МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ
РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ АГЕНТСТВО ПО ОБРАЗОВАНИЮ

НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
МОСКОВСКИЙ ЭНЕРГЕТИЧЕСКИЙ ИНСТИТУТ

В.Ф. КУЗИЩИН, Е.И. МЕРЗЛИКИНА

МЕТОДИЧЕСКИЕ УКАЗАНИЯ ПО ПРИМЕНЕНИЮ СРЕДЫ CODESYS ДЛЯ РАЗРАБОТКИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ДЛЯ КОНТРОЛЛЕРОВ

Методическое пособие по курсам
«Технические средства автоматизации» и
«Автоматизация систем теплоснабжения и кондиционирования»
для студентов, обучающихся по направлению
«Теплоэнергетика и теплотехника»

Москва Издательский дом МЭИ 2013

УДК
621.36
К 89

Утверждено учебным управлением МЭИ

Подготовлено на кафедре автоматизированных систем управления тепловыми процессами

Рецензент: доктор техн. наук, профессор А.В.Андрюшин

Кузищин В. Ф., Мерзликина Е.И.
К 89. Методические указания по применению среды программирования CoDeSys для разработки программного обеспечения для контроллеров: Методическое пособие/ В.Ф. Кузищин, Е.И. Мерзликина. – М.: Издательский дом МЭИ, 2013. – 32 с.

Посвящено изучению универсального комплекса программирования контроллеров CoDeSys на основе языков стандарта МЭК 61131-3. Рассматриваются языки программирования, вопросы создания проектов технологических программ для конкретных контроллеров, программные компоненты библиотек, используемых при создании систем автоматизации, способы создания визуализаций, а также привязка переменных к входам и выходам контроллеров серии ПЛК150 фирмы «Овен».

.

Предназначено для студентов ИТАЭ, ИПЭЭФ.

_____

Учебное издание
Кузищин Виктор Федорович
Мерзликина Елена Игоревна

МЕТОДИЧЕСКИЕ УКАЗАНИЯ ПО ПРИМЕНЕНИЮ СРЕДЫ CODESYS ДЛЯ РАЗРАБОТКИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ДЛЯ КОНТРОЛЛЕРОВ

Методическое пособие по курсам
«Технические средства автоматизации» и
«Автоматизация систем теплоснабжения и кондиционирования»
для студентов, обучающихся по направлению
«Теплоэнергетика и теплотехника»

Редактор издательства В.В. Сомова

Темплан издания МЭИ 2009, учебн.		Подписано в печать
Печать офсетная	Формат 60x84/16	Физ. печ. л. 2,0
Тираж 100 экз.	Изд. №	Заказ №

ОАО «Издательский дом МЭИ», 111250, Москва, Красноказарменная, д. 14
Отпечатано в типографии ФКП «НИИ «Геодезия», 141292, Московская обл.,
г. Красноармейск, просп. Испытателей, д. 14

Введение. Комплекс программирования CoDeSys

CoDeSys – универсальный комплекс разработки программ для программируемых логических контроллеров (ПЛК) на основе языков стандарта МЭК 61131-3. Его название расшифровывается как Controller Development System, он выпускается немецкой фирмой 3S-Smart Software Solutions GmbH [1]. Первая версия CoDeSys вышла в 1994 году, в настоящее время доступна третья версия пакета, в которой поддерживается объектно-ориентированное программирование [2].

В состав комплекса программирования ПЛК входят две обязательные части: среда разработки программ и система исполнения. Среда разработки программ CoDeSys функционирует на персональном компьютере (ПК) и не имеет конкретной аппаратной привязки, поэтому с её помощью можно создавать программы для любых контроллеров. Система исполнения (CoDeSys SP) функционирует в контроллере и устанавливается его производителем. Она обеспечивает загрузку кода программы, ее исполнение, а также отладочные функции.

Среда разработки программ CoDeSys является бесплатной, и ее можно скачать с сайта производителя. Компания 3S лицензирует только системы исполнения. Существует русифицированная версия CoDeSys. Этот программный пакет может быть установлен на компьютер, работающий под управлением операционной системы Windows.

Для загрузки написанной программы в какой-либо конкретный ПЛК необходим target-файл (файл целевой платформы), обычно поставляемый производителем контроллера. В этом файле находится информация о ресурсах данного контроллера, в том числе - о входах и выходах, типах и расположении данных в памяти и т.д. Для инсталляции target-файлов служит программа InstallTarget, которая устанавливается на компьютер вместе с пакетом CoDeSys.

CoDeSys применяется для программирования многими фирмами, в частности, WAGO, ABB, Beckhoff, и т.д.. В России CoDeSys широко применяется фирмой Овен (Москва) и Пролог (Смоленск). Одним из преимуществ CoDeSys является наличие в нем режима эмуляции, что позволяет отлаживать программы непосредственно на компьютере, не загружая их в контроллеры и не привлекая на стадии отладки работающее технологическое оборудование. Это свойство позволяет также широко использовать CoDeSys в учебном процессе.

Технические решения на базе CoDeSys и поддерживающих его контроллеров применяются, например, в Германии для автоматизации ветряных энергетических турбин Enercon [1] с использованием контроллеров Modular PLC. В России технические решения на базе CoDeSys и оборудования фирмы Овен применяются при автоматизации

технологических процессов в сфере ЖКХ, в энергетике, металлургии, химической, пищевой и других отраслях промышленности [3].

В настоящее время существует некоммерческая организация CoDeSys Automation Alliance (CAA) – объединение компаний-производителей ПЛК, поддерживающих CoDeSys, в которое входят более семидесяти фирм [1].

Языки программирования, встроенные в среду CoDeSys

В настоящее время для программирования контроллеров широко используются технологические языки программирования стандарта МЭК 61131-3 (по-английски – IEC 61131-3, IEC – International Electrotechnical Commission, Международная электротехническая комиссия): ST, IL, LD, FBD, SFC или их модификации. В CoDeSys используются шесть языков – вышеуказанные стандартные языки и CFC, нестандартный язык, родственный FBD.

Среди рассматриваемых языков два – текстовые (ST и IL), остальные – графические. Ниже дается их краткая характеристика.

Язык ST (Structured Text, структурированный текст) – текстовый язык высокого уровня, близкий по структуре и синтаксису к языку Pascal [4]. На ST удобно записывать функции и функциональные блоки, описывать действия с аналоговыми сигналами и числами с плавающей точкой, а также действия и переходы языка SFC. ST удобен для написания больших программ и в России традиционно достаточно популярен. Пример простой программы на языке ST представлен на рис. 1. Данная программа выполняет сложение двух целых чисел, после чего посредством команды ветвления (IF-THEN-ELSE) присваивает переменной некоторое значение в зависимости от значения полученной суммы.

```
0001 PROGRAM PLC_PRG
0002 VAR
0003     A: INT := 1;
0004     B: INT := 1;
0005     C: INT := 1;
0006     D: INT:=10;
0007 END_VAR
0001 A:=B+C;
0002 IF A>0 THEN
0003  D:=10;
0004 ELSE
0005  D:=0;
0006 END_IF;
```

Рис. 1. Пример программы на языке ST.

Язык IL (Instruction List, список инструкций) – также текстовый язык, ассемблер низкого уровня. Программа на этом языке представляет собой список последовательных инструкций, каждая из которых записывается в отдельной строке. Пример программы на языке IL приведен на рис. 2, данная программа выполняет сложение двух целых чисел. Этот язык достаточно популярен, но предназначен в первую очередь для профессиональных программистов и разработчиков.

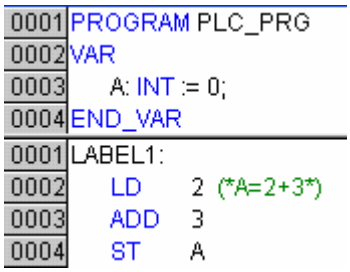

Рис. 2. Пример программы на языке IL.

Оставшиеся четыре языка, встроенные в CoDeSys – графические.

Язык LD (Ladder Diagram, язык релейных диаграмм или релейно-контактных схем) предназначен для реализации программ логического управления, при этом в данном языке используются только логические переменные (то есть, переменные типа Bool). Основными элементами программы на данном языке являются контакты и обмотки. Пример программы, реализующей логический элемент «И» на три входа, показан на рис. 3.

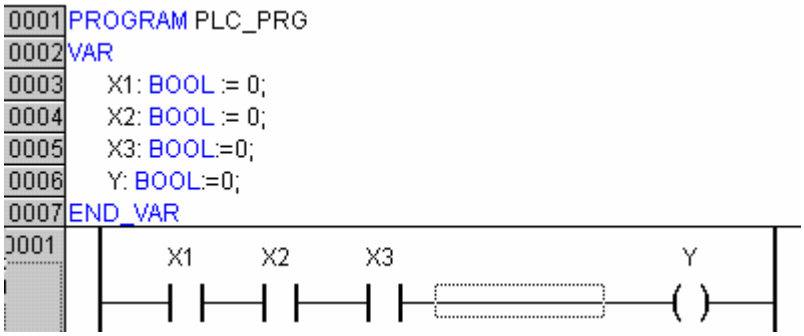

Рис. 3. Пример программы на языке LD.

Язык FBD (Function Block Diagram, язык функциональных блоков) является графическим. Программа на этом языке состоит из функциональных блоков, соединенных между собой, причем блоки организованы в цепи, расположенные одна под другой. Сначала выполняются слева направо все действия в первой сверху цепи, затем – во второй и т.д. Пример программы простой программы из одной цепи на

языке FBD приведен на рис. 4. Эта программа реализует выражение $Y = (A + B) * C$.

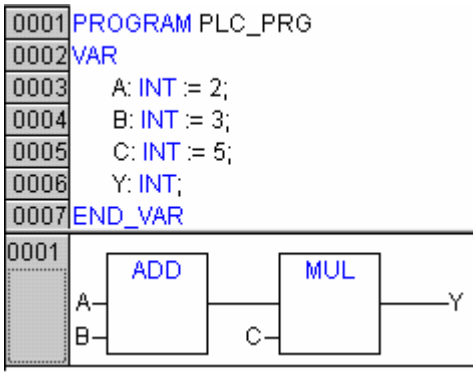

Рис. 4. Пример программы на языке FBD.

Язык CFC (Continuous Function Chart, язык непрерывных функциональных схем) – графический язык, родственный языку FBD. Этот язык не входит в стандарт МЭК 61131-3. В программах, написанных на CFC, функциональные блоки размещаются в области кода произвольно, кроме того, пользователь может сам установить порядок их выполнения. Оба языка – CFC и FBD – являются языками высокого уровня. Пример программы на языке CFC приведен на рис. 5. Данная программа реализует выражение $Y = \frac{A}{B} + C * D$.

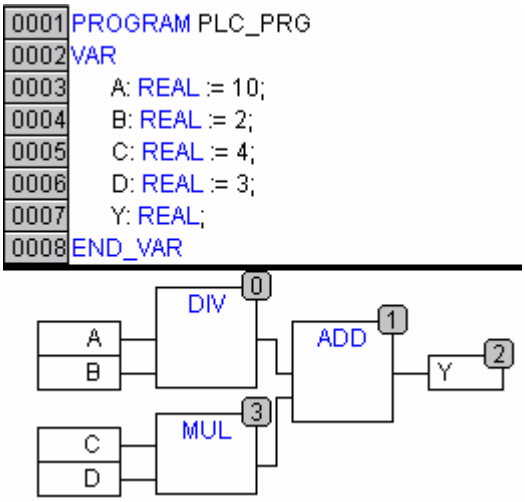

Рис. 5. Пример программы на языке CFC.

Язык SFC (Sequential Function Chart, язык диаграмм состояний), также относится к языкам высокого уровня, при этом он довольно значительно отличается от описанных выше языков. Программа на нем представляет собой совокупность шагов и условий перехода на следующий шаг. Условия или действия во время шагов могут быть записаны на любом

языке, как на SFC, так и на каком-либо другом. Пример программы на языке SFC показан на рис. 6, к сожалению, он не столь нагляден, как в предыдущих случаях, потому что действия, находящиеся внутри шагов не выводятся на экран без вызова. Необходимо отметить, что для разработки программ на языке SFC должна быть подключена библиотека iecsfc.lib.

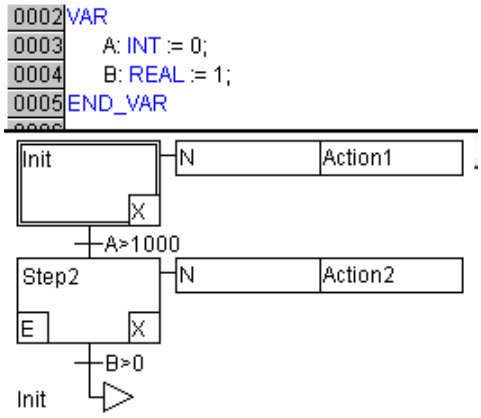

Рис. 6 Пример задачи на языке SFC.

Наиболее часто для реализации проектов в рамках изучаемых в ИТАЭ и ИПЭЭФ курсов, как показывает практика, применяются языки ST, LD, FBD и CFC.

Программа InstallTarget. Установка target-файлов

С помощью комплекса CoDeSys можно программировать любой контроллер, в котором его производителем установлена система исполнения CoDeSys SP. Кроме того, для данного типа контроллера должен быть target-файл от фирмы-производителя. В этом файле находится информация о ресурсах контроллера. При необходимости написать для данного типа контроллеров проект с помощью CoDeSys, следует установить соответствующий target-файл. Это делается с помощью программы InstallTarget, которую можно найти по следующему пути: Пуск/Все программы/3S Software/CoDeSys V2.3/InstallTarget (рис. 7).

При запуске программы InstallTarget появится диалоговое окно, показанное на рис. 8. В левом поле «Possible Targets» указаны target-файлы, которые можно установить (с жесткого или съемного диска), их можно выбрать, нажав кнопку «Open». В правом поле «Installed Targets» указаны инсталлированные файлы.

Перед тем, как проинсталлировать файл, необходимо в поле «Installation directory» указать папку, в которую он будет установлен. Можно выбрать папку, нажав кнопку справа от поля ввода и отметив

требуемую директорию в открывшемся окне «Choose Installation Directory» (выбор папки для инсталляции, рис. 9).

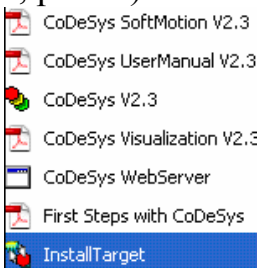

Рис. 7. Расположение программы InstallTarget.

На рис. 8 для инсталляции выбран target-файл PLC150.U-L, который будет установлен в папку с адресом «C:\CoDeSys\Targets…».

Для начала установки необходимо выбрать target-файл в левом окне и нажать кнопку «Install». В результате инсталляции данный файл появится в поле «Installed Targets» (рис. 10). Если требуется удалить target-файл, следует выбрать его в правом окне и нажать кнопку «Remove».

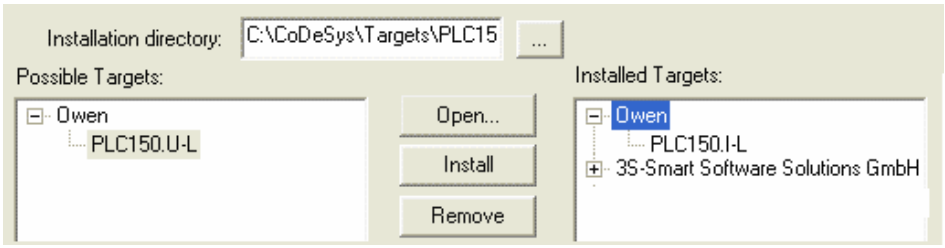

Рис. 8. Окно программы InstallTarget.

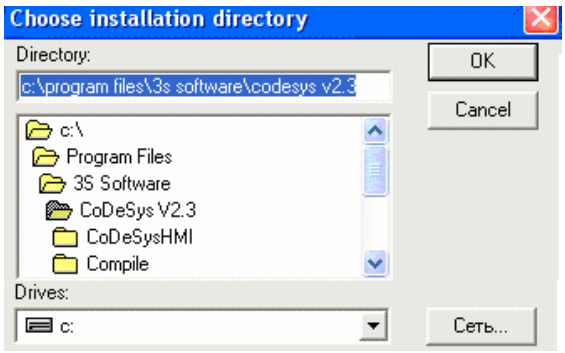

Рис. 9. Выбор папки для инсталляции target-файла.

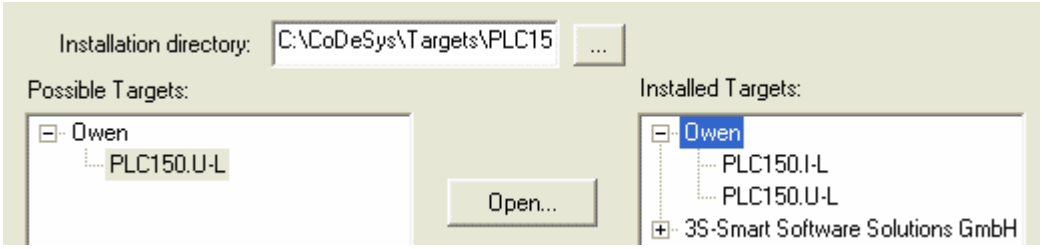

Рис. 10. Результат инсталляции target-файла.

Создание нового проекта

Программы в CoDeSys создаются в виде файлов проектов xxx.pro с помощью компонентов организации программ, обозначаемых как POU (Program Organization Unit). Чтобы создать новый проект, следует войти в пункт меню File/New (Файл/Создать). Появится диалоговое окно Target Settings (Настройка целевой платформы, рис. 11), в котором необходимо указать тип контроллера, для которого создается проект. Если проект не предполагается загружать в контроллер, то выбирается пункт None. Чтобы тип контроллера появился в списке, необходимо предварительно установить его target-файл, как это рассмотрено выше.

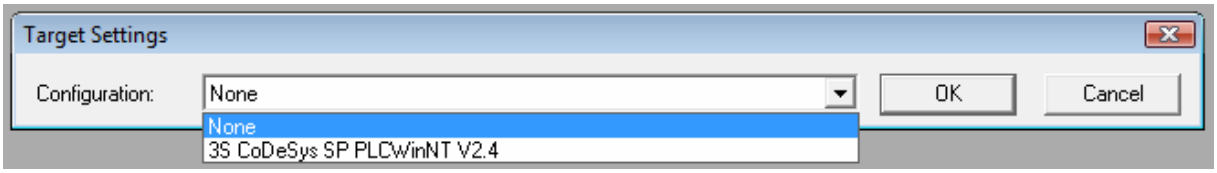

Рис. 11. Настройка целевой платформы.

После выбора целевой платформы на экране появится окно New POU (Новый программный компонент (POU), рис. 12), в котором требуется указать имя нового POU, выбрать его тип и язык реализации.

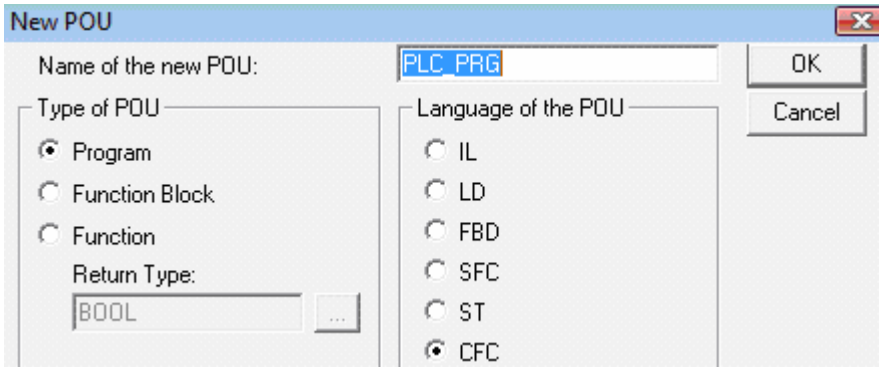

Рис. 12. Выбор вида программного компонента.

В CoDeSys существует всего три типа POU: «Программа» (Program), «Функциональный блок» (Function Block) и «Функция» (Function).

Если создаваемый проект предполагается загружать в контроллер, то в нем должен быть хотя бы один компонент «Программа» с именем PLC_PRG (это имя предлагается по умолчанию). Все остальные POU можно называть произвольно с учетом следующих требований: имя должно состоять только из латинских букв, цифр или знаков подчеркивания, при этом первым символом в имени должна быть буква. Имена POU в составе одного проекта не должны повторяться. Те же требования предъявляются к именам переменных и визуализаций. Если

создаваемый компонент является функцией, то нужно выбрать тип возвращаемого функцией значения и указать его в поле Return Type (тип возвращаемого значения). По умолчанию задается тип BOOL, то есть логический. Можно нажать на кнопку слева от поля Return Type и выбрать тип из открывающегося списка. После ввода необходимой информации следует нажать кнопку «OK», при этом диалоговое окно New POU закроется и откроется редактор кода выбранного языка программирования.

Структура проекта CoDeSys, меню, вкладки. Запуск проекта.

При создании (открытии) проекта в CoDeSys появляется окно проекта (рис.13), в верхней части которого находится меню, а под ним - панель инструментов с учетом выбранного языка. В левой части окна находится вертикальная панель, под которой расположены четыре вкладки. Справа, последовательно сверху вниз, находятся область объявления переменных, область кода выбранного языка и область сообщений об ошибках и предупреждениях при компиляции проекта.

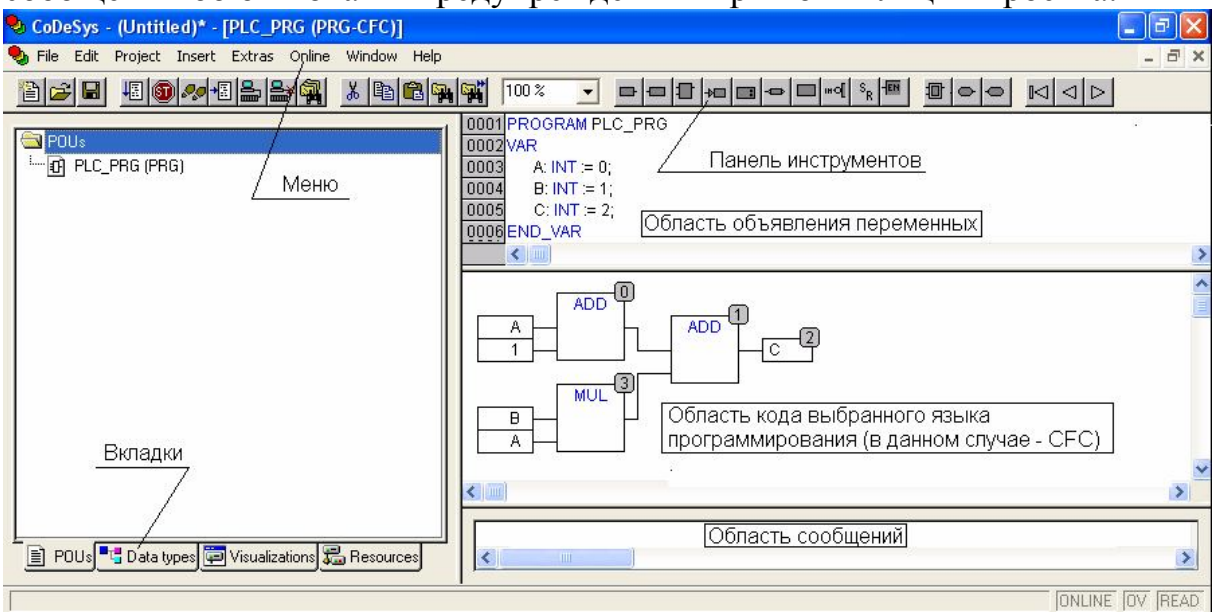

Рис. 13. Окно проекта CoDeSys.

Вкладки проекта CoDeSys отдельно показаны на рис. 14.

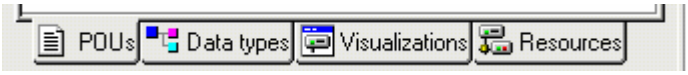

Рис. 14. Вкладки CoDeSys.

Первая слева вкладка – «POU», то есть компоненты проекта. При входе в эту вкладку на находящейся над ней вертикальной панели (Object Organizer - список объектов) отображаются все компоненты проекта (на рис. 13 компонент только один – программа (PRG) с именем PLC_PRG).

При щелчке левой кнопкой мыши по имени POU можно войти в редактор кода, соответствующий языку данного компонента. При щелчке правой кнопкой мыши по имени POU или по пустому полю вертикальной панели появляется меню, с помощью которого можно добавить новый POU, переименовать или удалить уже имеющийся и т.д.

Следующая вкладка – Data types (Типы данных) – предназначена для определения пользовательских типов данных, например, перечислимых типов данных и структур. Более подробное рассмотрение этих типов выходит за рамки данного пособия.

Вкладка Visualizations (Визуализации) предназначена для создания и просмотра операторских интерфейсов (визуализаций). При входе в эту вкладку на левой панели отображаются имеющиеся визуализации. При щелчке левой кнопкой мыши по имени визуализации можно войти в нее, при щелчке правой кнопкой - появляется контекстное меню.

Вкладка Resouces (Ресурсы) позволяет выполнить действия, связанные с ресурсами контроллера. Через нее можно войти в «Настройку целевой платформы», выполнить конфигурирование входов и выходов ПЛК, изменить прошивку ПЛК, войти в «Менеджер библиотек», и т.д.

Работа с вкладками Visualizations и Resouces рассматривается ниже.

Часть пунктов меню универсальна для всех языков, встроенных в CoDeSys, часть из них специфична для конкретного языка. Рассмотрим здесь наиболее важные пункты, общие для всех языков.

В пункте меню «Проект» (Project) имеются два первых элемента «Компилировать» (Build) и «Компилировать все» (Rebuild all) (см. рис. 15).

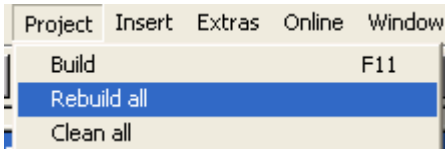

Рис. 15. Пункт меню «Проект».

При использовании элемента «Компилировать» компилируется только новая часть проекта. При обращении к элементу «Компилировать все» компилируется весь проект. В случае успешной компиляции в области сообщений появляется запись «0 Error(s), 0 Warning(s)» - нет ошибок и нет предупреждений. Проект, в котором есть ошибки, не может быть запущен до их полного исправления. При наличии предупреждений проект может работать.

По окончании компиляции можно выполнить загрузку и пуск программы через раздел меню «Онлайн» (Online) (рис.16). Если предполагается загружать программу в ПЛК, соединенным кабелем с ПК, то сначала с помощью элемента «Параметры связи» вводятся данные линии связи компьютера с ПЛК, затем с помощью элемента

«Подключение» (Login) производится их логическое соединение. Если программа является новой, выполняется действие «Загрузить новую программу». Если компьютер не соединен с ПЛК, то в меню «Онлайн» выбирается «Режим эмуляции». До принудительного пуска программа не начинает работу. Пуск программы происходит при обращении к элементу «Старт» (Run) меню «Онлайн» (рис. 17) или к одноименной иконке панели инструментов.

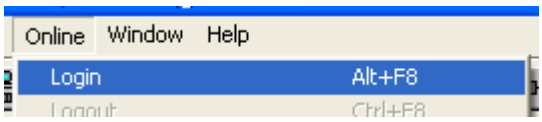

Рис. 16. Пункт меню «Онлайн».

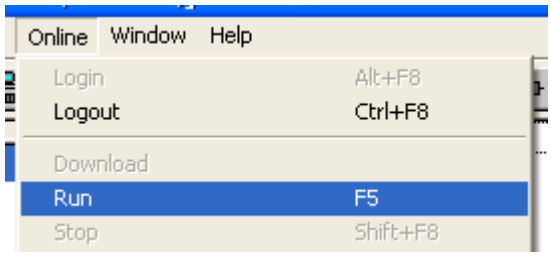

Рис. 17. Пункт меню «Онлайн», запуск программы.

Останова программы происходит при обращении к элементу «Стоп» (Stop) меню «Онлайн» (рис. 18) или к одноименной иконке панели инструментов. В этом случае работа программы прекращается, но компьютер не отключается от контроллера. При необходимости отключиться от компьютера используется подпункт «Отключение» (Logout) (см. рис. 18). Необходимо отметить, что если отключить компьютер от контроллера без останова программы, то в контроллере программа продолжит выполняться, что является нормальным режимом для производственных условий.

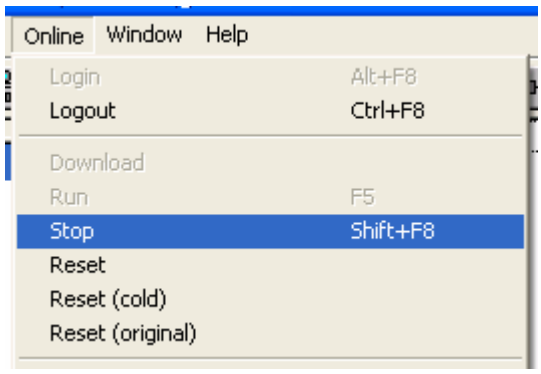

Рис. 18. Пункт меню «Онлайн», останов программы и отключение от контроллера.

Если нужно удалить программу из памяти контроллера, обращаются к подпункту «Сброс» (Reset) (см. рис. 18).

В некоторых ситуациях требуется изменить значения переменных, используемых в программе, не прекращая ее работы. Для этого необходимо открыть вкладку POU и кликнуть левой кнопкой мыши переменную в работающей программе. Появится диалоговое окно, представленное на рис. 19. В поле «Новое значение» (New Value) вводится требуемое значение. Затем следует обратиться к пункту «Записать значения» меню «Онлайн» или нажать F7 (см. рис. 20) после чего в программе будет учитываться новое значение переменной.

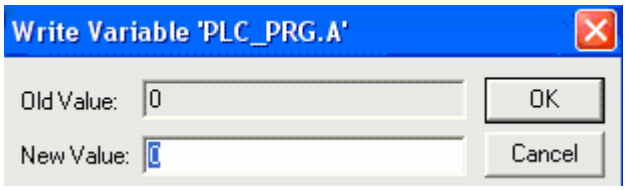

Рис. 19. Диалоговое окно для записи значений переменных.

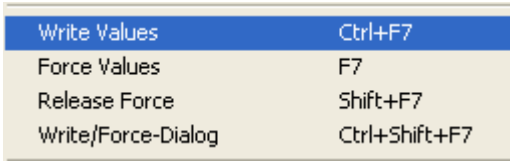

Рис. 20. Пункт меню «Онлайн», запись значений переменных.

Переменные. Объявление переменных

Согласно общепринятому определению, переменная – это поименованная или иным образом адресуемая область памяти. Имя (или адрес) этой области можно использовать для доступа к находящимся в ней данным – то есть, к значению переменной. У любой переменной имеется имя, тип и область действия; переменные в CoDeSys могут быть локальными и глобальными

Областью действия локальной переменной является тот программный компонент, внутри которого она объявлена. Если переменная объявлена внутри функции, то область её действия – эта функция и т.д. Область действия глобальной переменной – весь проект.

В CoDeSys все переменные, связанные со входами или выходами контроллера, являются глобальными по умолчанию. В прочих случаях лучше не пользоваться глобальными переменными без необходимости, так как сложно проследить, как изменяется значение данной переменной при работе программы в целом.

В CoDeSys все используемые переменные должны быть объявлены. Локальные переменные объявляются в области объявления переменных

редактора кода (см. рис. 13). CoDeSys отслеживает переменные, используемые в тексте, и, обнаружив новые необъявленные переменные, предлагает объявить их. При этом выводится окно объявления переменной «Declare Variable» (рис. 21).

В этом окне необходимо ввести имя и тип переменной. По умолчанию указывается тип BOOL. Если переменная имеет другой тип, следует нажать на кнопку справа от поля Type. Откроется окно «Ассистент ввода» (Input assistant), в котором можно выбрать требуемый тип (рис. 22).

В поле Class можно оставить значение по умолчанию (при создании пользовательского функционального блока в этом поле необходимо указать, является ли переменная входной, выходной или внутренней, см. ниже). В поле Address ничего вводить не следует.

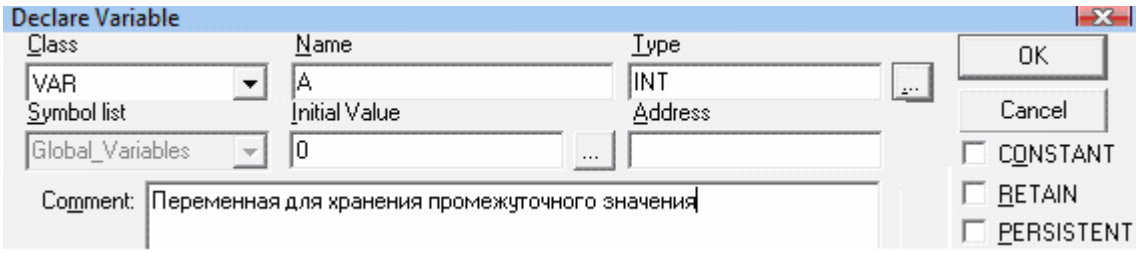

Рис. 21. Окно объявления переменной.

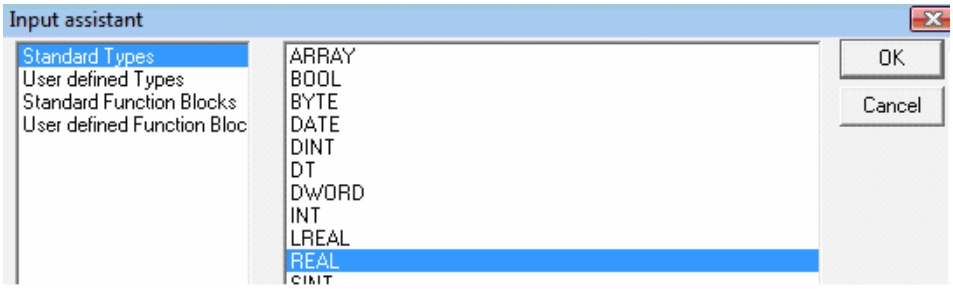

Рис. 22. Ассистент ввода.

В поле Initial value (начальное значение) вводится начальное значение переменной, его можно не изменять. В поле Comment можно ввести комментарий. После нажатия кнопки OK введенные данные отображаются в области объявления переменных редактора кода (Рис. 23).

Содержание области объявления переменных можно создать или изменить вручную. Следует помнить, что весь код, который был в этой области на момент начала работы над проектом, нельзя изменять. Объявления локальных переменных для компонента PROGRAM PLC_PRG всегда пишутся между ограничителями VAR и END_VAR.

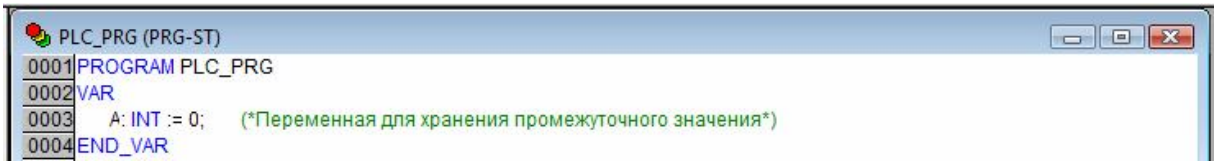

Рис. 23. Область объявления переменных редактора кода.

Глобальные переменные не объявляют в области редактора кода (объявляют в Ресурсах). Имя глобальной переменной не должно нигде повторяться, в том числе и для локальной переменной. Иначе глобальная переменная будет не видна в том компоненте, где объявлена одноименная локальная, что может привести к неправильной работе программы.

В табл.1 приведены основные типы данных, используемые при составлении программ в среде CoDeSys и их краткие описания.

Таблица 1

Основные типы данных, применяемые при составлении программ

Имя типа	Описание типа
INT	Целые числа в диапазоне от -32768 до 32767
REAL	Числа с плавающей точкой, 32 байта
BOOL	Логический тип данных, значения TRUE или FALSE
WORD	Целые числа в диапазоне от 0 до 65535.
TIME	Время (Например, T#5m30s0ms – 5 минут 30 секунд)
STRING	Строка от 1 до 255 символов

Компоненты организации программ

Проект в CoDeSys может содержать три вида компонентов организации программ (POU): программа (Program), функциональный блок (Function Block) и функция (Function). Программа может возвращать несколько значений, ее можно вызывать из других программ или из функциональных блоков, из функций программы вызывать нельзя.

Функции и функциональные блоки могут быть созданы пользователем самостоятельно или взяты из библиотек. Функция отображает множество значений входных параметров на один выход, а функциональный блок – на множество выходов [4].

Чтобы создать пользовательский компонент, следует вызвать вкладку POU, щелкнуть правой кнопкой мыши по левой вертикальной области и в появившемся контекстном меню (рис. 24) выбрать пункт Add object (добавить объект). Появится окно, показанное на рис. 25 (Выбор нового программного компонента).

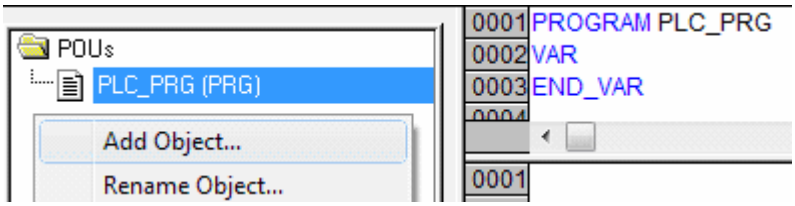

Рис. 24. Добавление нового пользовательского компонента.

Если новый пользовательский компонент – это функция, то необходимо указать не только имя, тип нового компонента и язык реализации, но и тип возвращаемого значения (Return type). В примере на рис.25 тип возвращаемого значения – Real. При нажатии кнопки справа от поля ввода появится окно ассистента ввода, в котором можно выбрать требуемый тип; кроме того, его можно ввести вручную.

После создания нового программного компонента появляется окно редактора кода для выбранного языка. В рассматриваемом примере для функции с именем My_function и типом возвращаемого значения Real область объявления переменных показана на рис. 26. Здесь в области объявления переменных для пользовательской функции есть две подобласти: VAR_INPUT…END_VAR и VAR…END_VAR. При объявлении переменных необходимо указать, к какому классу (VAR или VAR_INPUT) относится объявляемая переменная (Рис. 27).

Если переменная относится к классу VAR_INPUT, то эта переменная будет соответствовать одной из независимых переменных данной функции (т.е., одному из входных параметров). Если переменная относится к классу VAR, то она будет внутренней переменной данного компонента. Выходу, то есть, собственно значению функции, соответствует переменная с именем, совпадающим с именем функции.

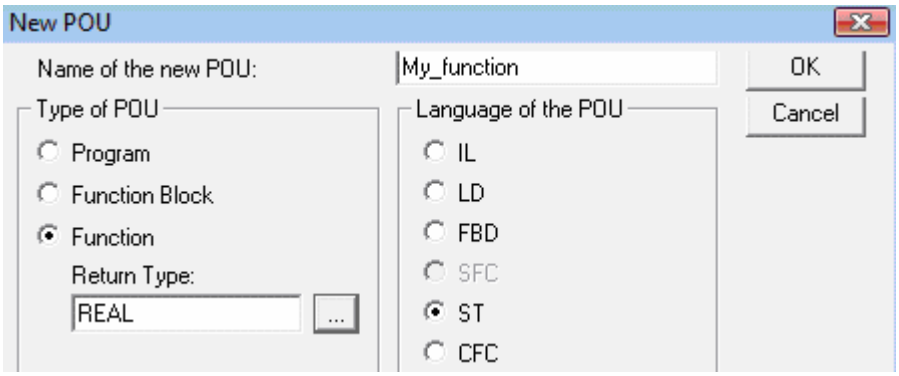

Рис. 25. Создание нового программного компонента Function.

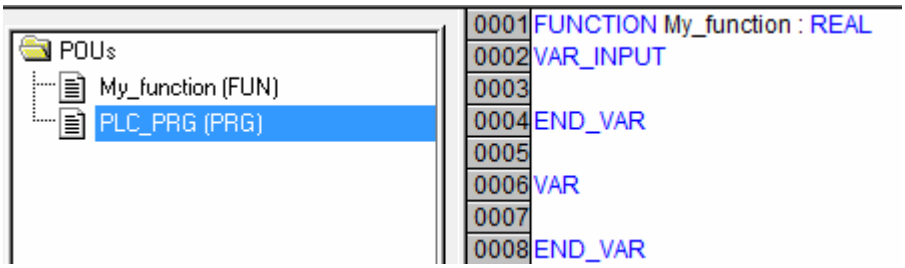

Рис. 26. Область объявления переменных для компонента Function.

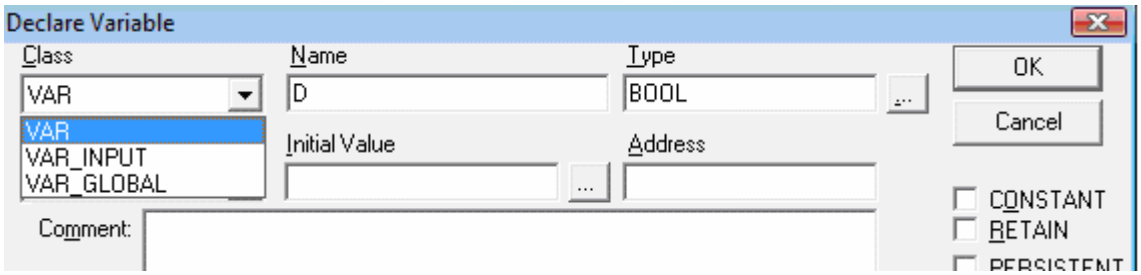

Рис. 27. Окно объявления переменных для компонента Function.

Если новый компонент является функциональным блоком, то тип возвращаемого значения не указывается, так как у функционального блока может быть не один выход. Переменные, используемые внутри пользовательского функционального блока, могут относиться к классам VAR, VAR_INPUT и VAR_OUTPUT. Переменные из класса VAR_INPUT соответствуют входам блока, переменные из класса VAR_OUTPUT – выходам блока, а переменные из класса VAR – внутренние переменные. Имена входов пользовательского блока будут соответствовать именам переменных класса VAR_INPUT, имена выходов – именам переменных класса VAR_OUTPUT. При этом совершенно не обязательно, чтобы имена переменных, связываемых с входами или выходами функционального блока извне, совпадали с именами его входов и выходов.

Все применяемые в программе пользовательские функциональные блоки и функциональные блоки из подключаемых библиотек должны иметь уникальные имена. К ним предъявляются такие же требования, как и к именам переменных. Имя функционального блока записывается на месте трех вопросительных знаков, находящихся над блоком.

Функциональный блок может быть написан на любом языке, встроенном в CoDeSys, функция – на любом языке, кроме SFC.

Пример создания пользовательского функционального блока

В качестве примера рассмотрим создание пользовательского функционального блока для моделирования апериодического звена с помощью разностного выражения:

$$y_i = \frac{\Delta t}{T} k x_{i-1} + (\frac{T}{T + \Delta t}) y_{i-1}, \quad (1)$$

где: y – выход апериодического звена; x – вход звена; T – постоянная времени звена, k – коэффициент передачи звена; Δt – шаг по времени. Новый POU «функциональный блок» с именем Azv создадим на языке ST.

Пусть у функционального блока Azv будет один выход (выход апериодического звена) и четыре входа: шаг по времени, коэффициент передачи, входное воздействие и постоянная времени. Шаг по времени Δt должен соответствовать значению, заданному в ресурсах контроллера.

На рис. 28 показан внутренний код для функционального блока Azv на языке ST. Все переменные, соответствующие входам, находятся в классе VAR_INPUT, соответствующие выходам - в классе VAR_OUTPUT.

Для использования функционального блока в программе необходимо создать его экземпляр с уникальным именем. Сначала вставим экземпляр ФБ типа Azv в область кода, как показано на рис. 29. Вместо трех знаков вопроса над экземпляром блока задаем ему уникальное имя «A1». При введении имени появляется окно объявления переменной (рис. 30). В нем можно ничего не изменять и просто нажать кнопку «ОК».

Программа на языке CFC с использованием данного экземпляра блока показана на рис. 31, состояние программы после пуска - на рис. 32.

```
0001 FUNCTION_BLOCK Azv
0002 VAR_INPUT
0003     delta: REAL := 0.001;      (*Шаг по времени*)
0004     k: REAL := 1;              (*Коэффициент передачи апериодического звена*)
0005     x: REAL := 1;              (*Вход апериодического звена*)
0006     T: REAL := 1;              (*Постоянная времени апериодического звена*)
0007 END_VAR
0008 VAR_OUTPUT
0009     y: REAL := 0;              (*Выход апериодического звена*)
0010 END_VAR
0011 VAR
0012 END_VAR
0001 y:=delta*k*x/T+(1-delta/T)*y;
```

Рис. 28. Внутренний код пользовательского функционального блока Azv.

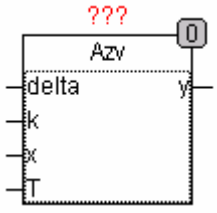

Рис. 29. Вставка пользовательского функционального блока в область кода, начальный этап.

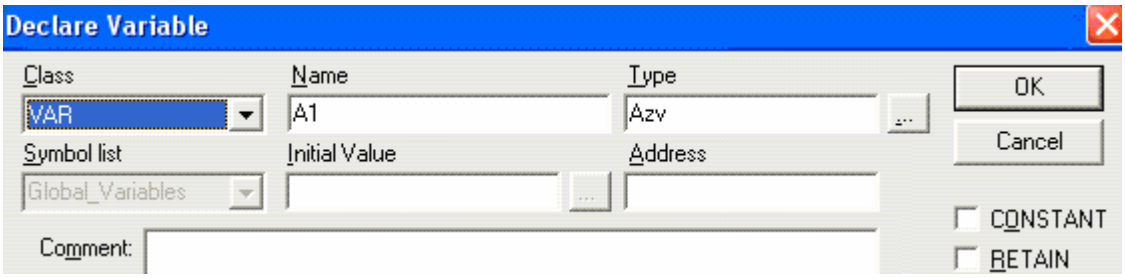

Рис. 30. Объявление экземпляра пользовательского функционального блока.

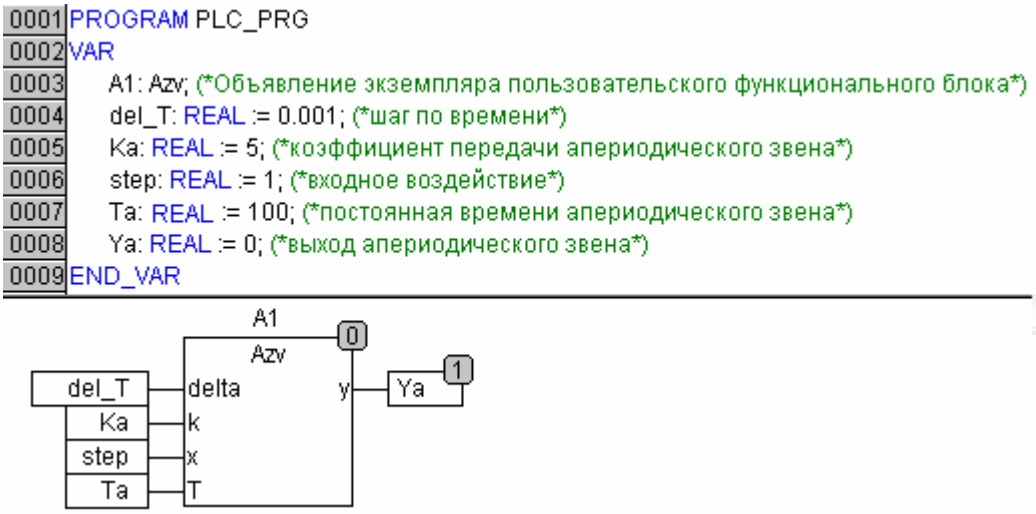

Рис. 31. Программе на языке CFC с экземпляром A1 функционального блока Azv

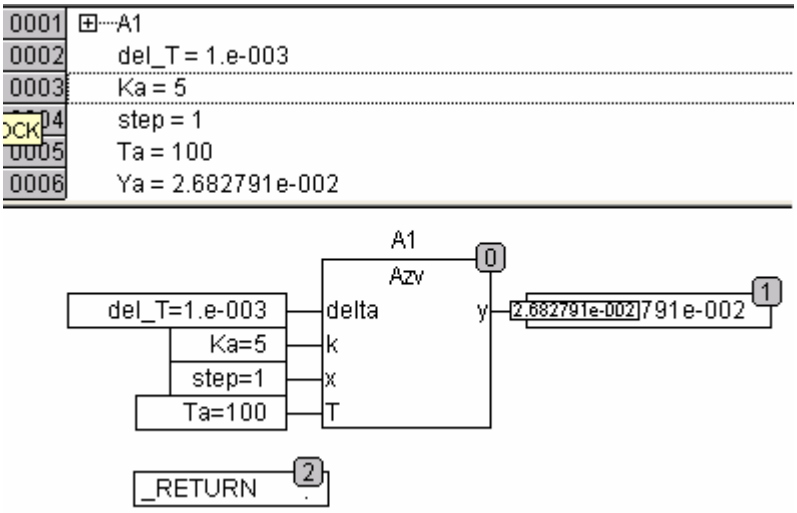

Рис. 32. Состояние программы с экземпляром A1 блока Azv после пуска

Подключение библиотек

Для подключения библиотек нужно зайти во вкладку «Ресурсы» (Resources) и левой кнопкой мыши открыть «Library Manager» (Менеджер библиотек). Далее кликом правой кнопки мыши в поле библиотек открыть

контекстное меню, войти в пункт «Добавить библиотеку» (Ins) и в появившемся окне Open (Открыть) выбрать требуемый файл типа xxx.lib.

Окно менеджера библиотек показано на рис. 33. Оно разделено на четыре части. В левой верхней части выводится список уже подключенных библиотек. В левой нижней части представлен список компонентов отмеченной библиотеки (на рис.33 это Util.lib). В правой верхней части выводится список локальных переменных отмеченного компонента (на рис.33 это блок PID). В правой нижней части показан отмеченный компонент со всеми входами и выходами. Таким образом, это окно можно использовать как справочный материал по данной библиотеке.

Если щелкнуть правой кнопкой мыши под списком библиотек, то появится контекстное меню (рис. 34). В нем предусмотрены следующие действия: добавить еще одну библиотеку (Ins), удалить (Del) или узнать свойства отмеченной библиотеки (Properties). В свойствах, например, указывается имя файла библиотеки, место его расположения и дата последнего изменения. Если файл библиотеки был перемещен, то с помощью пункта Properties можно указать его новый адрес.

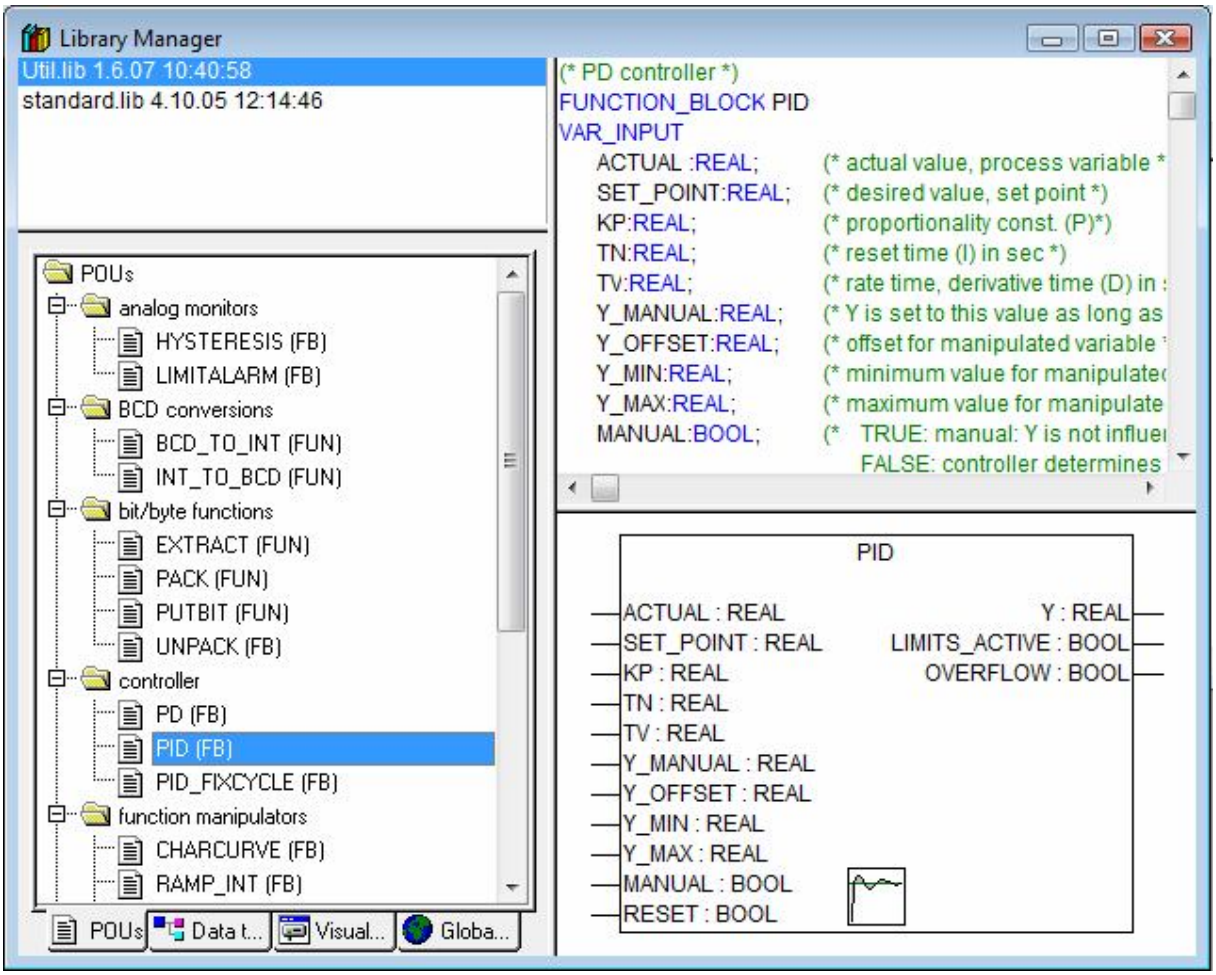

Рис. 33. Окно менеджера библиотек.

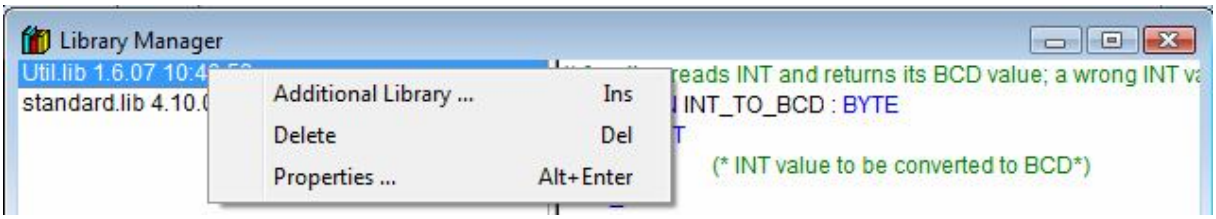

Рис. 34. Контекстное меню для менеджера библиотек.

Некоторые операции, функциональные блоки и подключаемые библиотеки

Рассмотрим ряд стандартных операций и блоков, используемых в CoDeSys. В табл.2 приведены некоторые функции и арифметические операции; в табл.3 - некоторые логические функции и функции выбора; в табл.4 - некоторые функции преобразования типов данных. Данный обзор не является полным. При необходимости следует обращаться к более обширным источникам, например, [2], [4] или справочной системе CoDeSys.

Таблица 2

Функции и арифметические операции

№	FBD,CFC	ST	Описание
1	ADD	+	Сложение
2	SUB	-	Вычитание
3	DIV	/	Деление
4	MUL	*	Умножение
5	SIN	SIN()	Синус угла
6	COS	COS()	Косинус угла
7	TAN	TAN()	Тангенс угла
8	ASIN	ASIN()	Арксинус числа
9	ACOS	ACOS()	Арккосинус числа
10	ATAN	ATAN()	Арктангенс от числа
11	EXP	EXP()	Экспонента от числа
12	LN	LN()	Натуральный логарифм числа
13	LOG	LOG()	Десятичный логарифм числа
14	SQRT	SQRT()	Корень квадратный из числа
15	ABS	ABS()	Модуль числа

Таблица 3

Логические функции и функции выбора

№	FBD,CFC	ST	Описание
1	AND	AND()	Логическое «И» (конъюнкция)
2	OR	OR()	Логическое «ИЛИ» (дизъюнкция)
3	NOT	NOT()	Логическое «НЕ» (инверсия)
4	MIN	MIN()	Выбор минимального из двух чисел
5	MAX	MAX()	Выбор максимального из двух чисел
6	GT	>	Значение выхода TRUE, если x1>x2
7	LT	<	Значение выхода TRUE, если x1<x2
8	GE	>=	Значение выхода TRUE, если x1≥x2
9	LE	<=	Значение выхода TRUE, если x1≤x2
10	EQ	=	Значение выхода TRUE, если x1=x2

Таблица 4

Функции преобразования типов данных

№	FBD,CFC	ST	Описание преобразования
1	BOOL_TO_WORD	BOOL_TO_WORD()	Из BOOL в WORD
2	BOOL_TO_INT	BOOL_TO_INT()	Из BOOL в INT
3	BOOL_TO_STRING	BOOL_TO_STRING()	Из BOOL в STRING
4	REAL_TO_WORD	REAL_TO_WORD()	Из REAL в WORD
5	REAL_TO_INT	REAL_TO_INT()	Из REAL в INT
6	STRING_TO_BOOL	STRING_TO_BOOL()	Из STRING в BOOL
7	STRING_TO_WORD	STRING_TO_WORD()	Из STRING в WORD

Рассмотрим несколько функциональных блоков из библиотек Util.lib и PID_Regulators.lib. Сначала рассмотрим функциональный блок HYSTERESIS библиотекb Util.lib. С помощью этого блока можно реализовать двухпозиционный регулятор с зоной гистерезиса [7]. Входы и выходы этого блока описаны в табл.5.

Таблица 5

Функциональный блок HYSTERESIS

№	Наименование	Тип	Описание
1	IN	INT	Входная величина регулятора
2	HIGH	INT	Верхнее пороговое значение
3	LOW	INT	Нижнее пороговое значение
1	OUT	BOOL	Выход блока. OUT=TRUE, если IN < LOW, и OUT= FALSE, если IN > HIGH.

Пример программы с использованием блока HYSTERESIS приведен на рис. 35. Это программа двухпозиционного регулирования температуры электронагревателя [7]. У экземпляра блока HYSTERESIS имеется уникальное имя (Hyst1).

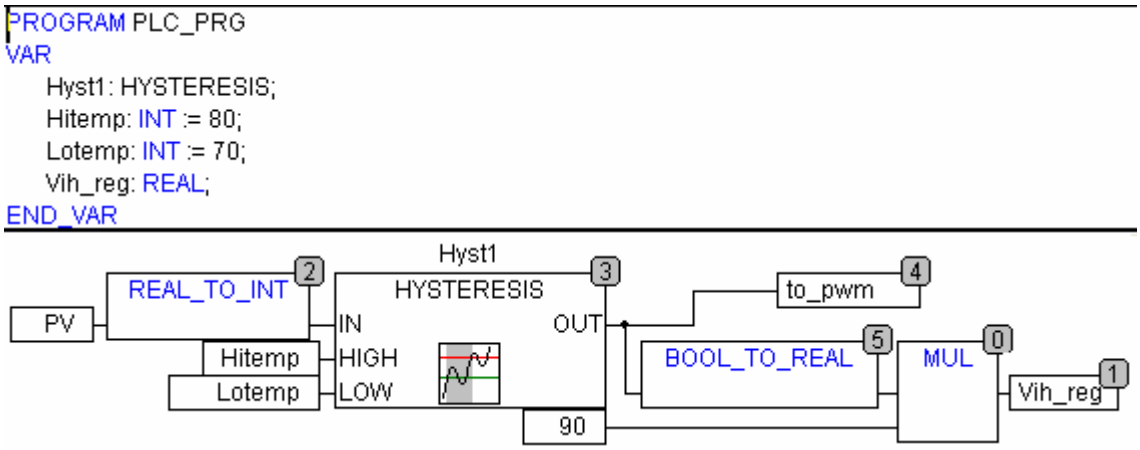

Рис. 35. Пример программы с использованием блока HYSTERESIS.

Двухпозиционный регулятор можно также реализовать с помощью блока ON_OFF_HIST_REG из библиотеки PID_Regulators.lib. Описание его входов и выходов приведено в табл.6.

Таблица 6

Входы и выходы функционального блока ON_OFF_HIST_REG

№	Наименование	Тип	Описание
Входы			
1	PV	REAL	Регулируемая переменная
2	SP	REAL	Уставка (задание) регулятора
3	HYST	REAL	Гистерезис (зона возврата)
4	DB	REAL	Порог срабатывания (смещение)
Выходы			
1	COOLER	BOOL	Реле включения «холодильника»
2	HEATER	BOOL	Реле включения «нагревателя»

При разработке систем регулирования могут потребоваться типовые линейные алгоритмы регулирования, например, П, ПИ и ПИД [6]. Для этого можно включить в проект экземпляры блоков PID и PD из библиотеки Util.lib. Входы и выходы блока PID описаны в табл.7, входы и выходы блока PD аналогичны, за исключением того, что у него отсутствует вход TN и выход OVERFLOW (так как в алгоритме нет И-звена).

Таблица 7

Описание входов и выходов блока PID

Входы		
Наименование	**Тип**	**Описание**
ACTUAL	REAL	Регулируемая переменная
SET_POINT	REAL	Задание
KP	REAL	Коэффициент передачи регулятора
TN	REAL	Постоянная времени интегрирования
TV	REAL	Постоянная времени дифференцирования
Y_MANYAL	REAL	Определяет Y, если MANUAL=TRUE
Y_OFFSET	REAL	Определяет Y при сбросе регулятора
Y_MIN, Y_MAX	REAL	Ограничения для Y
MANUAL	BOOL	Если TRUE, то режим «Руч»
RESET	BOOL	Если TRUE, то сброс регулятора
Выходы		
Наименование	**Тип**	**Описание**
Y	REAL	Выход регулятора
LIMITS_ACTIVE	BOOL	TRUE, если Y<Y_MIN или Y>Y_MAX
OVERFLOW	BOOL	TRUE, если переполнение

П-алгоритм регулирования можно реализовать на блоке PD при TD=0, ПИ-алгоритм - на блоке PID при TD=0. Пример программы с использованием блока PID [6] приведен на рис. 36.

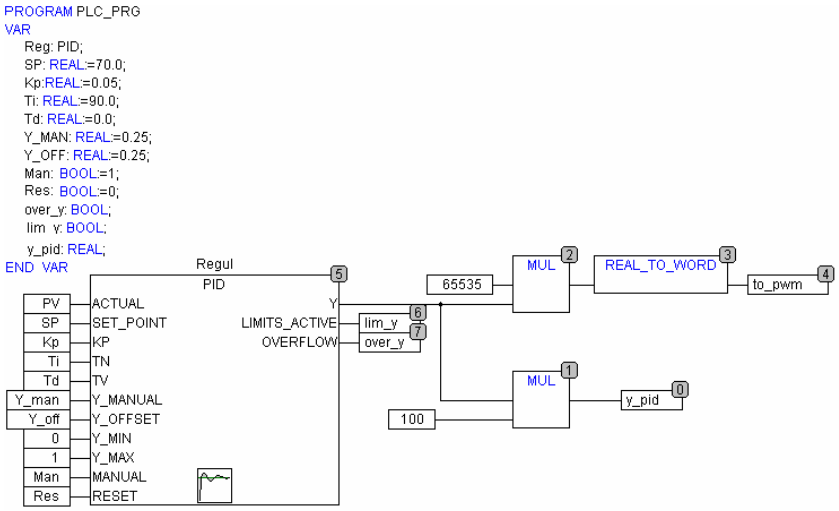

Рис. 36. Пример реализации ПИ-алгоритма регулирования.

В качестве выходного устройства здесь используется дискретный выход ПЛК, связанный с выходом регулятора через двухпозиционный ШИМ (широтно-импульсный модулятор, англ. - PWM), который изменяет скважность управляющих импульсов и среднюю мощность нагревателя в

соответствии с переменной «to_pwm». В случае двухпозиционного регулирования выход HEATER регулятора непосредственно привязывается к дискретному выходу ПЛК.

Рассмотрим далее переключатель с доминантой включения SR (триггер), являющийся компонентом библиотеки Standard.lib. Описание приведено в табл.8. Может использоваться для фиксации состояния при переключении режима работы, например, «Руч»-«Авт», при управлении импульсными сигналами Man_set и Auto_set от кнопок без фиксации (Рис.37).

Таблица 8

Входы и выходы переключателя SR

№	Наименование	Тип	Описание
1	SET1	BOOL	Вход 1
2	RESET	BOOL	Вход 2
3	Q1	BOOL	Выход: Q1 = (NOT RESET AND Q1) OR SET1

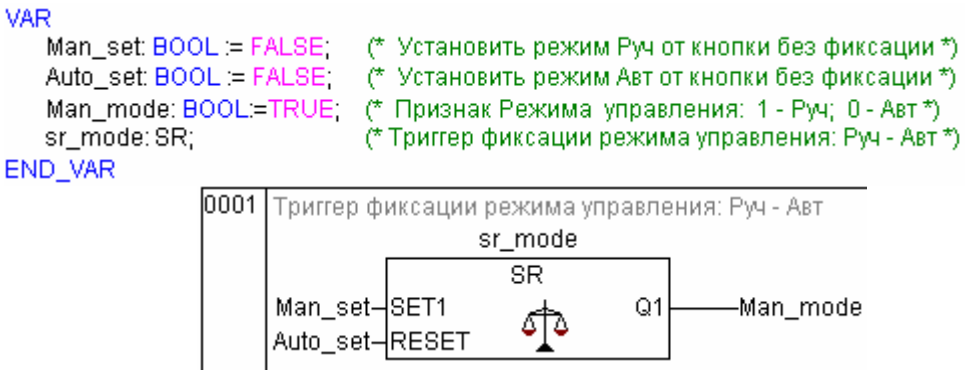

Рис. 37. Пример переключателя «Руч»-«Авт» с компонентом SR

Создание визуализаций

Можно сказать, что визуализация применительно к задаче автоматизации – это представление в виде изображения информации о технологическом процессе и создание виртуальных средств оперативного управления. Для создания визуализации в CoDeSys следует перейти во вкладку «Визуализации», кликнуть правой кнопкой мыши на левой вертикальной панели и через контекстное меню создать объект визуализации с уникальным именем.

Значительная часть открывшегося окна (справа внизу) занята полем для «рисования», т.е., для создания визуализации, над ним находится панель инструментов, представленная на рис. 38. Рассмотрим основные инструменты и конфигурирование полученных объектов.

Рис. 38. Панель инструментов для создания визуализации.

Первая слева кнопка (стрелка) предназначена для переключения на режим курсора мыши. Следующие семь кнопок предназначены для рисования линий и геометрических фигур. Эти фигуры можно использовать, в частности, для создания цифровых табло с функциями ввода-вывода. Создадим прямоугольник в области рисования и рассмотрим основные этапы его конфигурировании (рис. 39).

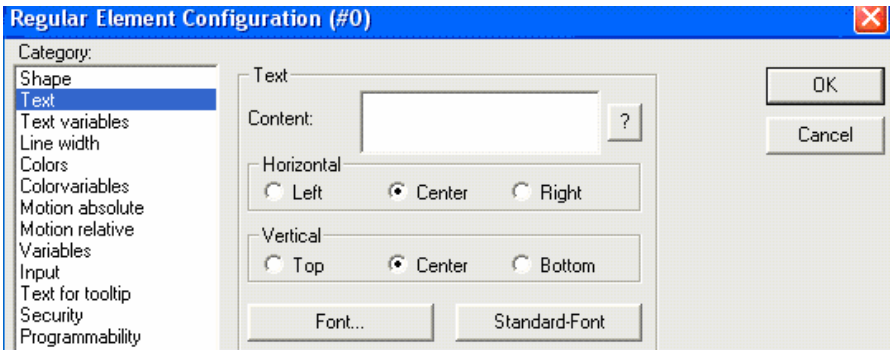

Рис. 39. Окно конфигурирования прямоугольника, категория «Текст».

В поле «Строка» (Content) в правой части можно набрать текст, который будет выводиться в прямоугольник. В меню выбора «Горизонтальный» (Horizontal) и «Вертикальный» (Vertical) можно сменить ориентацию и выравнивание текста. При нажатии на кнопку «Шрифт» (Font) появляется стандартное диалоговое окно выбора шрифта. При нажатии на кнопку «Стандартный шрифт» (Standard-Font) устанавливается шрифт по умолчанию.

Кроме стационарного текста в поле «Строка» можно выводить значения переменных. Для этого требуется указать имя переменной, тип и формат выводимого значения. Примеры представлены в табл.9.

Таблица 9

Типы и форматы значений, выводимых в поле «Строка»

№	Запись	Описание результата
1	%s	Значение переменной выводится в виде строки символов
2	%d	Вывод целого числа
3	%f	Вывод числа с плавающей точкой, количество символов в целой и дробной частях неограничено
4.	%3.2f	Вывод числа с плавающей точкой, три символа в целой части и два символа в дробной (количество символов зависит от указанных чисел)

Для указания имени переменной следует войти в категорию «Переменные» (Variables) и указать полное имя переменной в поле «Выв. текста» (Textdisplay) (рис. 40). Имя переменной необходимо указывать полное, то есть, с именем компонента. Чтобы не ошибиться, проще всего установить в это поле курсор и нажать F2, после чего выбрать переменную из появившегося окна «Ассистента ввода». Переменная автоматически запишется с полным именем.

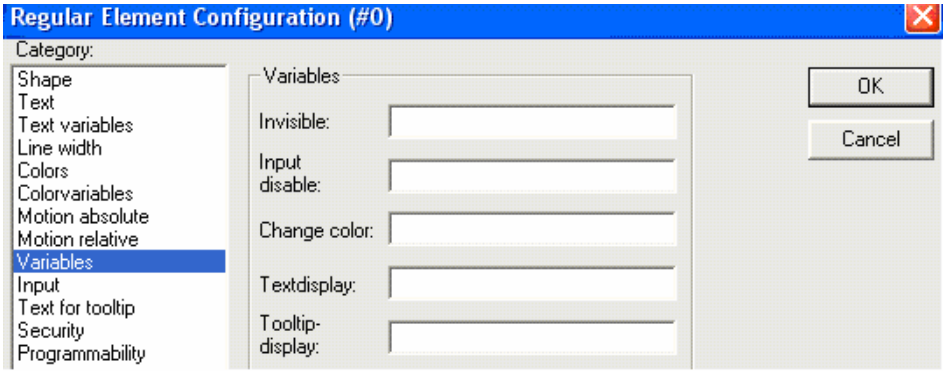

Рис. 40. Конфигурирование прямоугольника, указание имени переменной.

Для организации табло с функцией ввода следует войти в категорию «Ввод» и поставить галочку в поле «Ввод в переменную Выв_текста» (Text input of variable ‘Textdisplay’) (рис. 41). Через меню «Категории» можно также изменить цвет элемента, толщину линии и т.д., аналогично тому, как это делается в любом графическом редакторе.

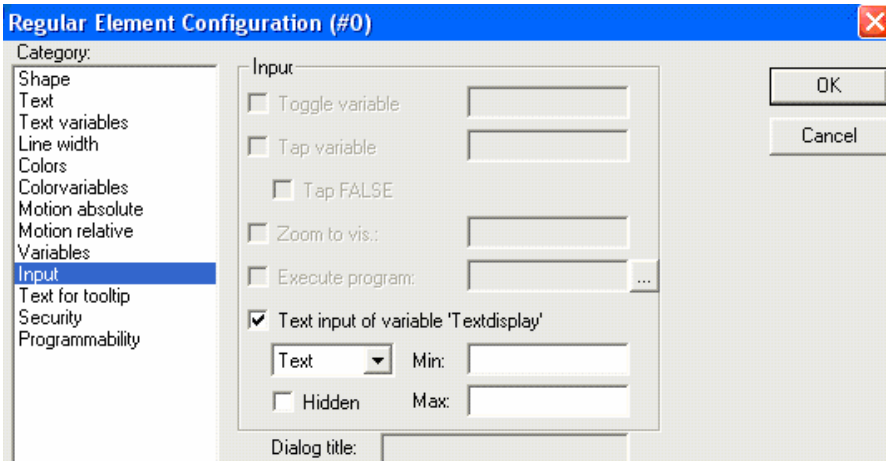

Рис. 41. Конфигурирование прямоугольника. Организация функции ввода.

Кнопки управления в область рисования вводятся с помощью символа кнопки «ОК» на панели инструментов. Часть окна конфигурирования кнопки аналогична окну конфигурирования прямоугольника. На кнопке может быть что-либо написано. Кнопка может,

в принципе, использоваться как цифровое табло ввода-вывода, конфигурирование аналогично предыдущему случаю.

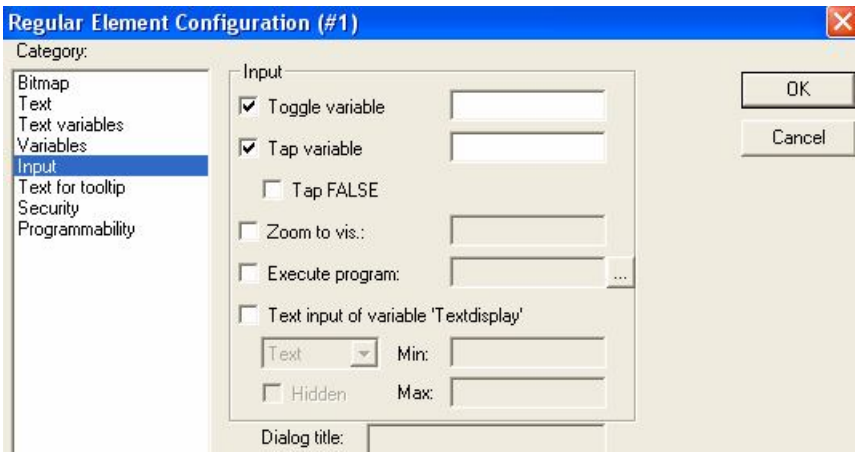

Рис. 42. Конфигурирование элемента «Кнопка».

Кроме этого, можно организовать ввод значений переменных типа BOOL нажатием и отпусканием кнопки (рис. 42). Для этого нужно поставить галочку в поле «Toggle variable» (Переменная переключения) или «Tap variable» (Переменная кнопка), после чего в поле справа ввести имя изменяемой переменной (через ассистент ввода, нажав F2). Вариант «Переменная переключения» создает кнопку с фиксацией, состояние которой можно изменить повторным нажатием. Вариант «Переменная кнопка» создает кнопку без фиксации, которая присваивает переменной значение «TRUE» при нажатии и значение «FALSE» после отпускания. С помощью кнопки можно также запустить какую-либо программу. Для этого нужно поставить галочку в поле «Вып. программы» (Execute program) и в поле справа выбрать имя запускаемой программы.

Через два инструмента вправо от «Кнопки» находится элемент «Тренд», предназначенный для графического отображения изменения переменных во времени. Окно конфигурирования тренда показано на рис.43 [5].

Рис. 43. Конфигурирование графика.

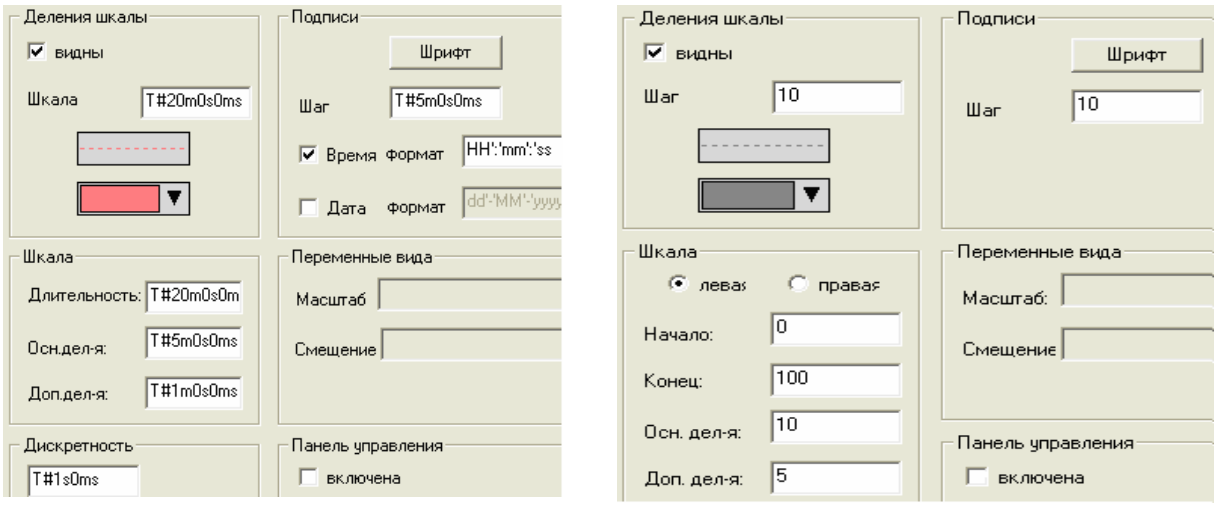

Рис. 44. Конфигурирование горизонтальной и вертикальной осей графика.

Для конфигурирования оси абсцисс (время) или ординат следует нажать кнопку «Горизонтальная ось» или «Вертикальная ось».

Пример конфигурирования горизонтальной и вертикальной осей показан на рис. 44. Количество делений рекомендуется задавать порядка 10, чтобы подписи легко читались. Пределы осей зависят от конкретной задачи. Параметры горизонтальной оси вводятся в формате времени (см. табл.1). В окне конфигурирования горизонтальной оси имеется поле «Дискретность», где указывается шаг записи данных в файл.

Для записи данных в файл в окне конфигурирования категории «Тренд» выбирается панель «Регистрация», вариант «История», далее нажимается кнопка «Настройка». В появившемся окне (рис.45) в поле «Директория» вводят путь к папке для записи, в поле «Имя файла» - имя файла без расширения (автоматически присваивается расширение «trd»).

Для выбора отображаемых на «тренде» переменных следует кликнуть кнопку «Выбор переменной» (рис. 43) и с помощью ассистента ввода выбрать переменные, а также установить цвета графиков.

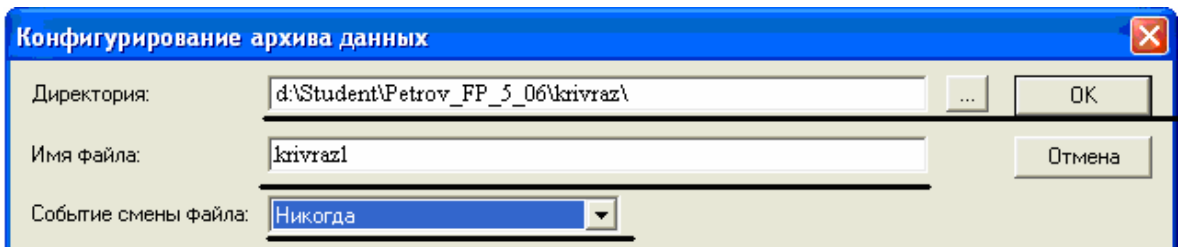

Рис. 45. Конфигурирование архива данных.

Привязка переменных ко входам и выходам ПЛК

У ПЛК могут быть аналоговые и дискретные входы и выходы, информация о которых хранится в его target-файле. Для конфигурирования входов и выходов нужно войти во вкладку «Ресурсы», а в ней – в «Конфигурацию ПЛК». Входы и выходы ПЛК описаны в Таблице 10.

Таблица 10

Входы и выходы ПЛК

Английский	Русский (примеры)
Discrete input	Дискретный вход (сигналы от датчиков-реле)
Discrete output	Дискретный выход (сигналы включения нагревателя или ИМ постоянной скорости)
Unified signal sensor	Аналоговый вход (сигналы аналоговых датчиков, например, преобразователей давления)
Analog output	Аналоговый выход (сигнал к позиционеру)

Рассмотрим для примера конфигурирование аналогового входа, к которому подключен термометр сопротивления.

Для настройки следует кликнуть кнопкой мыши выбранный аналоговый вход (нумерация сверху вниз), выбрать в появившемся меню «Заменить элемент» и выбрать «RTD sensor» (см. рис. 46).

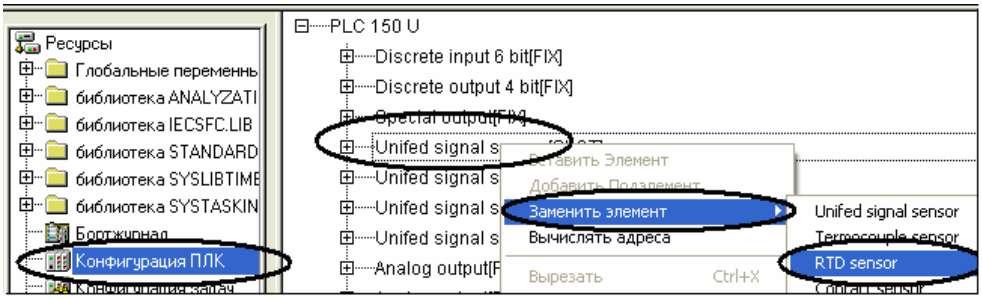

Рис. 46. Конфигурирование аналоговых входов. Выбор типа датчика.

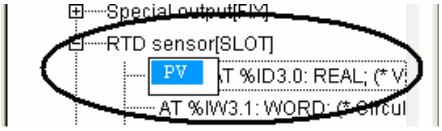

Рисунок а.

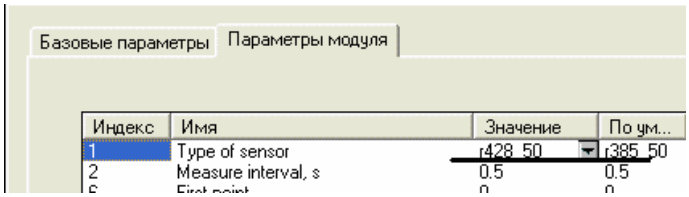

Рисунок б.

Рис. 47. Конфигурирование аналоговых входов. Продолжение.

Далее следует кликнуть «AT» в первой строке настраиваемого входа и ввести имя переменной (PV, рис. 47а). Эта переменная будет глобальной. Затем кликнуть вид датчика («RTD sensor»), открыть «Параметры модуля»

(рис. 47б) и выбрать тип датчика (r428_50). Здесь можно вводить поправки в трех выбранных точках с линейной интерполяцией. Кроме того, указывается время цикла измерения переменной по настраиваемому входу.

Рассмотрим добавление и конфигурирование ШИМ. Для этого открываем дискретные выходы, открываем контекстное меню кликом кнопкой правой кнопки мыши и выбираем «Вставить Pulse-Width Modulator». Появится группа «Pulse-width modulator» под всеми дискретными входами. Далее открываем группу ШИМ и присваиваем имя переменной (на рис. 48 - to_pwm). Затем входим во вкладку «Параметры модуля» и указываем в первой строке выход, к которому подключен ШИМ (следует иметь в виду, что нумерация в CoDeSys начинается с нуля), во второй строке – период ШИМ в сотнях мкс, в третьей – минимальную длительность импульса в сотнях мкс.

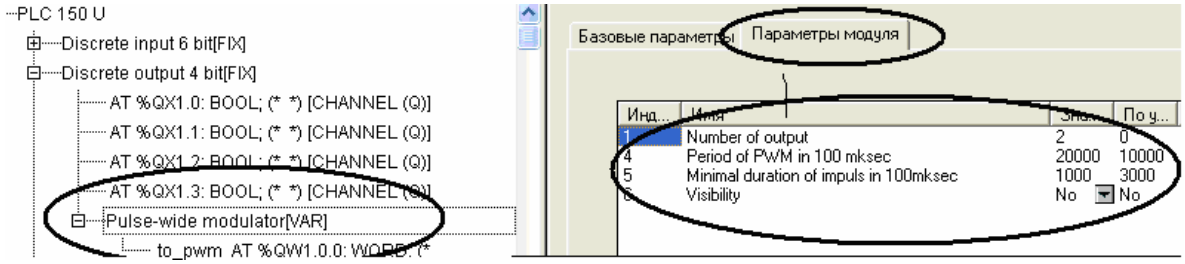

Рис. 48. Конфигурирование ШИМ.

Задание времени цикла выполнения программы.

Для некоторых задач или компонентов требуется задать шаг по времени Δt, через который они будут вызываться на исполнение в контроллере. Чтобы задать этот шаг, следует вызвать «Ресурсы» и кликнуть в верхней части поля название контроллера (например, PLC 150U). В открывшейся панели выбрать «Параметры модуля» и ввести значение в поле «Минимальная продолжительность цикла» (Min CycleLenght ms) в мс, например, 5. Это значение будет определять Δt.

Список литературы

1. Сайт фирмы 3S-Smart Software Solutions GmbH www.3s-software.com.
2. Сайт фирмы НПФ Пролог www.codesys.ru
3. Сайт фирмы ОВЕН www.owen.ru.
4. Петров И.В. Программируемые контроллеры. Стандартные языки и приемы прикладного проектирования. – М.: СОЛОН-ПРЕСС, 2007. – 256 с.

5. Кузищин В.Ф., Мерзликина Е.И. Исследование характеристик объекта управления (электрической печи) на базе контроллера ПЛК 150. Лабораторная работа №1. – М.: Издательский дом МЭИ, 2011. – 12 с.
6. Кузищин В.Ф., Мерзликина Е.И. Исследование АСР температуры электрической печи на базе контроллера ПЛК 150. Лабораторная работа №2. – М.: Издательский дом МЭИ, 2011. – 12 с.
7. Кузищин В.Ф., Мерзликина Е.И. АСР температуры электронагревателя с позиционным алгоритмом регулирования на базе контроллера ПЛК 150. Лабораторная работа №3. – М.: Издательский дом МЭИ, 2012. – 12 с.

Оглавление