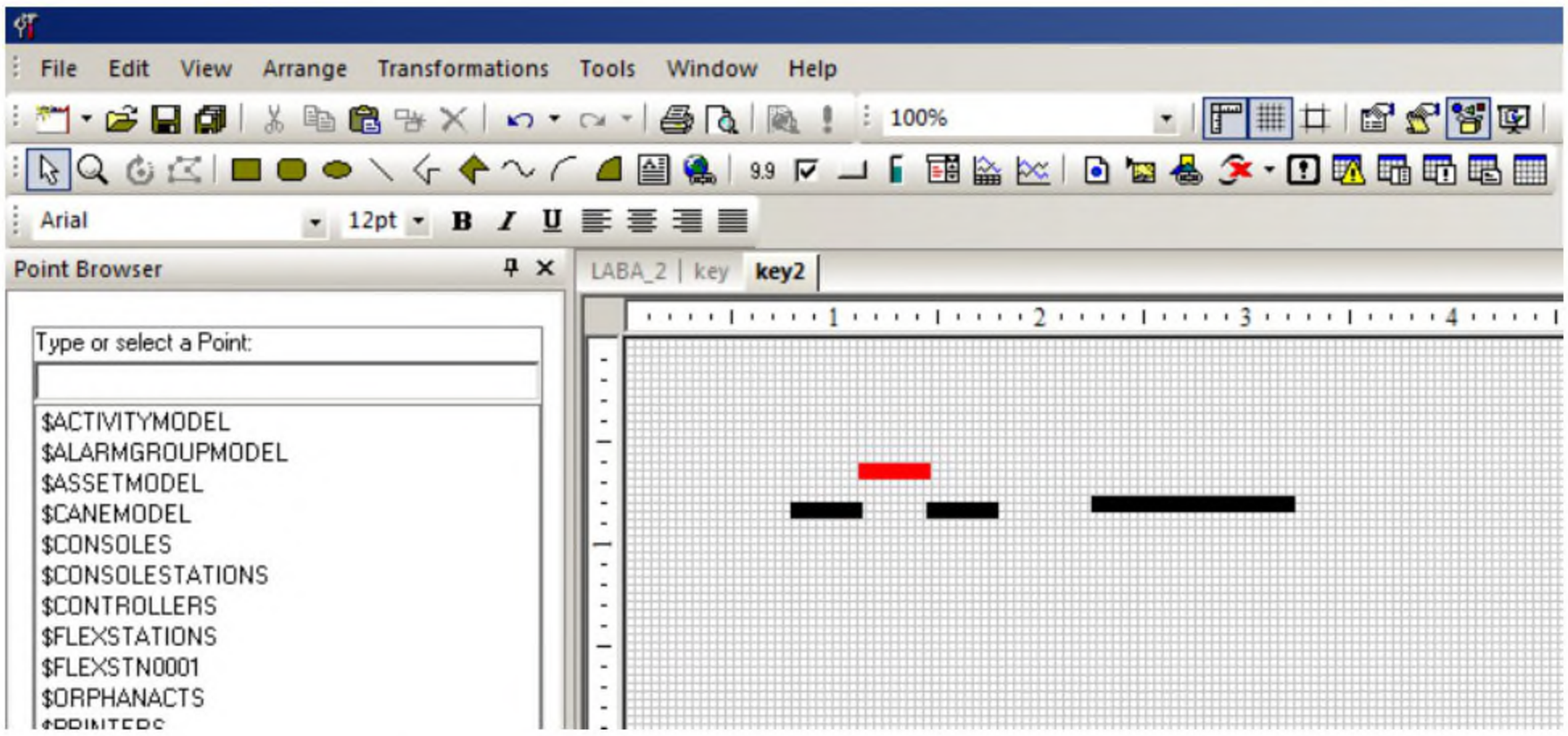


Рис. 7.41. Разветка шейпа «Ключ»

Настройка разрабатываемой мнемосхемы (рис. 7.46) осуществляется, как показано на рис. 7.42–7.45.

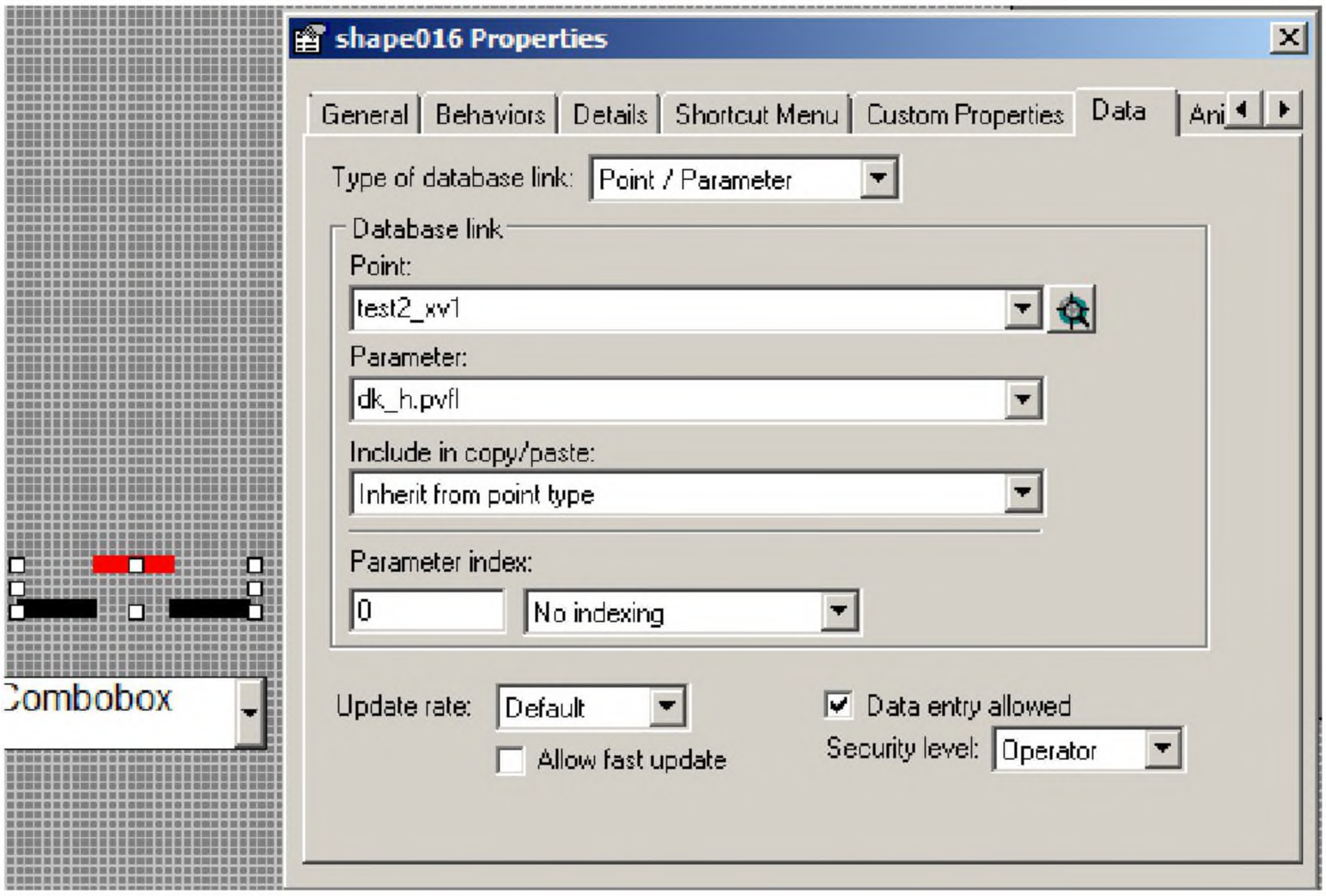


Рис. 7.42. Привязка шейпа «Ключ 1»

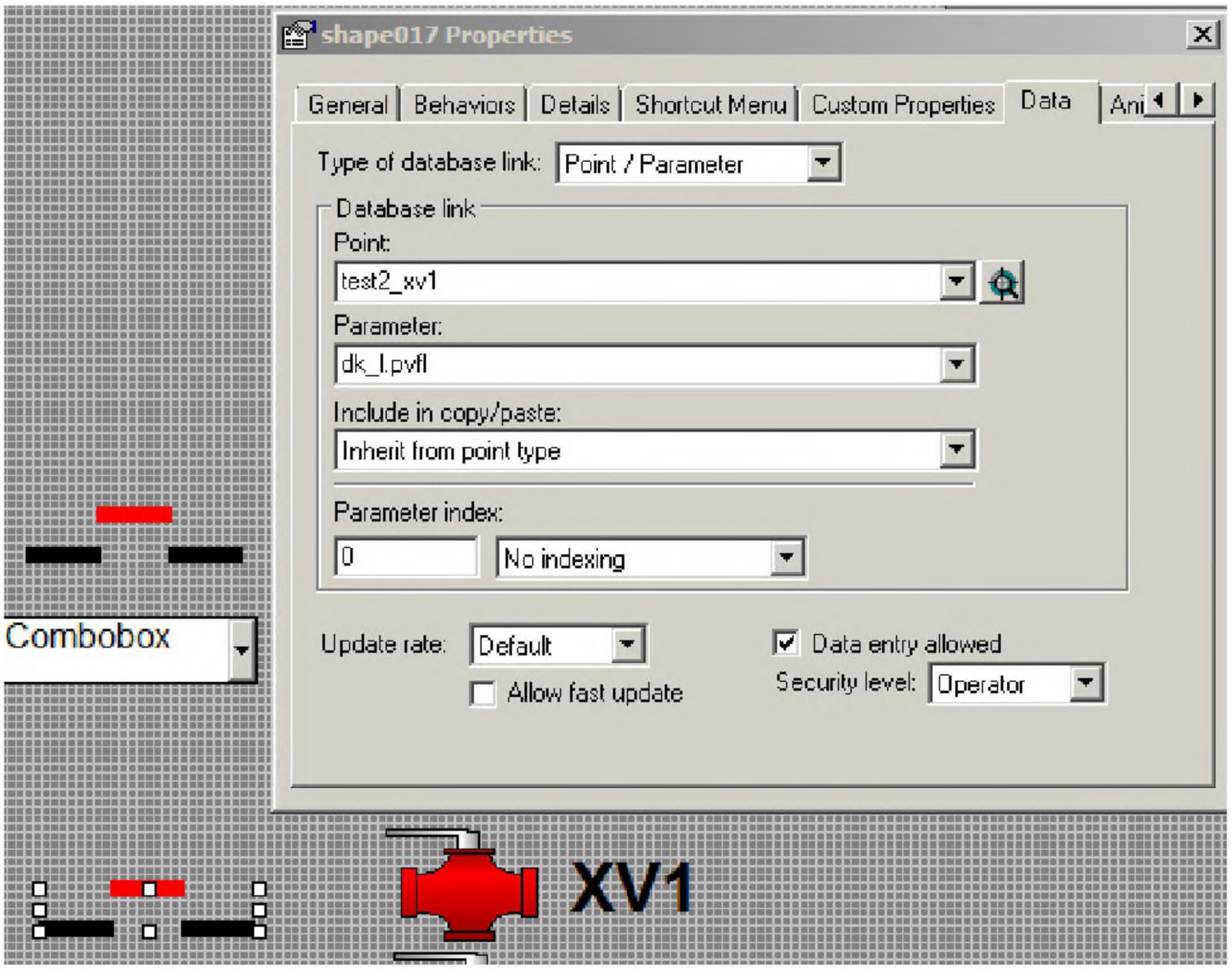


Рис. 7.43. Привязка шейпа «Ключ 2»

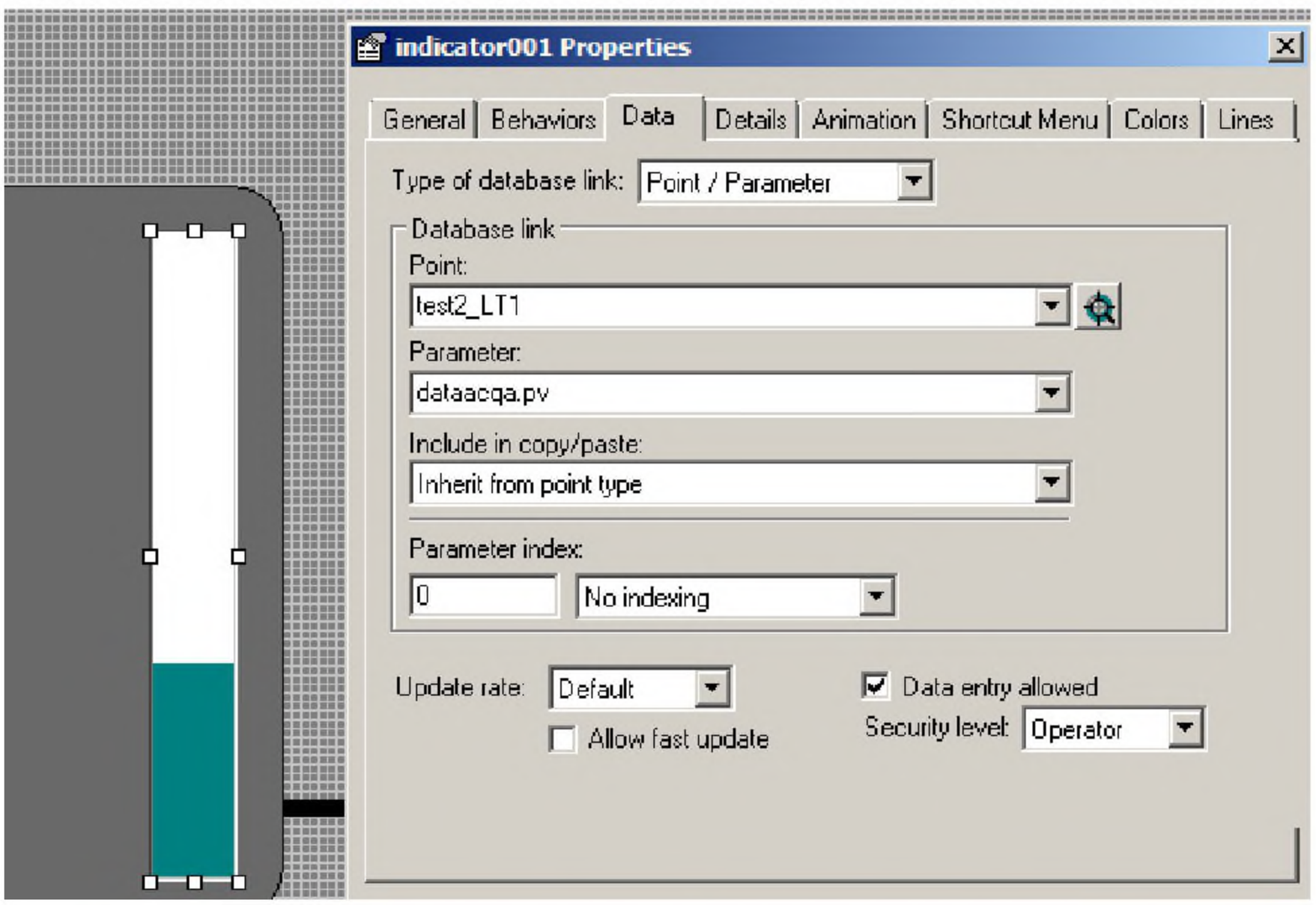


Рис. 7.44. Привязка шейпа отображения уровня жидкости

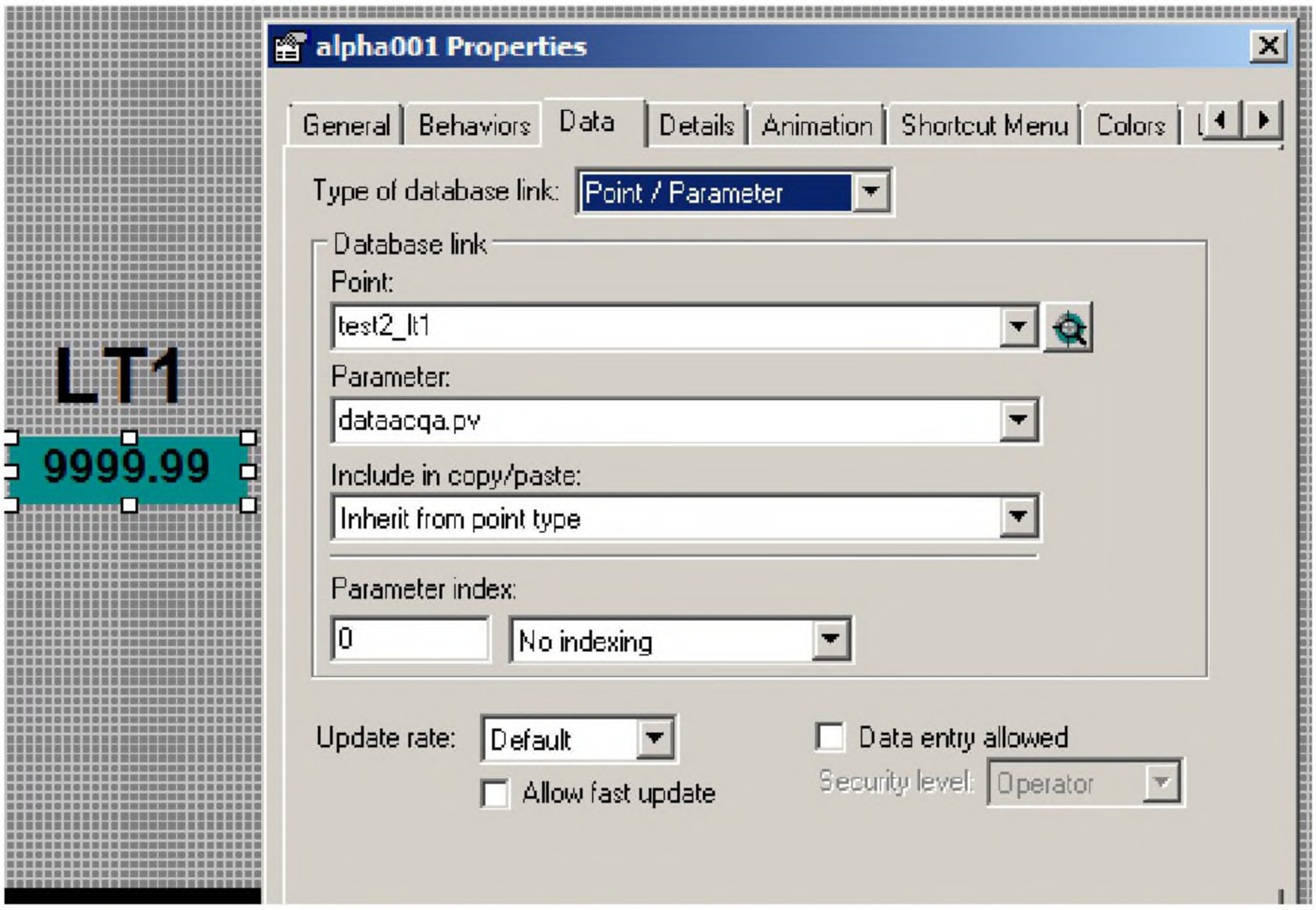


Рис. 7.45. Привязка шейпа показаний уровня жидкости

Мнемосхема, полученная в результате представленных выше конфигураций, отражена на рис. 7.46.

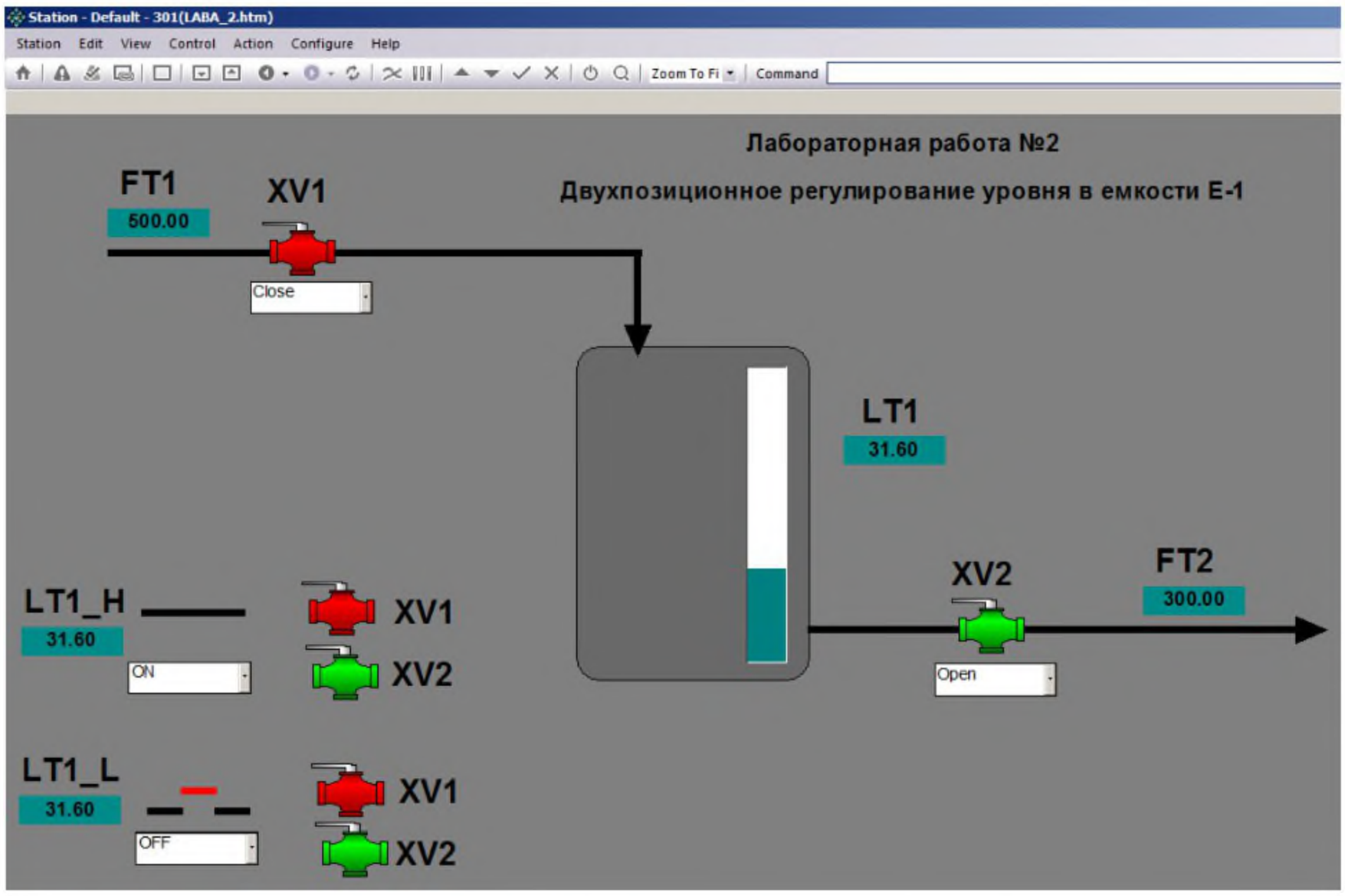


Рис. 7.46. Мнемосхема в процессе работы

7.3. Алгоритмизация системы управления насосом

Система работает следующим образом. При достижении уровнем жидкости в емкости минимального значения происходит закрытие клапана XV2 и включение насоса. Спустя время, заданное в блоке задержки ONDELAY, происходит открытие клапана XV1. При достижении уровнем максимального значения происходит открытие клапана XV2 и отключение насоса, которое влечет за собой закрытие клапана XV1.

Логическая схема функционирования первого клапана, состояние которого зависит от готовности и текущего состояния насоса, представлена на рис. 7.47, а схема функционирования второго клапана, также управляющая работой насоса, представлена на рис. 7.48.

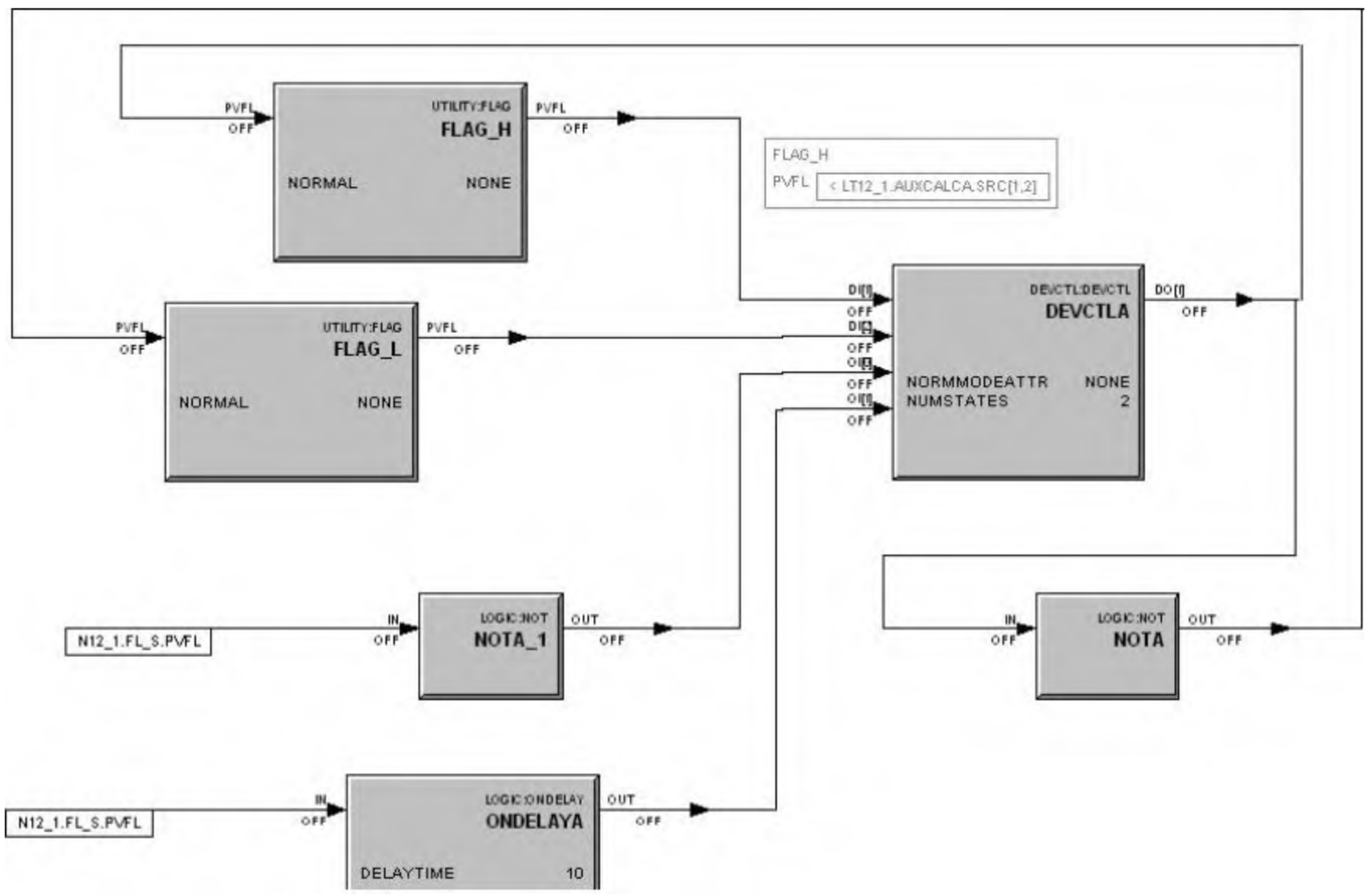


Рис. 7.47. Схема функционирования первого клапана

Логическая схема функционирования насоса представлена на рис. 7.49.

В редакторе мнемосхем DisplayBuilder разрабатываемая мнемосхема будет выглядеть, как показано на рис. 7.50.

Для задания анимации шейпа насоса, отображения и ручного переключения состояния насоса свойства добавленных shap и combobox необходимо задать соответственно рис. 7.51. В результате получим мнемосхему, представленную на рис. 7.52.

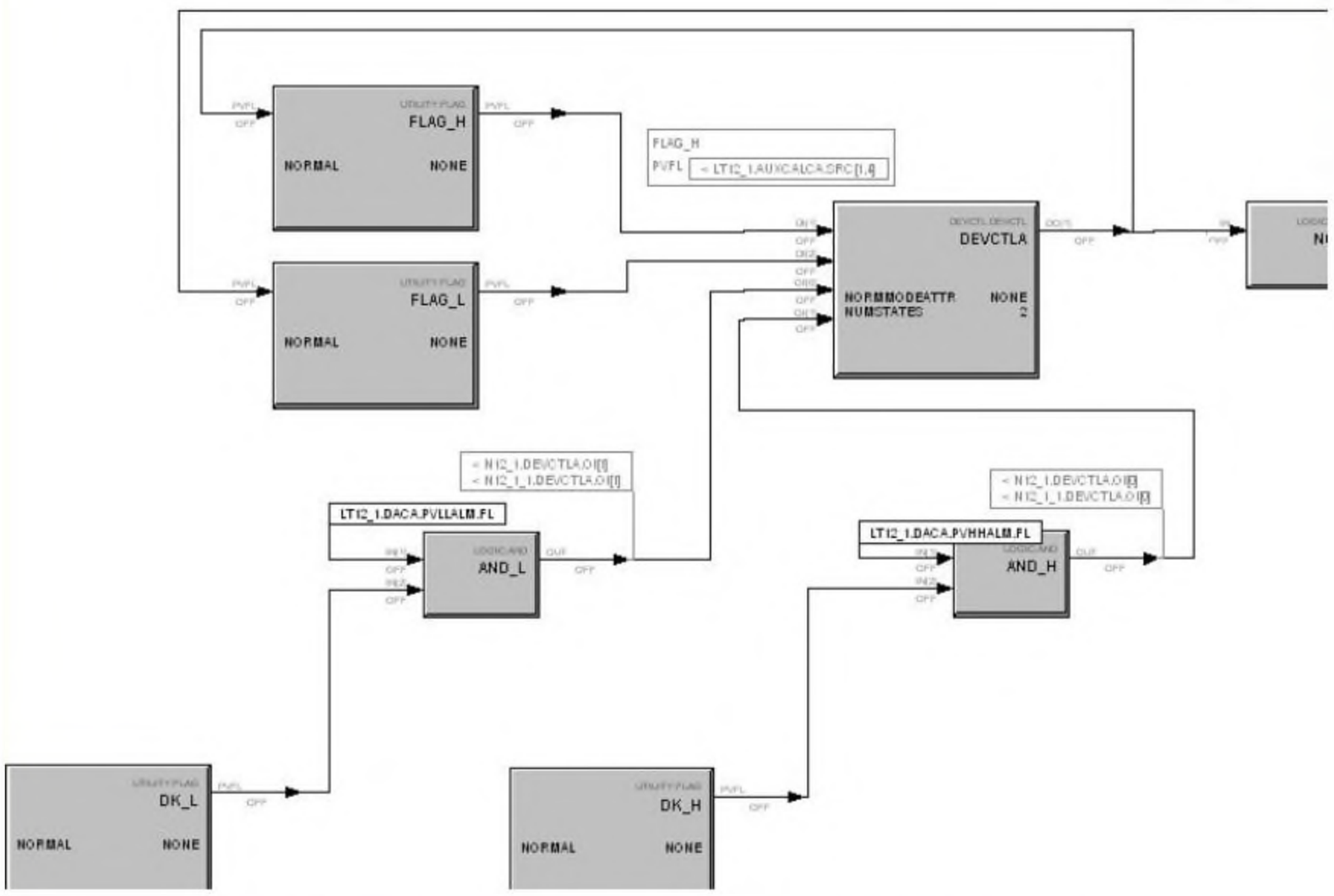


Рис. 7.48. Схема функционирования второго клапана

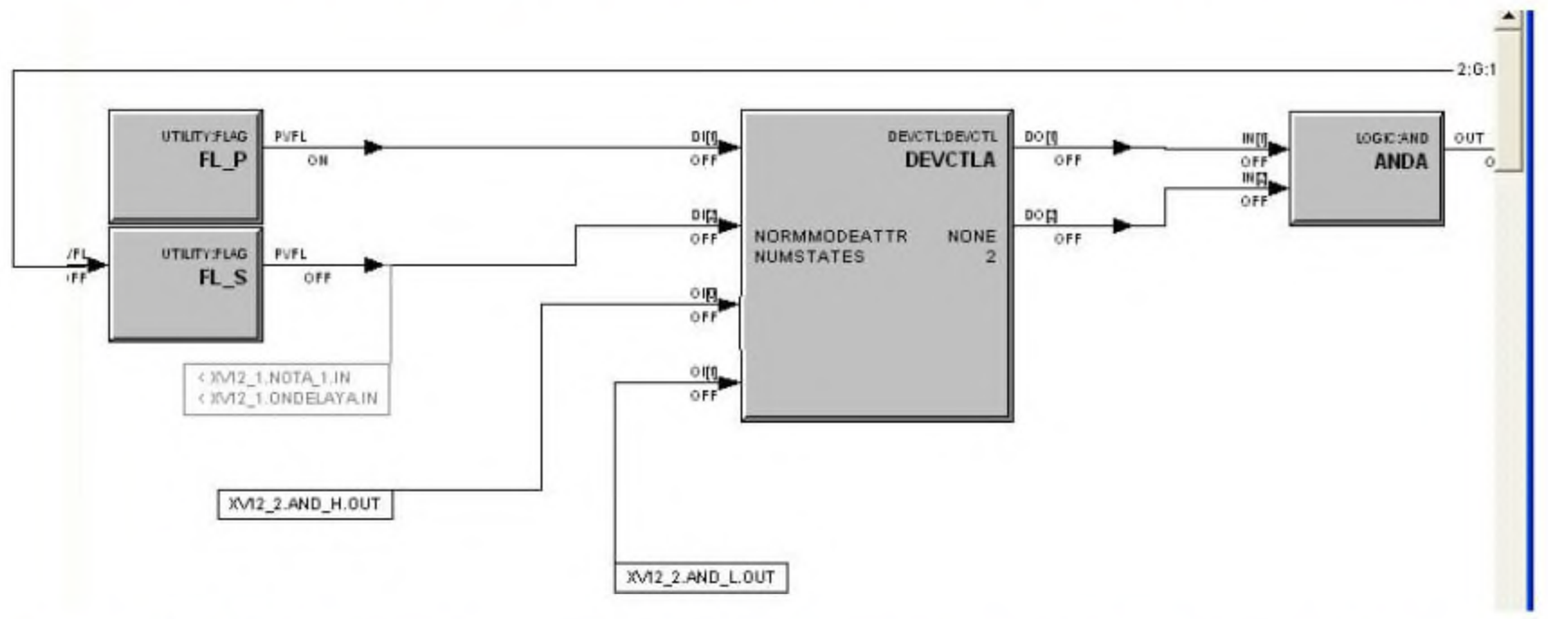


Рис. 7.49. Логическая схема функционирования насоса

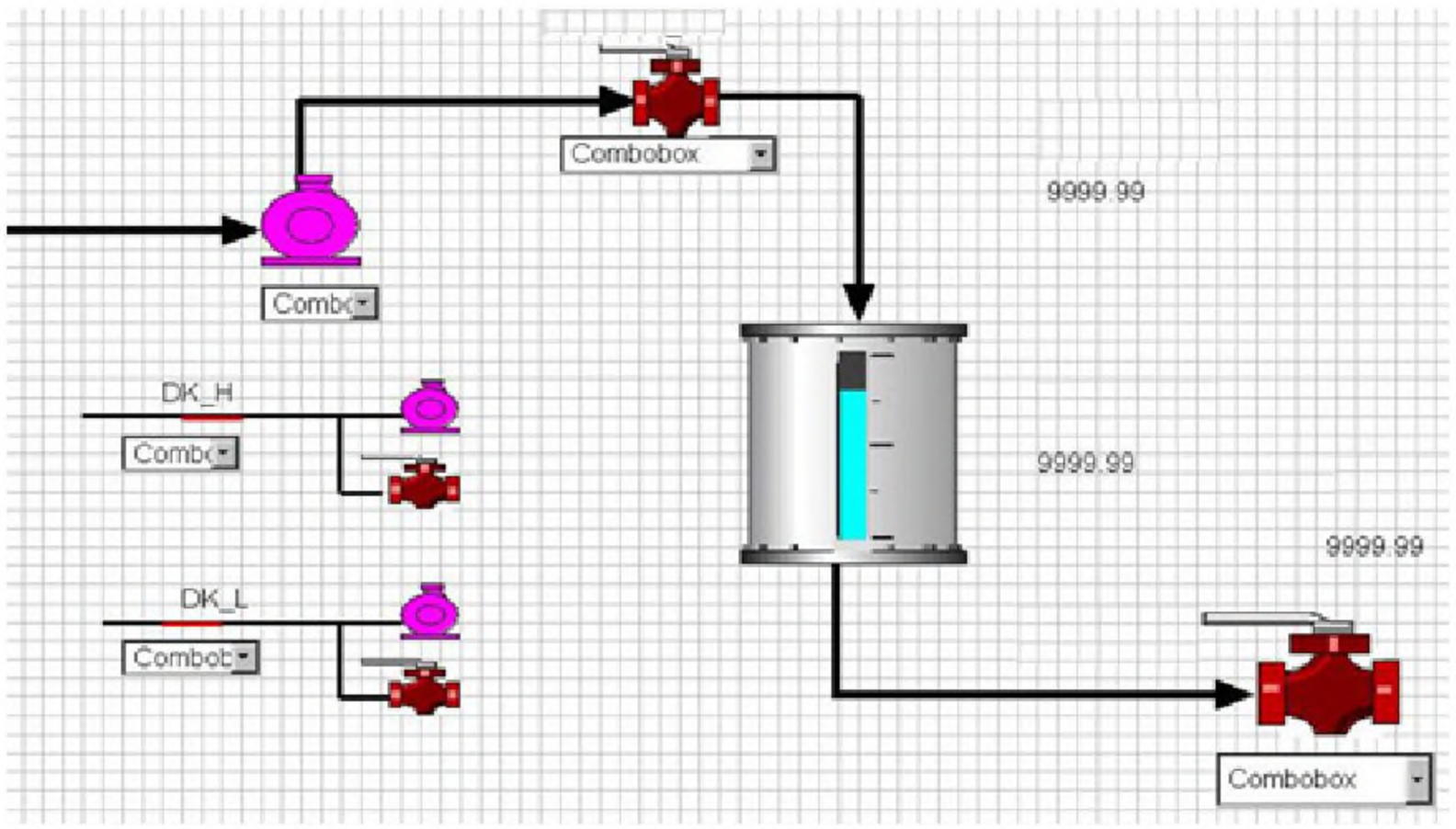


Рис. 7.50. Разрабатываемая мнемосхема
в редакторе мнемосхем DisplayBuilder

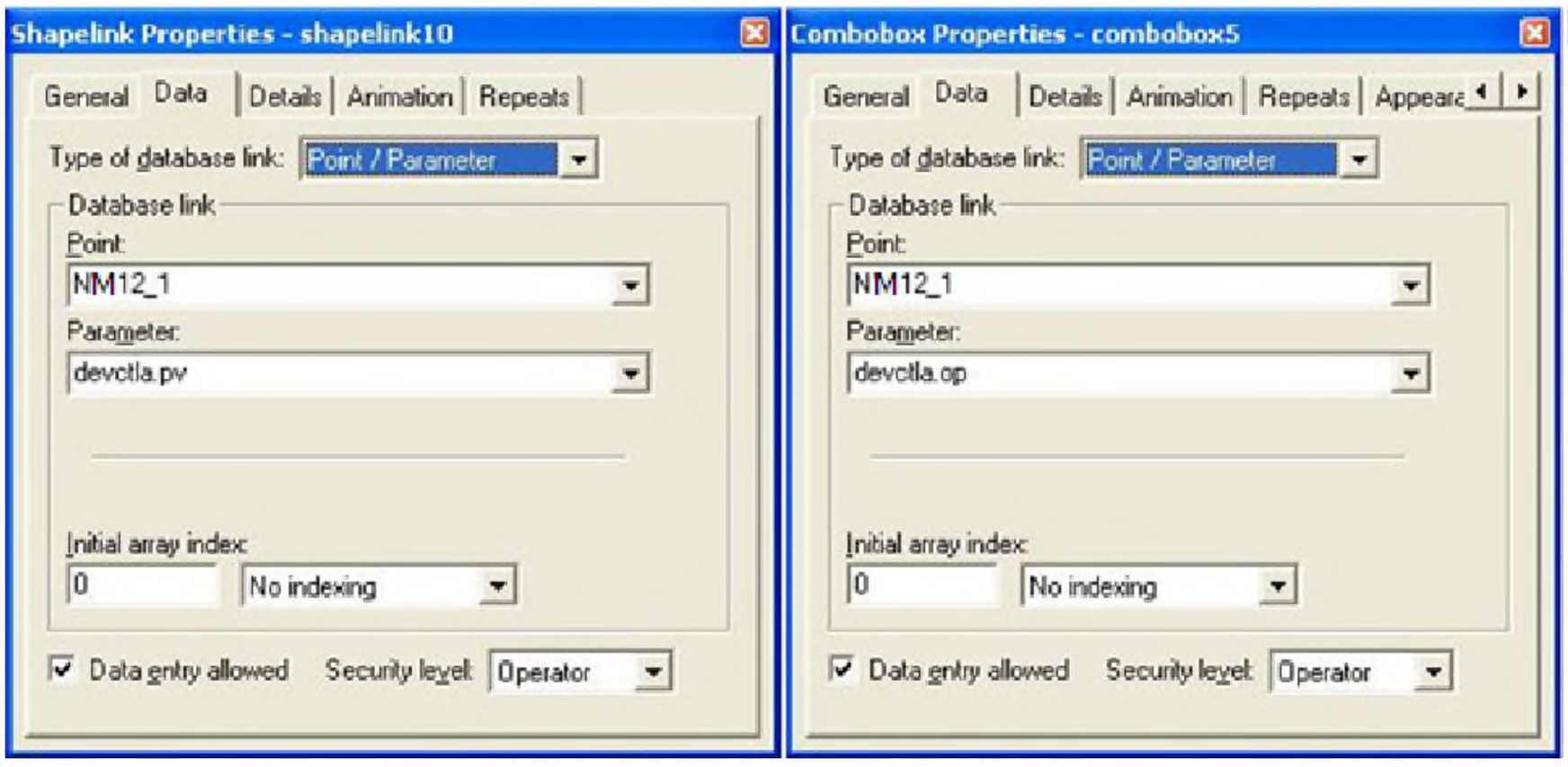


Рис. 7.51. Настройка анимации шейпов

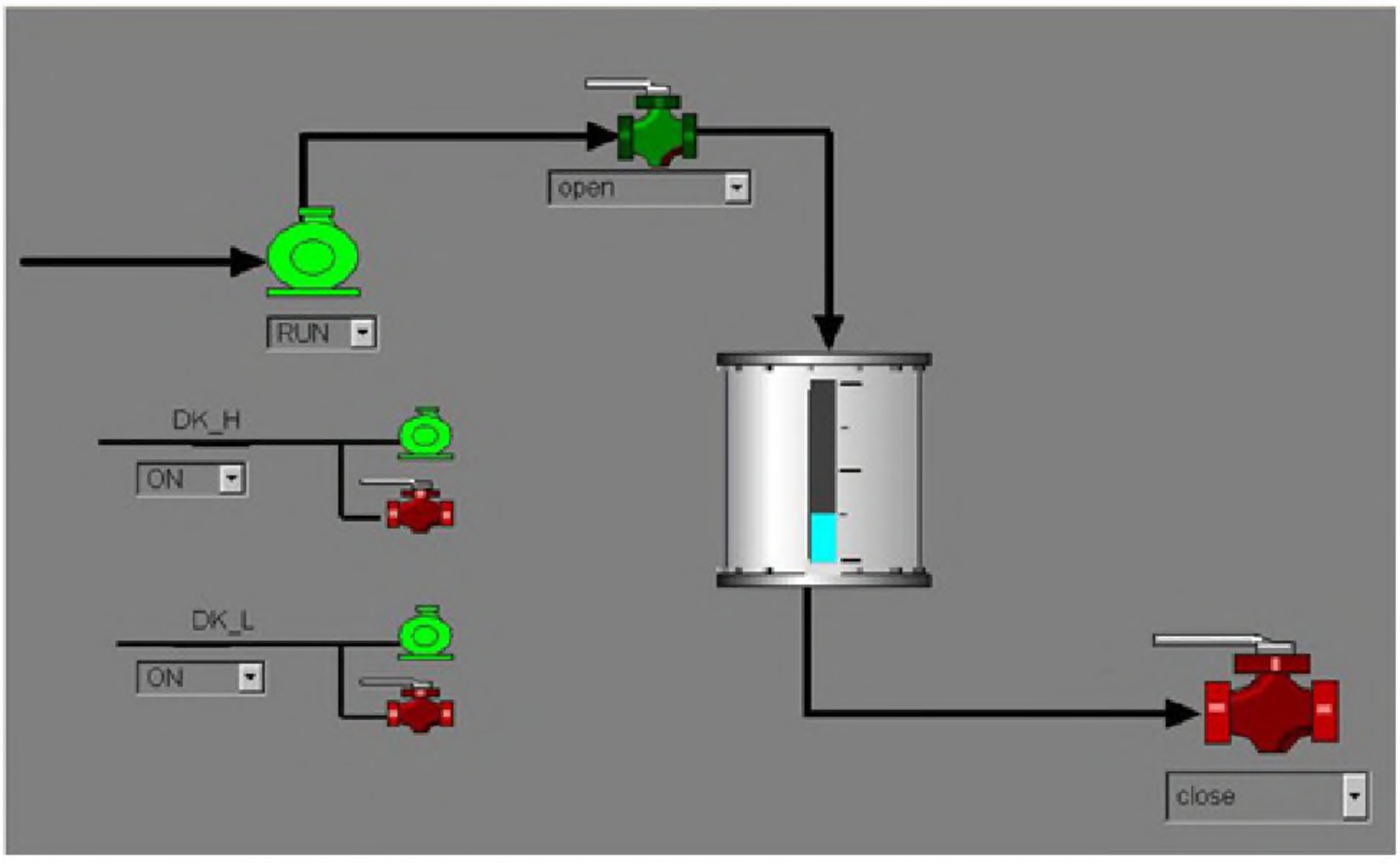


Рис. 7.52. Мнемосхема в процессе работы

7.4. Алгоритмизация системы управления трехходовым краном

В данной системе реализуется трехходовой кран, необходимый для перераспределения потока на сливе из резервуара между двумя технологическими линиями. Имитировать работу такого трехходового крана будет система, состоящая из трех клапанов – XV2, XV3 и XV4. Клапан XV3 отвечает за слив жидкости из участка трубопровода между клапанами XV2 и XV4, а клапан XV4 дублирует

действия клапана XV2. При достижении уровнем максимального значения происходит закрытие клапана XV3, затем сигнал подается на открытие клапанов XV2 и XV4 и отключение насоса. При достижении минимального значения закрываются клапаны XV2 и XV4, далее открывается клапан XV3 и включается насос. Логические схемы для клапанов XV2, XV3 и XV4 представлены на рис. 7.53–7.55 соответственно. Логика их работы следующая.

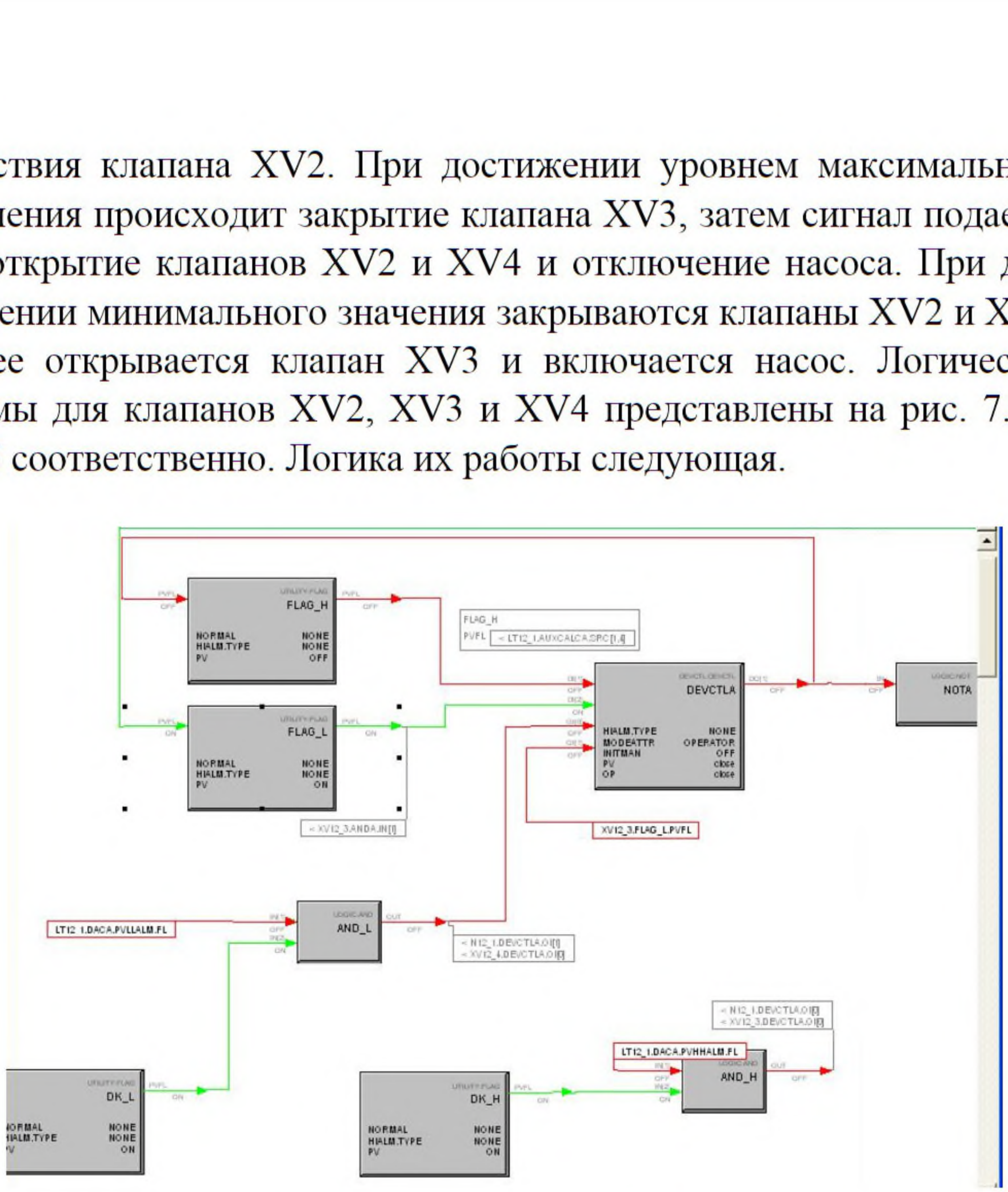


Рис. 7.53. Схема функционирования второго клапана

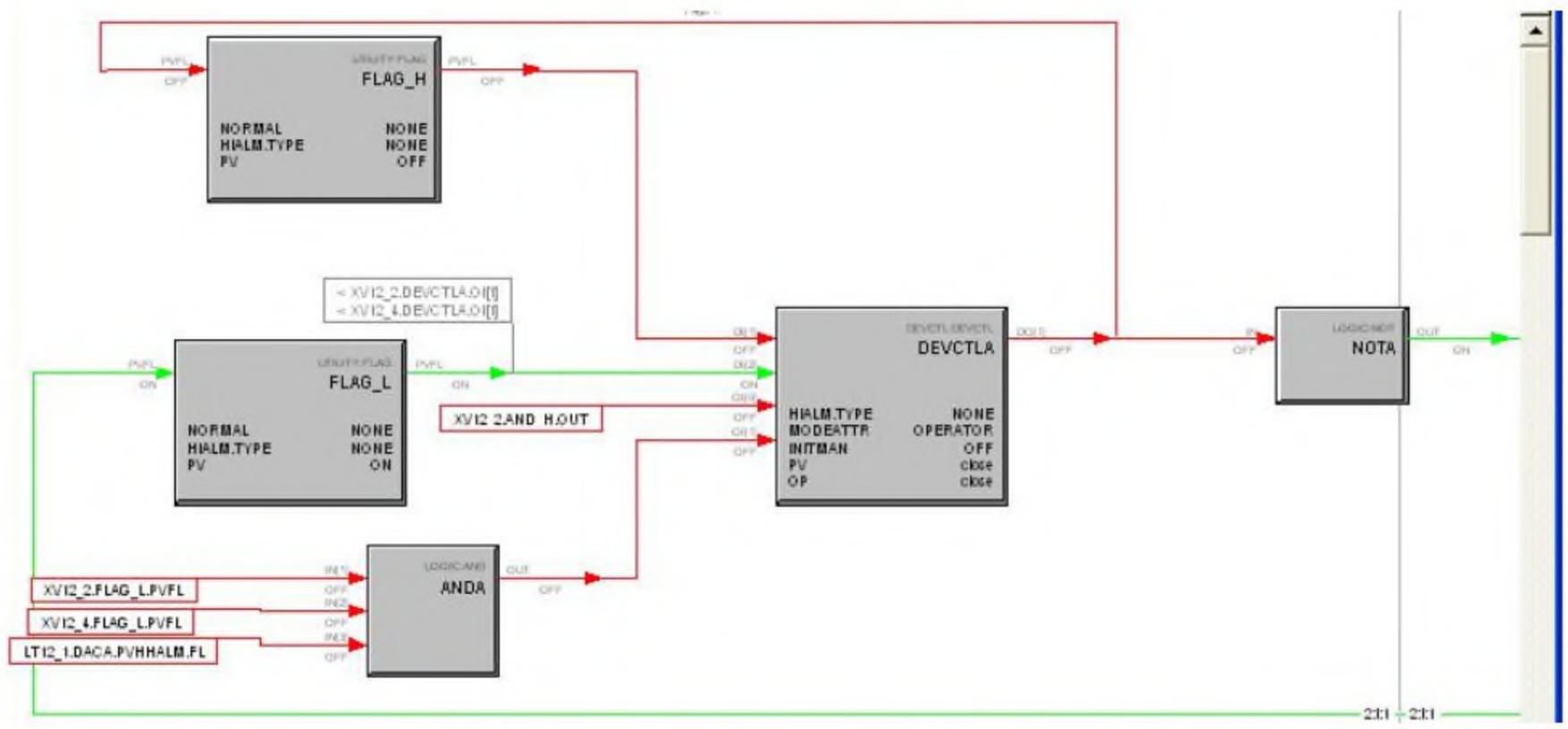


Рис. 7.54. Схема функционирования третьего клапана

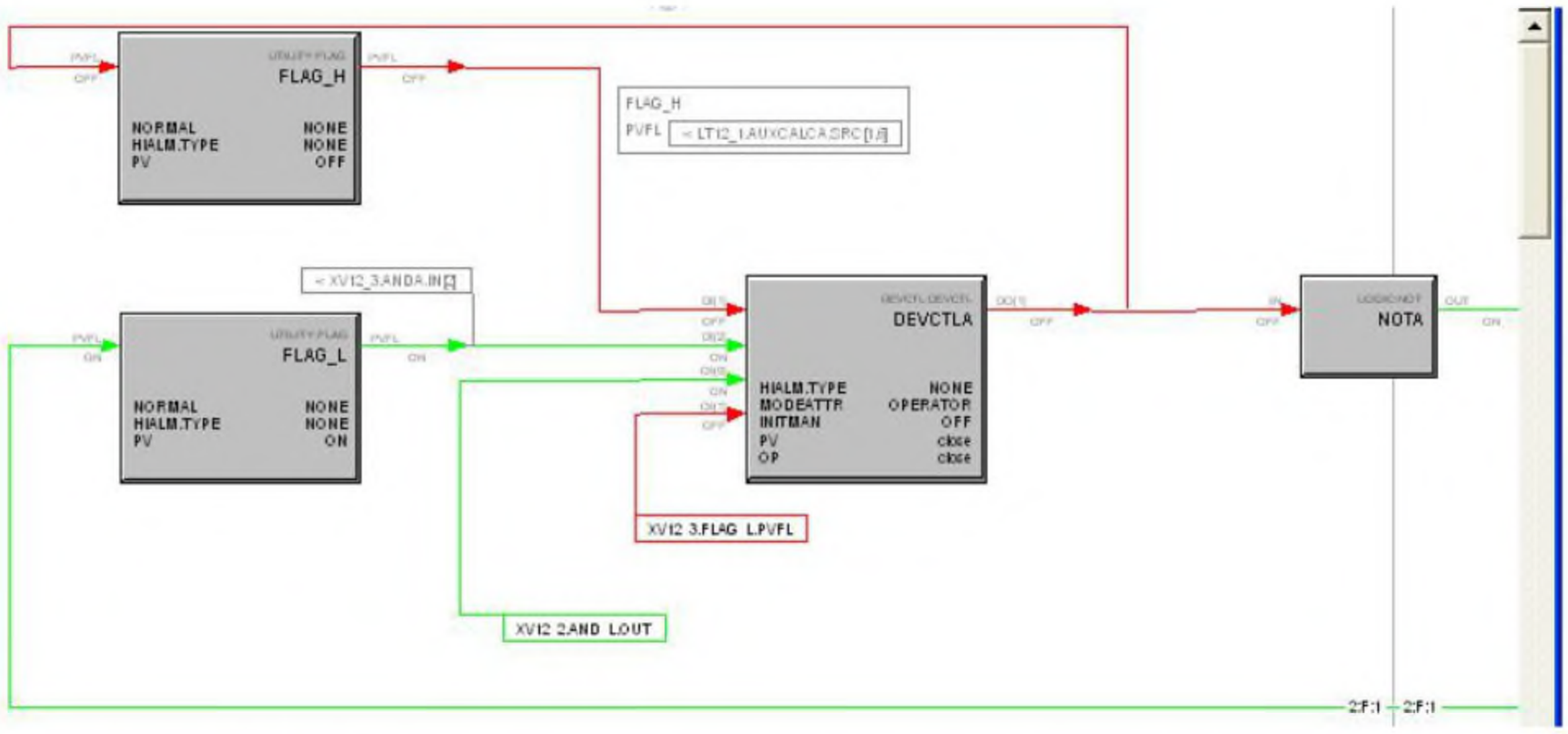


Рис. 7.55. Схема функционирования четвертого клапана

При достижении максимального значения уровня на вход OI(0) третьего клапана подается «1», что приводит к принудительному закрытию клапана. В соответствии с состоянием флага FLAG_L этого клапана на входы OI(1) второго и четвертого клапанов подаются «1», что приводит к их открытию. При достижении минимального значения уровня единичные сигналы подаются на входы OI(0) второго и четвертого клапанов. Клапан XV3 откроется только тогда, когда флаги FLAG_L второго и четвертого клапанов будут активированы и уровень будет не максимальным. Для этого третий вход блока AND клапана XV3 обладает инверсией.

В результате получим мнемосхему, представленную на рис. 7.56.

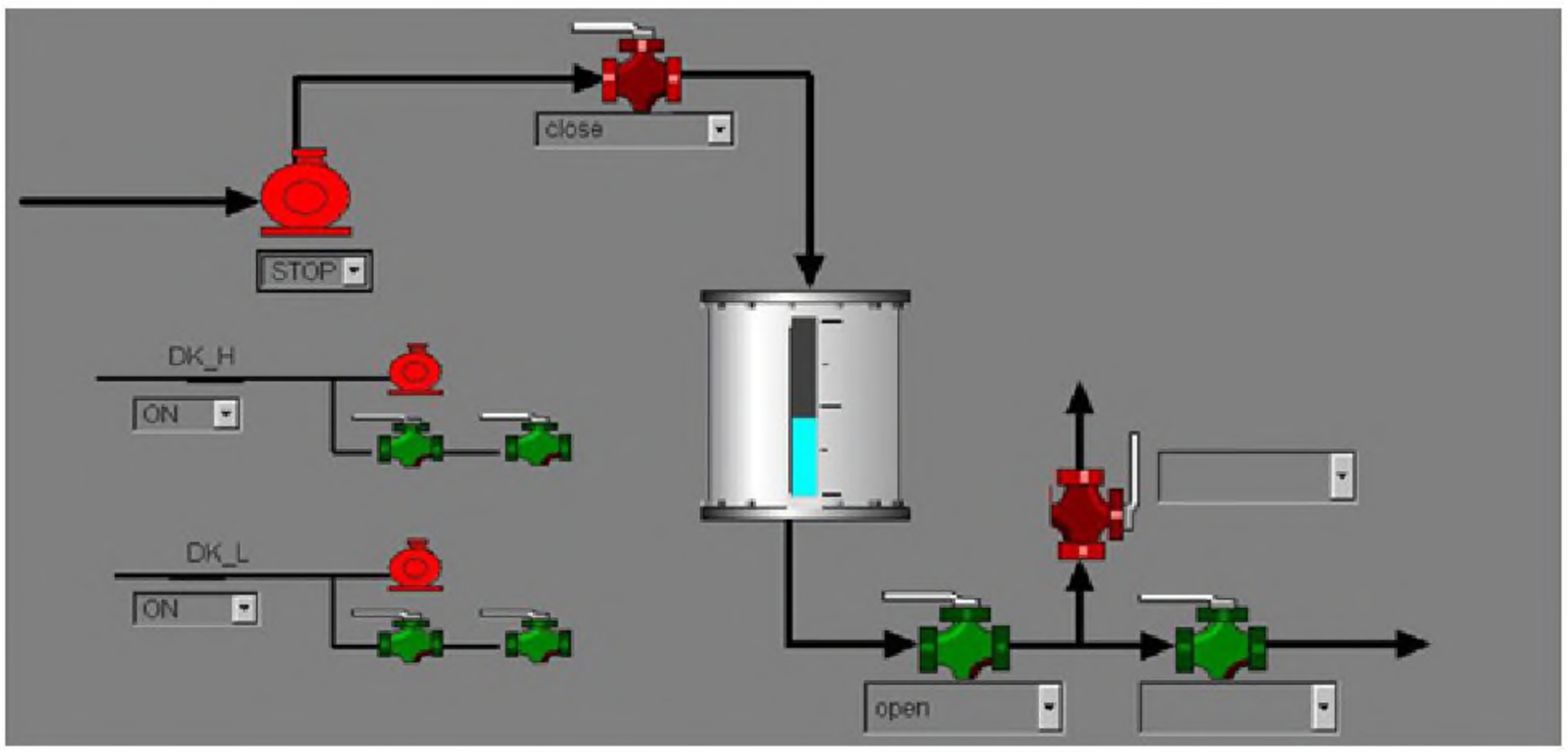


Рис. 7.56. Мнемосхема в процессе работы

7.5. Контрольные вопросы

1. Назначение и возможности пакета программирования Honeywell Experion PKS.
2. Разделы пакета программирования Honeywell Experion PKS и их назначение.
3. Языки программирования промышленных контроллеров, реализованных в пакете программирования Honeywell Experion PKS.
4. Функциональные блоки, используемые в лабораторных работах, их назначение и принцип работы.
5. Настройка функциональных блоков под конкретные прикладные задачи.
6. Таблицы истинности, их назначение, правила построения и реализация средствами Honeywell Experion PKS.
7. Этапы создания системы управления клапаном-отсекателем.
8. Этапы создания системы управления уровнем жидкости в технологическом резервуаре.
9. Этапы создания системы управления насосом.
10. Этапы создания системы управления трехходовым краном.
11. Реализация связи модулей управления между собой.
12. Этапы загрузки проекта в эмулятор контроллера и его активации.
13. Этапы создания мнемосхемы.
14. Этапы создания динамических элементов (шейпов).
15. Этапы наладки систем управления в пакете программирования Honeywell Experion PKS.

Контрольная работа № 1

Задание № 1

Придумать высказывание, описываемое формулой, представленной в табл. А.1. Указать лингвистические переменные, используемые в высказывании.

Таблица А.1

№ варианта	Формула
1	$(A \wedge B) \leftrightarrow (\overline{C} \vee D)$
2	$(A \wedge (B \rightarrow C)) \vee \overline{D}$
3	$A \wedge (\overline{B} \leftrightarrow (C \vee D))$
4	$(\overline{A} \wedge B) \rightarrow (C \vee D)$
5	$A \wedge ((B \rightarrow \overline{C}) \vee D)$
6	$(A \rightarrow (\overline{B} \wedge C)) \vee D$
7	$(A \wedge (B \vee \overline{C})) \rightarrow D$
8	$((A \wedge B) \leftrightarrow C) \rightarrow \overline{D}$
9	$(A \rightarrow (B \wedge \overline{C})) \leftrightarrow D$
10	$(A \wedge \overline{B}) \rightarrow (C \vee D)$

Задание № 2

По заданной в табл. А.2 последовательности Y построить таблицу истинности, содержащую стандартные комбинации входных сигналов (трех переменных x_1, x_2, x_3) и выходного сигнала Y. На основании составленной таблицы истинности представить переключательную схему.

Таблица А.2

№ варианта									
1	2	3	4	5	6	7	8	9	10
Y									
0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	1	0	0
0	1	0	0	0	1	0	0	1	0
0	0	1	0	1	0	1	1	0	1
1	0	1	1	0	0	1	0	1	0
0	1	0	1	1	1	0	1	0	1
1	0	1	0	1	1	0	0	1	0
0	0	0	1	0	0	1	0	0	1

Задание № 3

На основании таблицы истинности, составленной при выполнении задания 2, представить функцию *Y* в ДСНФ и КСНФ.

Задание № 4

На основании таблицы истинности, составленной при выполнении задания 2, построить карту Карно и по ней минимизировать функцию *Y*.

Задание № 5

Дать определение понятиям «изолированное состояние», «тупиковое состояние», «локальное состояние».

По заданной ниже для своего варианта матрице смежности построить граф состояний и определить по нему изолированные, локальные и тупиковые состояния.

Вариант 1

№	1	2	3	4	5	6	7	8
1								
2		1						
3			3				8	
4				4		7		
5		2			6			10
6							9	
7				5				11
8								12

Вариант 2

№	1	2	3	4	5	6	7	8
1						9		
2				6				
3		4						11
4	1		5	7				12
5	2				8	10		
6	3							
7								
8								

Вариант 3

№	1	2	3	4	5	6	7	8
1			3					
2				5			10	
3	1							12
4		2					11	
5				6				
6					7			
7						8		
8			4			9		

Вариант 4

№	1	2	3	4	5	6	7	8
1	1							
2		2				6		
3			3				8	10
4				4				
5							9	11
6				5				
7						7		
8								12

Вариант 5

№	1	2	3	4	5	6	7	8
1								
2	1			6				
3		2			7			
4		3	4				9	
5			5				10	
6								11
7								12
8								

Вариант 6

№	1	2	3	4	5	6	7	8
1								
2						7		
3				5			10	
4			3		6			
5			4			8		
6							11	
7		2				9		12
8	1							

Вариант 7

№	1	2	3	4	5	6	7	8
1								
2	1				9			
3		3		7		10		
4			5				12	
5	2					11		
6		4		8				
7			6					
8								

Вариант 8

№	1	2	3	4	5	6	7	8
1								11
2	1		4					
3		2				6		
4							9	
5						7		
6							10	
7						8		
8		3	5					12

Вариант 9

№	1	2	3	4	5	6	7	8
1								11
2	1			7				
3						9		
4	2	3		8		10		
5								
6		4	5					
7								
8			6					12

Вариант 10

№	1	2	3	4	5	6	7	8
1	1							
2			5					
3		2		6				
4		3		7			10	
5		4			8			11
6								
7					9			
8								12

Контрольная работа № 2

Задание

Построить циклограмму работы двух исполнительных устройств (ИУ) *X* и *Y*, представленных на рис. Б.1, с указателями положения A_1, A_2 и B_1, B_2 соответственно для следующей последовательности их работы:

Вариант 1. $\overline{X},\overline{Y},Y,X$.

Вариант 2. $\overline{X},X,Y,\overline{Y}$.

Вариант 3. $X,\overline{X},Y,\overline{Y}$.

Вариант 4. $\overline{X},Y,\overline{Y},X$.

При этом функции реализуют прямой ход штока (слева направо), функции с отрицанием – обратный (справа налево).

Рис. Б.1. Исполнительные устройства

Для этого:

1) построить временную диаграмму работы ИУ и обозначить на ней реализацию функций;

2) построить таблицу включений;

3) построить начальную циклограмму работы ИУ;

4) построить ряд весовых коэффициентов согласно начальной циклограмме;

5) построить ряд весовых коэффициентов с учетом включения/отключения элемента памяти;

6) построить реализуемую циклограмму работы ИУ.

СПИСОК РЕКОМЕНДУЕМОЙ ЛИТЕРАТУРЫ

1. Круглов В.В., Дли М.И, Голунов Р.Ю. Нечеткая логика и искусственные нейронные сети. – М.: Физматлит, 2000. – 224 с.

2. Леоненков А.В. Нечеткое моделирование в среде MATLAB и fazzyTECH. – СПб.: БХВ-Петербург, 2005.

3. Мельников В.Н. Логические задачи. – Одесса: Выща шк., 1989. – 344 с.

4. Петров И.В. Программируемые контроллеры: Стандартные языки и инструменты / под ред. проф. В.П.Дьяконова. – М.: СОЛОН-Пресс, 2003. – 256 с.

5. Усенко В.В. Логические системы управления: пособие для работников АСУ тепловых электростанций. – М.: Изд-во МЭИ, 2001. – 72 с.

6. Чикуров Н.Г. Логический синтез дискретных систем управления: учеб. пособие. – Уфа: Уфимский гос. авиац. техн. ун-т, 2003. – 132 с.

7. Шалыто А.А. Логическое управление: Методы аппаратной и программной реализации. – СПб.: Наука, 2000. – 780 с.

8. Шалыто А.А. Switch-технология. Алгоритмизация и программирование задач логического управления. – СПб.: Наука, 1998. – 627 с.

9. Штовба С.Д. Проектирование нечетких систем средствами MATLAB. – М.: Телеком, 2007.

10. Яновская С.А. Лекции по алгебре логики. – СПб.: Ленанд, 2015. – 264 с.

Учебное издание

Сташков Сергей Игоревич

АЛГОРИТМИЗАЦИЯ СИСТЕМ ПРОГРАММНО-ЛОГИЧЕСКОГО УПРАВЛЕНИЯ

Учебное пособие

Редактор и корректор *М.А. Шемякина*

Подписано в печать 26.11.2021. Формат 60×90/16.
Усл. печ. л. 6,2. Тираж экз. Заказ № 253/2021.

Издательство
Пермского национального исследовательского
политехнического университета.
Адрес: 614990, г. Пермь, Комсомольский пр., 29, к. 113.
Тел. (342) 219-80-33.