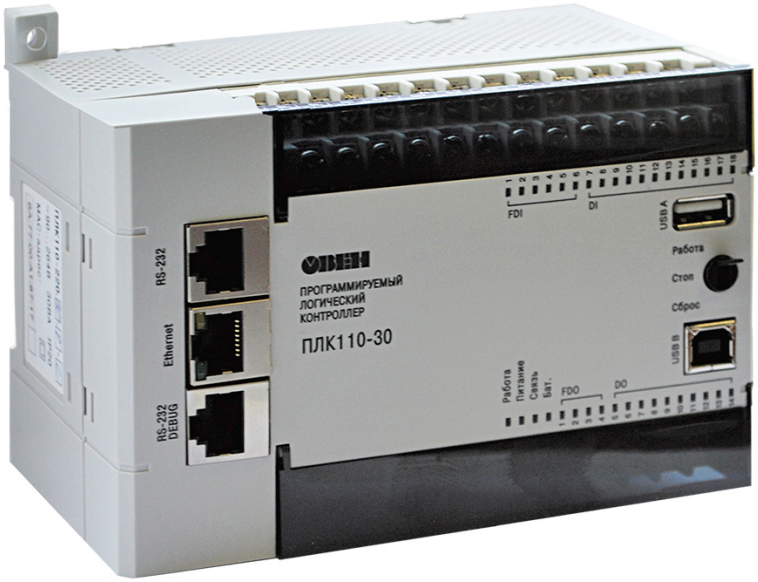
Программирование программируемых логических контроллеров ПЛК110

Руководство пользователя



Регистрационный номер: 16

Москва 2016

Содержание

Введение

В данном руководстве изложены основы процедуры создания рабочей программы для программируемых логических контроллеров ПЛК 110.

1 Использование документа

1.1 Содержание разделов

В данном руководстве изложены основы процедуры создания рабочей программы для программируемых логических контроллеров ПЛК 110.

Первая часть документа (разделы 1 - 3) содержит краткое описание последовательности операций, выполняемых в ходе создания рабочей программы.

Вторая часть документа (разделы 4 - 7) содержит базовую часть информации, требуемой для реализации программ ПЛК, решающих задачи любой сложности.

Третья часть документа (разделы 8 – 10) содержит дополнительную информацию, требуемую при решении определенных задач программирования ПЛК.

Полностью информация, требуемая для создания программ, содержится в Руководстве пользователя программного обеспечения (ПО): создание рабочей программы для программируемого логического контроллера выполняется в ПО CODESYS.

При первоначальном ознакомлении с руководством рекомендуется ознакомиться с содержанием настоящего «Введения» и разделов 1 - 3 документа. В дальнейшем рекомендуется обращаться к разделам второй части документами, содержащим достаточный для работы объем информации по конкретным вопросам.

Информация по установке, вводу в эксплуатацию, обслуживанию и устранению ошибок работы программируемых логических контроллеров содержится в документе «Руководство по эксплуатации ПЛК110».

Процедура программирования ПЛК включает следующие этапы:

1) Предварительный этап: установка операционной системы и ПО (среды программирования) CODESYS (**Co**ntroller **De**velopment **Sys**tem).
2) Выбор контроллера. Установка требуемого файла настроек целевой платформы (target-файла).
3) Создание и отладка проекта.
4) Установление связи с контроллером. При установке связи ПО CODESYS автоматически компилирует проект и предлагает загрузку скомпилированного кода в ОЗУ контроллера.
5) Запуск выполнения проекта (пользовательской программы ПЛК), проверка ее работоспособности и, при необходимости, отладка.
6) В случае корректной работы проекта (пользовательской программы ПЛК) – сохранение ее в энергонезависимой Flash-памяти контроллера для последующей загрузки и выполнения при включении питания ПЛК. В случае некорректной работы проекта – возврат на этап 5 (в процессе отладки проекта перечисленные выше операции могут выполняться многократно).

Программирование ПЛК рекомендуется выполнять до монтажа контроллера на объекте, но можно выполнить его и после монтажа.

Внимание! **Фрагменты текста, выделенные в документе аналогично данному фрагменту, содержат критически важную информацию, на которую рекомендуется обратить особое внимание.**

Внимание! Фрагменты текста, выделенные в документе аналогично данному фрагменту, содержат важную информацию, на которую рекомендуется обратить особое внимание.

1.2 Термины и аббревиатуры

Определения основных терминов и расшифровка аббревиатур, используемых в тексте данного документа, приведены в таблице 1.1.

При работе с документацией следует обратить внимание на то, что терминология, используемая в интерфейсах и документации ПО CODESYS, специфична и не всегда соответствует требованиям стандартов ЕСПД. Так, режимы выполнения программного обеспечения – обозначаются различными терминами. Например, режимы редактирования текстов программ именуются «редакторы»; режимы редактирования объектов программного обеспечения, отнесённых в интерфейсе ПО CODESYS к «ресурсам» – именуются «ресурсами» (например, режим редактирования конфигурации ПЛК – «Ресурсом “Конфигурация ПЛК”») и т.п. Но описания режимов работы ПО проиллюстрированы и достаточно подробны, чтобы разнобой в терминологии не мог повлиять на понимание текста.

Таблица 1.1 – Термины и аббревиатуры

Термины и аббревиатуры	Определения и расшифровки
ПК	Персональный компьютер
ПЛК	Программируемый логический контроллер
ПО	Программное обеспечение
ОЗУ	Оперативное запоминающее устройство
Файл настроек целевой платформы (Target file)	Файл, поставляемый производителем ПЛК и описывающий аппаратные и программные особенности конкретного ПЛК. Обеспечивает корректное взаимодействие программного обеспечения CODESYS и программируемого логического контроллера.
Проект	Пользовательская программа программируемого логического контроллера, разрабатываемая в программном обеспечении CODESYS. После отладки и загрузки в контроллер обеспечивает правильную работу контроллера.
ПО CODESYS	Специализированное программное обеспечение, предназначенное для подготовки пользовательских программ программируемого логического контроллера (CODESYS)
OPC	**O**LE – objectlinkingandembedding – for**P**rocess**C**ontrol, объектное связывание и встраивание для контроля процессов. OPC – открытый для использования набор спецификаций, разработанный организацией OPC Foundation на основе технологий MicrosoftCOM/DCOM.

Окончание таблицы 1.1

Термины и аббревиатуры	Определения и расшифровки
OPC DA	Спецификация Data Access (DA) OPC (см. OPC), которая позволяет читать и писать данные в прибор, организовывать подписку на данные и поредавать клиенту уведомление об обновлении данных
SCADA	Диспетчерское управление и сбор данных (англ. Supervisory Control And Data Acquisition). Программное обеспечение, выполняемое на ПК с целью получения и отображения данных в удобном для пользователя виде, с возможностью управления

1.3 Нормативно-справочная документация

Перечень нормативно-справочной и эксплуатационной документации, использованной в данном документе, приведен в приложении Н.

1.4 Аппаратно программные требования к оборудованию и ПО используемым при программировании ПЛК

1.4.1 Требования к оборудованию

Программирование ПЛК производится с использованием персональных компьютеров (ПК) с характеристиками, определяемыми тем, что программирование выполняется с использованием ПО CODESYS (**Co**ntroller **De**velopment **Sys**tem) производства компании «3S – Smart Software Solutions GmbH» и устанавливаемыми производителем ПО:

- Pentium IV, 2 ГГц,
- 512 Мб ОЗУ (рекомендуется 1024),
- 500 Мб жесткий диск,
- CD ROM привод,
- интерфейсы RS-232, Ethernet или USB; используются для подключения ПЛК.

Выполнение соединения ПЛК с ПК для программирования ПЛК производится с помощью кабеля КС14, входящего в комплект поставки ПЛК. См. также раздел 3.3.5, в котором описано, как можно запрограммировать ПЛК110, соединившись с ним по другим интерфейсам.

Подключение описано в документах «ПЛК110. Руководство по эксплуатации».

1.4.2 Требования к ПО

Программирование ПЛК производится с использованием следующего программного обеспечения (ПО):

- Windows: операционная система, установленная на персональном компьютере (ПК) и необходимая для инсталляции, запуска и выполнения ПО CODESYS.
- CODESYS (**Co**ntroller **De**velopment **Sys**tem): программное обеспечение (среда программирования) производства компании «3S – Smart Software Solutions GmbH», работающее на персональном компьютере (ПК) и применяемое при подготовке пользовательских программ ПЛК. Рекомендуемая версия ПО – 2.3.9.9. Бесплатные обновления версий ПО CoDeSys доступны на сайтах www.codesys.ru, www.3s-software.com и www.owen.ru.
- Комплект файлов библиотек дополнительных программных модулей (см. раздел 4.2.4)

– Файл настроек целевой платформы (target-файл), соответствующий используемому контроллеру.

1.4.3 Требования к персоналу

Персонал, выполняющий программирование ПЛК, должен:

– Владеть приемами работы с графическим интерфейсом операционной системы и программного обеспечения.
– Владеть методикой программирования ПЛК с использованием ПО CODESYS в объеме, изложенном в документе «Руководство пользователя по программированию ПЛК в CODESYS 2.3».
– Владеть методикой эксплуатации ПЛК в объеме, изложенном в документе «Руководство по эксплуатации ПЛК110».
– Владеть методикой программирования ПЛК в объеме, изложенном в настоящем документе.

2 Предварительный этап программирования ПЛК

В данном разделе описывается предварительный этап процедуры программирования ПЛК.

2.1 Предварительный этап

В данном разделе описываются предварительные этапы работы: установка операционной системы и установка ПО CODESYS.

2.1.1 Установка операционной системы

Для инсталляции, запуска и выполнения ПО CODESYS на ПК необходима установка операционной системы Windows. Установка операционной системы производится в соответствии с инструкциями, размещенными на дистрибутивном диске ОС. В данном документе установка операционной системы не рассматривается.

2.1.2 Установка ПО (среды программирования) CODESYS

Установка ПО (среды программирования) CODESYS (**Co**ntroller **De**velopment **Sys**tem) производится запуском программы-инсталлятора (файл Codesys_v2399.exe на дистрибутивном диске ПЛК) и выполнением инструкций, отображаемых в окнах программы.

Примечания

1) ПО CODESYS бесплатно и не требует лицензирования (за исключением отдельных необязательных приложений).

2) При установке ПО CODESYS следует обратить внимание на то, что выбор языка работы программы в процессе ее установки осуществляется дважды; при первом выборе русский язык отсутствует в списке доступных языков, при втором – присутствует.

2.2 Выбор контроллера: размер памяти

Связь ПЛК с внешними устройствами (модулями ввода-вывода и т.д.) производится по сети через специальную область памяти ПЛК: «Память ввода-вывода».

Размер памяти ввода-вывода определяется типом лицензии CODESYS контроллера ОВЕН ПЛК. Тип лицензии указывается в маркировке конкретного ПЛК, в последнем знаке обозначения: «L» или «M».

- Для контроллеров, последняя буква обозначениния которых – латинская буква «L», устанавливается ограничение объема памяти ввода-вывода размером 360 байт, при этом 122 байта отводятся для памяти ввода (%I), 234 байта отводятся для памяти вывода (%Q) и оставшиеся 4 байта – под специальную память (%M).
  Например, контроллер программируемый логический **ПЛК110-24.60.Р-L** – имеет ограничение объема памяти ввода-вывода размером 360 байт
- Для контроллеров, последняя буква обозначения которых – «M», ограничений в размере памяти не вводится. По умолчанию суммарный объем памяти ввода (%I) и вывода (%Q) установлен равным 16кБ. Этого достаточно для большинства задач, но этот объем может быть увеличен пользователем до 32 кБ (на вкладке «Memory Layout» окна «Target Setting», в строках Input и Output, см. рисунок 2.4).
  Например, контроллер программируемый логический **ПЛК110-24.60.Р-M** – не имеет ограничения объема памяти ввода-вывода.

Примечание. Ограничение до 360 байт распространяется только на размер памяти области ввода-вывода, количество внутренних переменных программы контроллера ограничивается только количеством свободной оперативной памяти.

Внимание! Выбор контроллера с требуемой лицензией следует делать до покупки ПЛК.

Для расчета необходимого объема памяти ввода / вывода и выбора требуемого типа лицензии можно воспользоваться методикой, изложенной в разделе 3.4.1 .

Задание конфигурации памяти ввода / вывода описано в разделе 7 настоящего документа.

3 Этапы создания пользовательской программы («проекта») ПЛК

В данном разделе содержится краткое описание последовательности операций, выполняемых при создании пользовательской программы («проекта»), описание интерфейса, основных режимов работы ПО CODESYS и приемов работы в этих режимах.

Полностью процедуры создания и отладки пользовательской программы («проекта») ПЛК описаны в документе «Руководство пользователя по программированию ПЛК в CODESYS 2.3».

Описания тех аспектов процедуры, которые специфичны для работы с ПЛК, например описания настройки (конфигурирования) области ввода / вывода памяти ПЛК, приведены далее.

Внимание! Терминология описания, принятая в документации разработчика ПО CODESYS, и, вслед за ней, в данном документе, несколько отличается от принятой в ЕСПД. Так, например, различные режимы работы ПО именуются различно, в зависимости от способа вызова режима. Например, режимы написания программ именуются «Редакторы».

3.1 Компоненты проекта

Пользовательская программа («проект») ПЛК в ПО CODESYS содержит программные компоненты (POU), типы данных, визуализации, ресурсы и библиотеки, сведения о ресурсах ПЛК и некоторую другую информацию, хранимую в одном файле («name.pro»).

3.1.1 Программные компоненты проекта (POU)

Проект создается в ПО CODESYS на любом из доступных языков программирования. Проект может состоять из одного или нескольких программных компонентов (POU, Program Organization Unit). Главная программа, выполняемая циклически, должна называться PLC_PRG.

К программным компонентам (POU) относятся функциональные блоки, функции и программы. Отдельные POU могут включать действия (подпрограммы).

Каждый программный компонент состоит из раздела объявлений и кода. Для написания всего кода POU используется только один из МЭК языков программирования (IL, ST, FBD, SFC, LD или CFC).

CODESYS поддерживает все описанные стандартом МЭК 61131 компоненты. Для их использования достаточно включить в свой проект библиотеку дополнительных программных компонентов «standard.lib» (подробнее о библиотеках см. разделы ниже).

POU могут вызывать другие POU, но рекурсии недопустимы.

Кроме того, в проекте могут быть явно определены несколько задач с различными условиями выполнения. Работа с задачами описана в разделе 6.7 «Конфигуратор задач (Task Configuration)» документа «Руководство пользователя по программированию ПЛК в CODESYS 2.3».

Приемы работы при написании программ и примеры программ представлены в документе «Первые шаги в CODESYS».

Выполнение программы начинается с программного компонента POU «PLC_PRG» и выполняется циклически.

Создание и отладка проекта производится в несколько этапов, перечисленных ниже.

3.2 Установка настроек целевой платформы (target-файла)

В данном разделе описывается этап установки целевой платформы (т.е. программируемого контроллера).

Спомощью ПО CODESYS можно разрабатывать программы для большого количества моделей ПЛК. На этапе установления связи с контроллером программа должна взаимодействовать с конфигурацией конкретного контроллера, содержащей определенным образом настроенные параметры его входов, выходов, интерфейсов связи, и некоторые другие характеристики. Исходная информация о конфигурации ПЛК содержится в предварительных настройках целевой платформы (target-файле) контроллера, поставляемых производителем контроллера и размещенных в папке «Target-файлы» дистрибутивного диска. Настройки целевой платформы поставляются в виде набора файлов, основным (указываемым пользователем в процессе установки настроек) среди которых является Target-файл, имеющий расширение *.tnf, (**T**arget I**nf**ormation).

Файл содержит информацию о ресурсах конкретного ПЛК (о количестве и типах входов и выходов, интерфейсов, памяти, дополнительных устройств и т.д.), с которыми работает ПО CODESYS.

Для того, чтобы определенная модель ПЛК стала доступной для разработки программ в конкретной системе с установленным ПО CODESYS, требуется в этой системе установить Target-файл целевой платформы (контроллера).

Внимание! Компания ОВЕН совершенствует производимые контроллеры и программное обеспечение и периодически предлагает пользователю обновленные версии встроенного ПО микроконтроллера и Target-файлов. Подробнее об обновлении встроенного ПО микроконтроллера и Target-файлов см раздел 10

Target-файлы для разных версий встроенного ПО одной и той же модели контроллера могут быть установлены в один и тот же экземпляр CODESYS; их названия всегда различаются, например, указанными в них номерами версий. Проект, созданный с использованием target-файла для встроенного ПО одной версии может оказаться несовместимым со встроенным ПО другой версии. Для переноса проекта между контроллерами с различными версиями встроенного ПО необходимо выполнить действия, описанные в Приложении И.

Установка настроек целевой платформы (target-файла) может производиться двумя способами, описанными ниже.

Внимание! При выборе Target-файла в процессе установки настроек следует обратить внимание на то, что имя Target-файла не полностью совпадает с наименованием контроллера: в наименовании контроллера использована кириллица (например, ПЛК110), а в названии Target-файла – латиница (например, PLC110). Для каждой модификации ПЛК в поставку включен соответствующий Target-файл. Так, для ПЛК110 на дистрибутивном диске размещены Target-файлы PLC110.30-L_v2, PLC110.30-M_v2, PLC110.32-L_v2, PLC110.32-M_v2, PLC110.60-L_v2, PLC110.60-M_v2.

3.2.1 Способ 1

Установка настроек целевой платформы(target-файла) производится при помощи утилиты «InstallTarget». Утилита представляет собой компонент ПО CODESYS и устанавливается на ПК совместно с ПО.

Порядок инсталляции Target-файлов таков:

1) Выбрать команду **Пуск | Программы | 3S Software | CODESYS V2.3 | InstallTarget**.
2) В открывшемся окне утилиты «InstallTarget» окне (рисунок 3.1, а) – нажать кнопку «Open (Открыть)».
3) В открывшемся окне выбора файла – указать путь к инсталлируемому Target-файлу.
   В перечне доступных файлов следует выбрать требуемый: соответствующий модификации программируемого контроллера.
   В поле «Installation directory» отобразится выбранный путь к папке.
   В поле «Possible Taigets» отобразится список доступных Target-файлов.
   В поле «InstalledTargets» отобразится перечень ранее установленных файлов (см. рисунок 3.1, б).
4) После того, как требуемый Taiget-файл выделен, следует нажать кнопку «Install». Taiget-файл будет инсталлирован на используемый ПК. В поле «InstalledTargets» он отобразится в перечне установленных файлов (см. рисунок 3.1, в).
5) При необходимости (например, при ошибке в выборе файла), установленный Taiget-файл может быть деинсталлирован. Для этого его следует выделить в списке в поле «InstalledTargets» и нажать кнопку «Remove». Файл будет деинсталлирован и перестанет отображаться в списке в поле «InstalledTargets».
6) После завершения инсталляции требуемого Taiget-файла – нажать кнопку «Close». Окно утилиты «InstallTarget» закроется.

На этом процедура инсталляции Target-файла завершается.

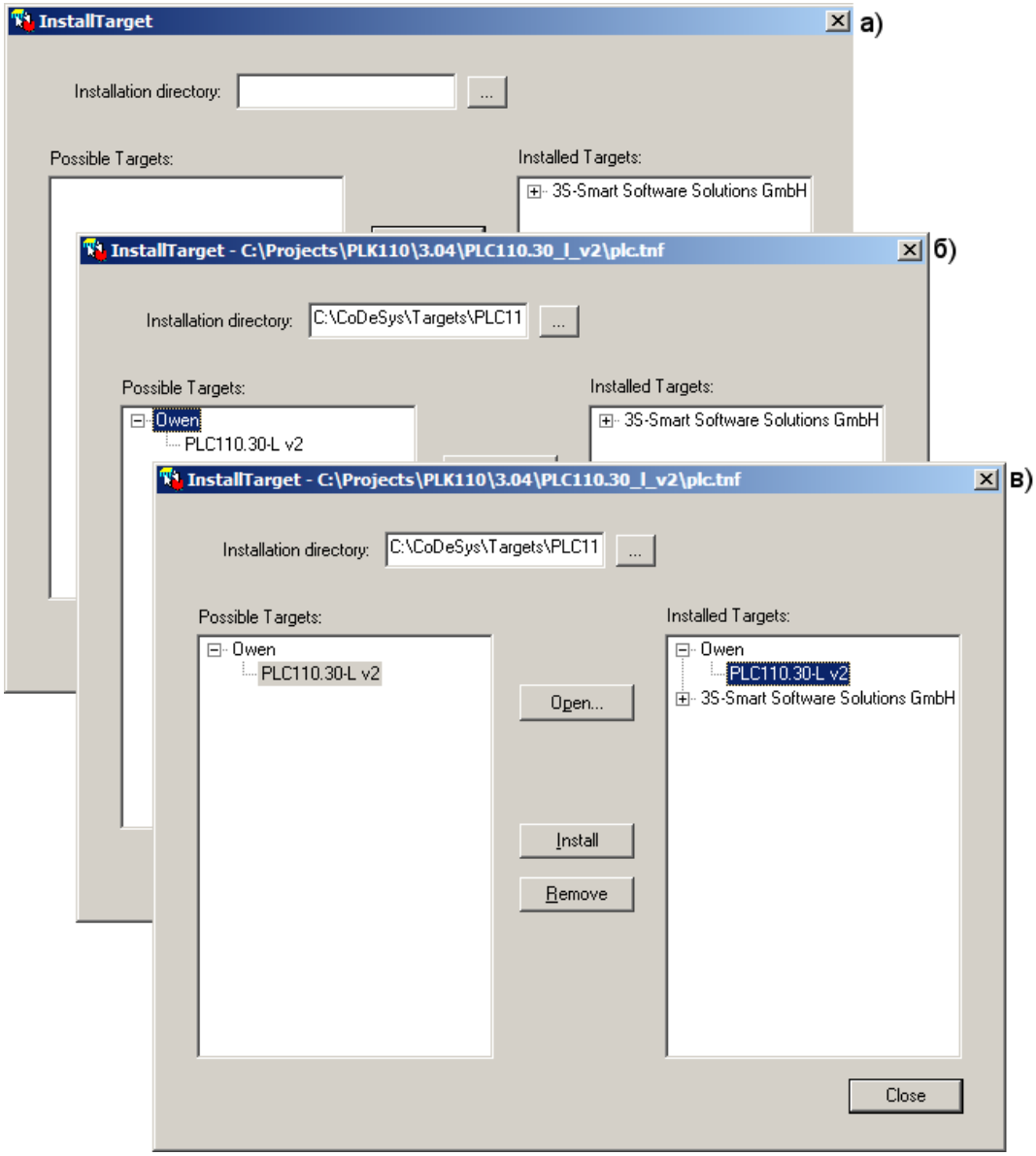


Рисунок 3.1 – Окно «InstallTarget» утилитыInstallTarget

3.2.2 Способ 2

Установка настроек целевой платформы(target-файла) производится при помощи специализированной утилиты «InstallTarget» от производителя ПЛК. Утилита представляет собой файл InstallTarget.bat, который размещенн на дистрибутивном диске.

Порядок инсталляции Target-файлов таков:

1) Нажатием требуемой кнопки навигатора дистрибутивного диска запустить на выполнение файл InstallTarget
2) Дождатся окончания выполнения установки.
3) После завершения установки среда программирования – CodeSys v2.3 требует перезапуска.

На этом инсталляция Target-файла завершается. Никаких дополнительных операций выполнять не требуется.

После инсталляции Target-файлов следует перейти к собственно программированию ПЛК.

3.3 Запуск ПО CODESYS. Главное окно программы

Запуск ПО осуществляется любым из способов, доступных в ОС Windows. Например, вызовом команды **Пуск | Программы | 3S Software | Codesys 2.3 | Codesys 2.3**.

После запуска ПО CODESYS открывается Главное окно программы (см. рисунок 3.3).

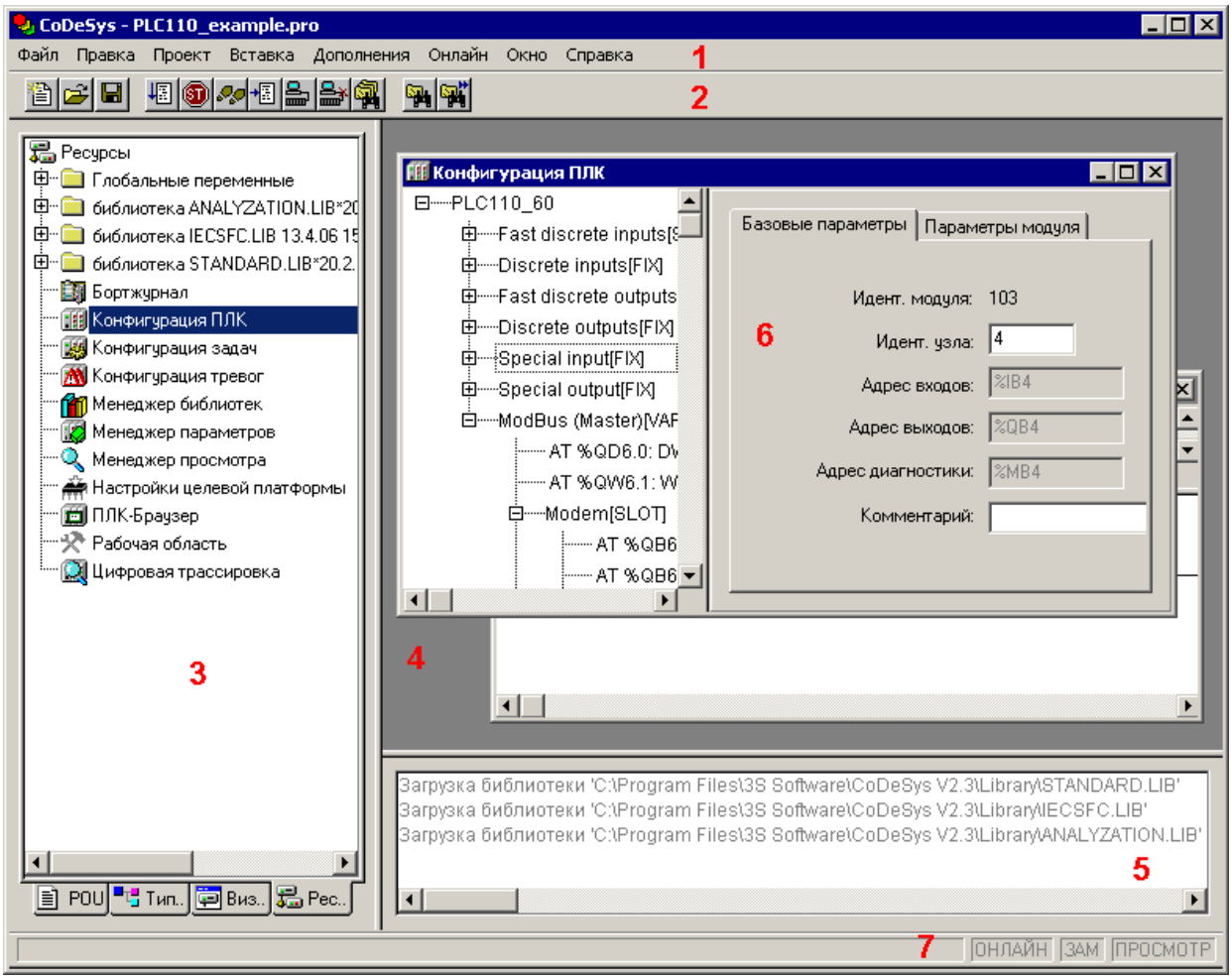


Рисунок 3.2 – Главное окно ПО CODESYS

Главное окно программы содержит следующие элементы (помечены красными цифрами на рисунке 3.2):

- Главное меню (1), содержащее перечень доступных групп команд программы. В различных режимах работы группы команд главного меню дополняются специализированными командами.
- Панель инструментов (2), содержащая кнопки, дублирующие часто используемые команды программы. В различных режимах работы панель инструментов дополняется специализированными панелями.
- «Организатор объектов» (3) – переключатель групп режимов работы программы; включает четыре вкладки: «POU», «Типы данных (Datatypes)», «Визуализации (Visualizations)» и «Ресурсы (Resources)».
- Рабочая область программы (4), в которой отображаются окна (6) режимов работы программы (в правой верхней части).

- Окно (область) сообщений (5, в правой нижней части). В этом окне появляются сообщения компилятора, результаты поиска и список перекрестных ссылок.
- Строка статуса (7), содержащая информацию о текущем состоянии проекта (в нижней части). При выборе пункта меню в строке статуса отображается его описание; при работе в текстовом редакторе – указывается позиция, в которой находится курсор (например, **Line:5, Col.11**); в режиме визуализации – отображаются координаты курсора X и Y, которые отсчитываются относительно верхнего левого угла окна; если указатель мыши находится на элементе, или над элементом производятся какие-либо действия, то отображается номер этого элемента; при вставке элемента – отображается его название (например, Rectangle).
  При работе в режиме «Online» надпись **Online** в строке статуса выделяется черным цветом, в ином случае надпись серая.
  В режиме «Online» можно определить, в каком состоянии находится программа: **SIM** – в режиме эмуляции, **RUN** – программа запущена, **BP** – установлена точка останова, **FORCE** – происходит фиксация переменных.

Области Главного окна разделены линиями – разделителями, которые могут перемещаться с помощью мыши. Это позволяет подобрать оптимальное сочетание размеров областей.

Управление работой программы осуществляется выбором требуемых команд главного меню, которое отображается в верхней части главного окна программы и содержит пункты «Файл (File)», «Правка (Edit)», «Проект (Project)», «Вставка (Insert)», «Дополнения (Extras)», «Онлайн (Online)», «Окно (Window)» и «Справка (Help)».

Вызов команд главного меню дублируется кнопками панели инструментов, командами контекстных меню и горячими клавишами.

Перечень элементов управления программой с описанием запускаемых ими операций приведен в таблице А.1.

3.3.1.1 «Организатор объектов»

Организатор объектов ПО CODESYS расположен в левой части главного окна программы (см. рисунок 2.2) и предназначен для вызова режимов работы ПО.

Организатор объектов включает четыре вкладки: «POU», «Типы данных (Datatypes)», «Визуализации (Visualizations)» и «Ресурсы (Resources)».

В пределах вкладок отображаются иерархические списки соответствующих объектов программы (например, режимов работы ПО или объектов в пределах одного и того же режима).

Для перехода на требуемую вкладку следует щелкнуть левой кнопкой мыши на наименовании требуемой вкладки (в нижней части «Организатора объектов»).

Для перехода к требуемому объекту в пределах выбранной вкладки следует щелкнуть левой кнопкой мыши на наименовании требуемого объекта или перейти на требуемую строку с помощью клавиш со стрелками.

Для открытия окна режима (или окна одного из объектов режима) следует дважды щелкнуть левой кнопкой мыши на наименовании требуемого объекта или, выбрав наименование требуемого объекта, нажать клавишу <Enter>. Окно режима (или окно одного из объектов режима) откроется в рабочей области (в правой верхней части) главного окна.

На вкладке «**POU**» отображается иерархический список программных компонентов (POU) проекта: функциональных блоков, функций и программ. Отдельные POU могут включать действия (подпрограммы). Каждый программный компонент состоит из раздела объявлений и кода. Для написания всего кода POU используется только один из МЭК языков программирования (IL, ST, FBD, SFC, LD или CFC).

CODESYS поддерживает все описанные стандартом МЭК компоненты. Для их использования достаточно включить в свой проект библиотеку standard.lib.

На вкладке «**Типы данных (Data types)**» отображается иерархический список типов данных, используемых в проекте. Кроме стандартных типов данных, в проекте могут быть использованы определяемые пользователем типы данных: структуры, перечисления и ссылки.

На вкладке «**Визуализации (Visualizations)**» отображается иерархический список элементов визуализации проекта – **графических представлений** объекта управления. Визуализация непосредственно связана с созданной в CODESYS программой контроллера. **Редактор визуализации** CODESYS предоставляет набор готовых графических элементов, которые могут быть связаны соответствующим образом с переменными проекта.

В Online режиме представление элементов на экране изменяется в зависимости от значений переменных.

Визуализация может исполняться в системе программирования, в отдельном приложении CODESYS HMI или как Web или целевая (в ПЛК) визуализация.

На вкладке «**Ресурсы (Resources)**» отображается иерархический список ресурсов – объектов CODESYS, обеспечивающих конфигурацию проекта, включая:

- Глобальные переменные, используемые во всем проекте.
- Менеджер библиотек для подключения необходимых библиотек к проекту.
- Журнал записи действий во время исполнения.
- Конфигурация тревог для конфигурирования обработки тревог в про-екте.
- Конфигурация ПЛК для конфигурирования аппаратуры контроллера..
- Конфигурация задач для управления задачами.
- Менеджер просмотра для просмотра и заказа наборов значений переменных.
- Настройки целевой платформы.

Двойной щелчок левой кнопкой мыши на требуемой записи в списке «Ресурсы» приводит к открытию в рабочей области главного окна окна выбранного режима («ресурса»).

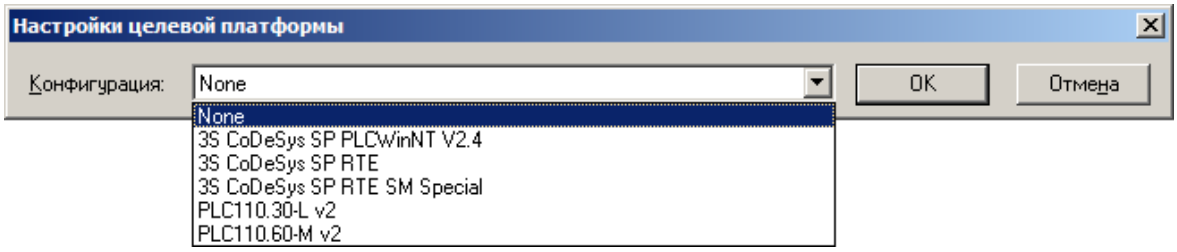
3.3.2 Проект. Выбор контроллера и языка программирования

Для созданияновогопроекта(пользовательской программы ПЛК) следует:

1) Выбрав команду **Пуск | Программы | 3S Software | CODESYS V.2.3 | CODESYS V.2.3**, запустить ПО CODESYS.
2) В открывшемся главном окне ПО CODESYS (рисунок 3.2) – вызвать команду «Файл | Новый (File | New)**»** главного меню или нажать кнопку «Новый (New)» () панели инструментов.
3) В открывшемся окне «Настройки целевой платформы (Target Settings)» (рисунок 3.3, а) – нажатием на кнопку у правого края поля «Конфигурация (Configuration)» раскрыть список предварительно установленных на ПК Target-файлов (см. п. 3.2). В списке – выделить требуемый файл (рисунок 3.3, б) и щелкнуть на его названии левой кнопкой мыши.

а)

б)



в)

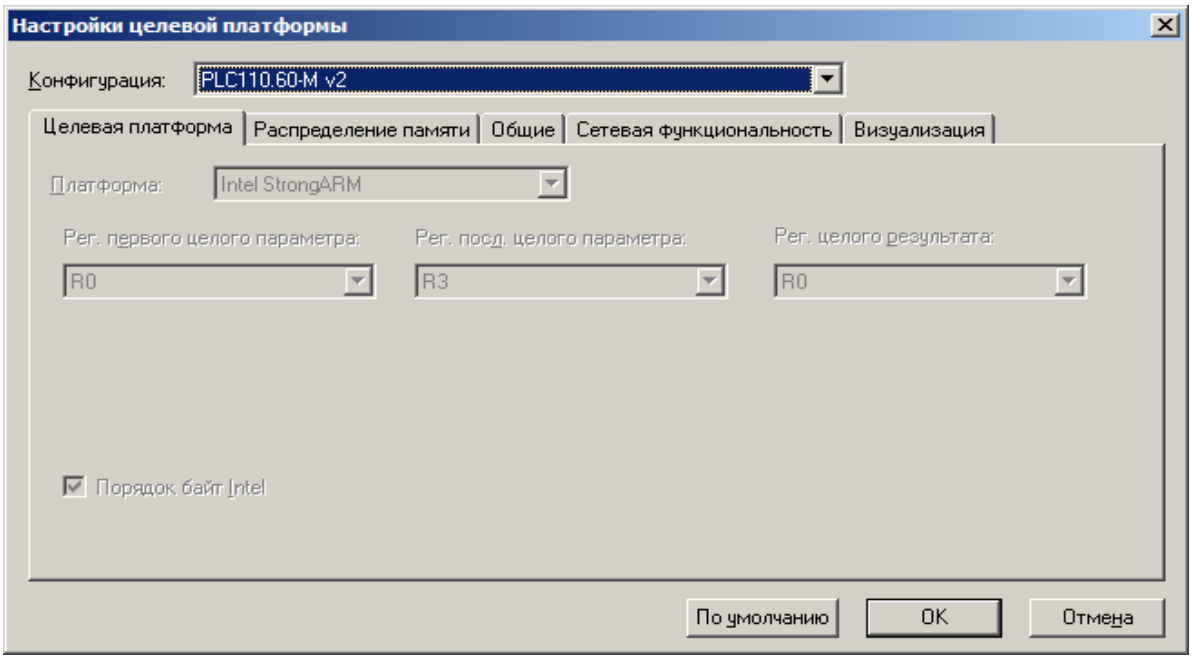


Рисунок 3.3 – Окно «Настройки целевой платформы (TargetSetting)»

4) В открывшихся вкладках окна «Настройки целевой платформы (Target Setting)» отображаются установленные производителем значения параметров целевой платформы (рисунок 3.4, в).
Как правило, установленные производителем значения параметров не требуют изменения.
Исключение могут составить размеры сохраняемой при отключении питания Retain-памяти и памяти входов/выходов.
Объем retain-памяти по умолчанию установлен «16#4000»[1], что соответствует 16 кбайт.

Для того, чтобы увеличить размеры памяти входов/выходов, следует перейти на вкладку «Распределение памяти (Memory Layout)» окна «Настройки целевой платформы (Target Setting)» (см. рисунок 3.5), и исправить значение в строках «Входы»/«Выходы»: значение «16#1FFF» заменить на «16#2FFF»

[1]В предыдущей версии ПО контроллера для того, чтобы увеличить размер Retain-памяти, следовало перейти на вкладку «Распределение памяти (Memory Layout)» окна «Настройки целевой платформы (Target Setting)» (см. рисунок 3.4), и исправить значение в строке Retain («Энергонез.»): значение «16#1000» заменить на «16#4000», в настоящей же версии это значение по умолчанию установлено «16#4000».

5) Нажать кнопку «ОК» окна «Настройки целевой платформы (Target Setting)».

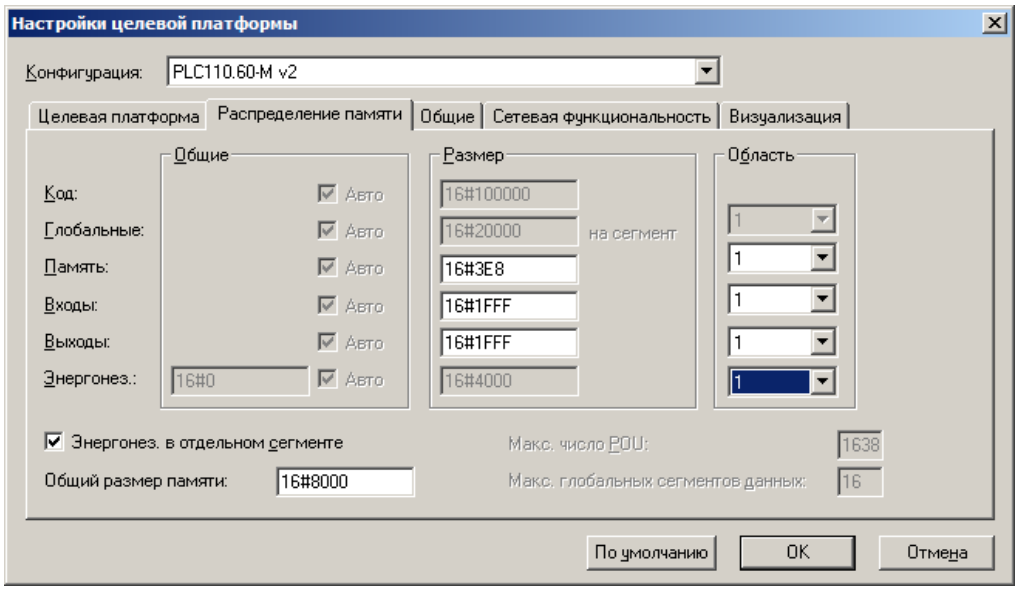


Рисунок 3.4 – Окно « Настройки целевой платформы (Target Setting)», вкладка «Распределение памяти (MemoryLayout)»

6) В открывшемся окне «Новый программный компонент (NewPOU)» (см. рисунок 3.5), в поле «Имя нового POU (NameofnewPOU)» – отображается заданное по умолчанию имя новой главной программы проекта (**PLC_PRG**); его не следует изменять. В группе переключателей «Тип POU (TypeofPOU)» отображается заданный по умолчанию тип новой главной программы проекта (**Программа (Program)**); его также не следует изменять.

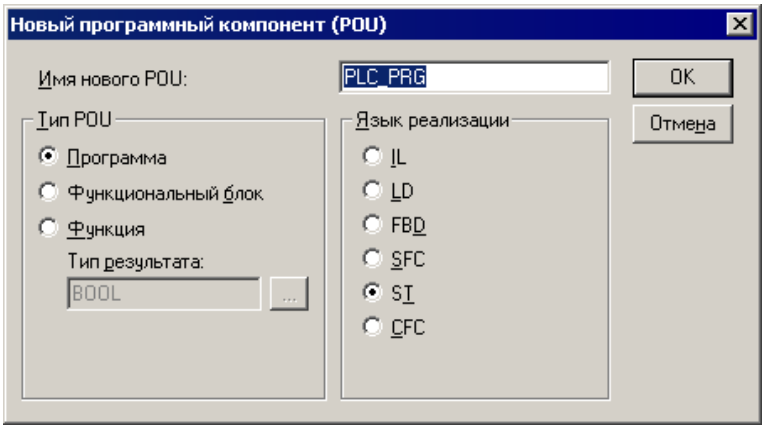
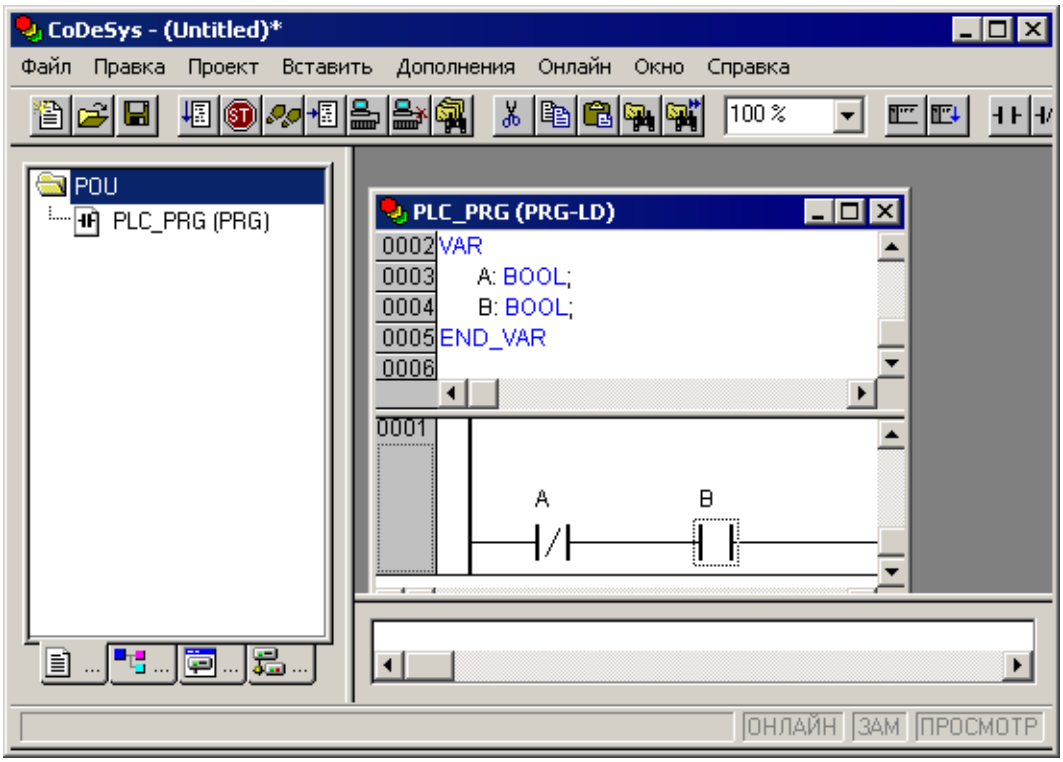


Рисунок 3.5 – Окно «Новый программный компонент (NewPOU)»

7) В группе переключателей «Язык реализации (LanguageofthePOU)» следует выбрать требуемый язык программирования (о языках программирования – см. раздел 3.3.2.1).
В правой верхней области главного окна программы откроется окно редактора, в котором создается программа, исполняемая контроллером. В зависимости от выбранного языка программирования это окно выглядит по-разному (на рисунке 3.6, а – пример для языка LD (Ladder

Diagram – Язык релейных диаграмм).
В верхней части этого окна отображается область объявления переменных – «Редактор объявлений», в нижней – область редактора собственно программы.
Одновременно главное меню программы (команда «Вставить (Insert)») и контекстное меню области редактирования программы (см. рисунок 3.6, б) дополняются командами, специфичными для выбранного языка. Кроме того, панель инструментов дополняется локальной панелью, содержащей кнопки, соответствующие этим командам.

а)



б)

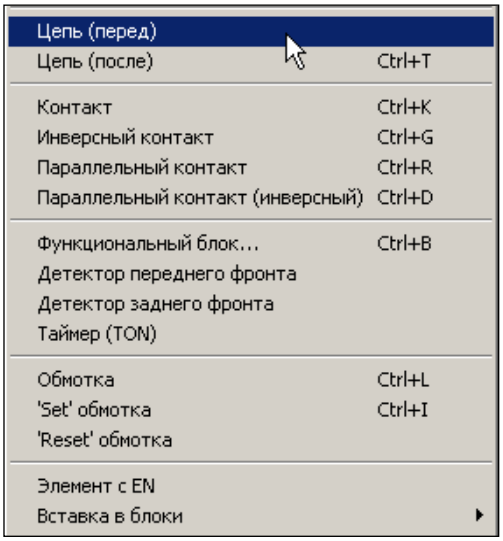


Рисунок 3.6 – Окно создания проекта (а) PLC_PRG (PRG-LD) и контекстное меню области редактирования программы (б) на языке LD (Ladder Diagram – Язык релейных диаграмм)

3.3.2.1 Языки программирования

В соответствии с требованиями стандарта МЭК 61131, ПО CODESYS поддерживает следующие языки программирования.

Кроме того, ПО CODESYS поддерживает «Язык непрерывных функциональных схем» (**CFC**), схожий с FBD, но, в отличие от последнего, блоки и соединители в этом языке располагаются свободно, разрешаются циклы и свободные соединения

Каждый из перечисленных языков обладает специфическими чертами, определяющими их применение для решения определенных задач.

Подробное описание языков программирования приведено в документе документа «Руководство пользователя по программированию ПЛК в CODESYS 2.3».

Краткие описания языков программирования приведены ниже.

3.3.2.1.1 Язык «IL» – список инструкций

Текстовый язык, схожий с ассемблером STEP5 фирмы SIEMENS; все операции производятся через аккумулятор; легко читается в случае небольших программ.

Каждая инструкция начинается с новой строки и содержит оператор и, в зависимости от типа операции, один и более операндов, разделенных запятыми.

Перед операндом может находиться метка, заканчивающаяся двоеточием (:). Комментарий должен быть последним элементом в строке. Между инструкциями могут находиться пустые строки. Пример IL программы приведен на рисунке 3.7

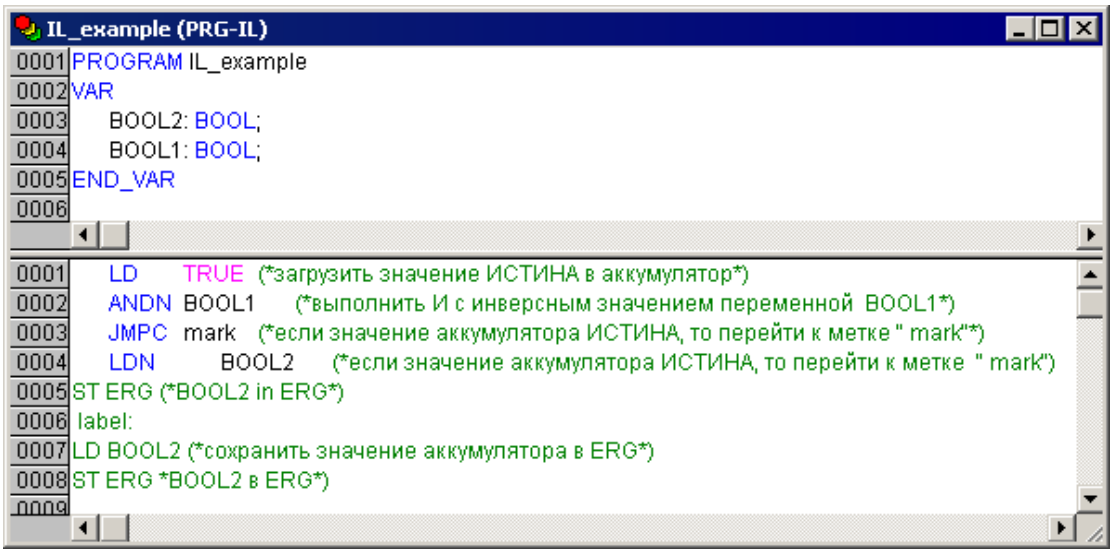


Рисунок 3.7 – Пример программы на языке IL

3.3.2.1.2 Язык «ST» – структурированный текст

Текстовый язык высокого уровня, схожий с языком «Паскаль»; оптимален для программирования циклов и условий. Представляет собой набор инструкций, которые могут использоваться в условных операторах (IF...THEN...ELSE) и в циклах (WHILE...DO).

Пример ST программы приведен на рисунке 3.8.

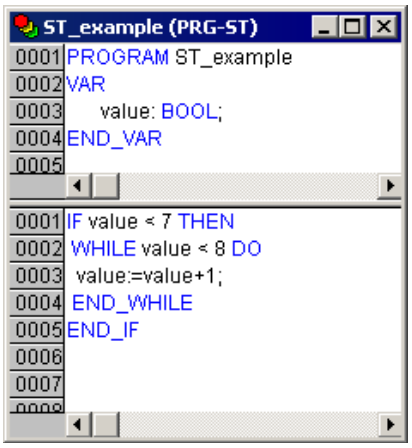


Рисунок 3.8 – Пример программы на языке ST

3.3.2.1.3 Язык «FBD»– функциональные блоковые диаграммы

Графический язык программирования. Работает со схемами, состоящими из блоков и операндов – с последовательностью цепей, каждая из которых содержит логическое или арифметическое выражение, вызов функционального блока, переход или инструкцию возврата.

Пример FBD программы приведен на рисунке 3.9.

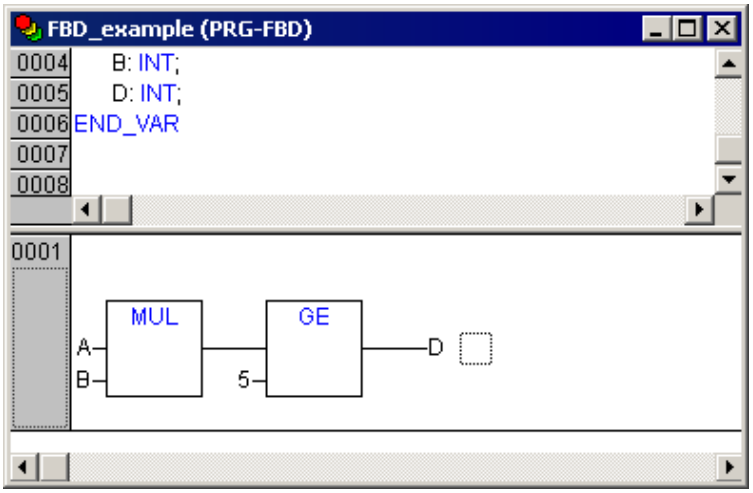


Рисунок 3.9 – Пример программы на языке FBD

3.3.2.1.4 Язык «LD» – релейные диаграммы

Графический язык, реализующий структуры электрических цепей; программа на языке LD состоит из схем с последовательностью цепей, каждая из которых содержит логическое или арифметическое выражение, вызов функционального блока, переход или инструкцию возврата. Сложен в использовании для работы с аналоговыми типами данных.

Лучше всего LD подходит для построения логических переключателей, но достаточно легко можно создавать на нем и сложные цепи – как в FBD. Кроме того, LD достаточно удобен для управления другими компонентами POU.

Используется для программирования большинства ПЛК. Допустимо переключение между языками FBD и LD.

Диаграмма LD состоит из ряда цепей. Слева и справа схема ограничена вертикальными линиями – шинами питания. Между ними расположены цепи, образо-

ванные контактами и обмотками реле, по аналогии с обычными электронными цепями. Слева любая цепь начинается набором контактов, которые посылают слева направо состояние «ON» или «OFF», соотвеетствующие логическим значениям ИСТИНА или ЛОЖЬ. Каждому контакту соответствует логическая переменная. Если переменная имеет значение «ИСТИНА», то состояние передается через контакт, если «ЛОЖЬ», то правое соединение получает значение «Выключено (OFF)».

Пример программы на языке LD приведен на рисунке 3.10.

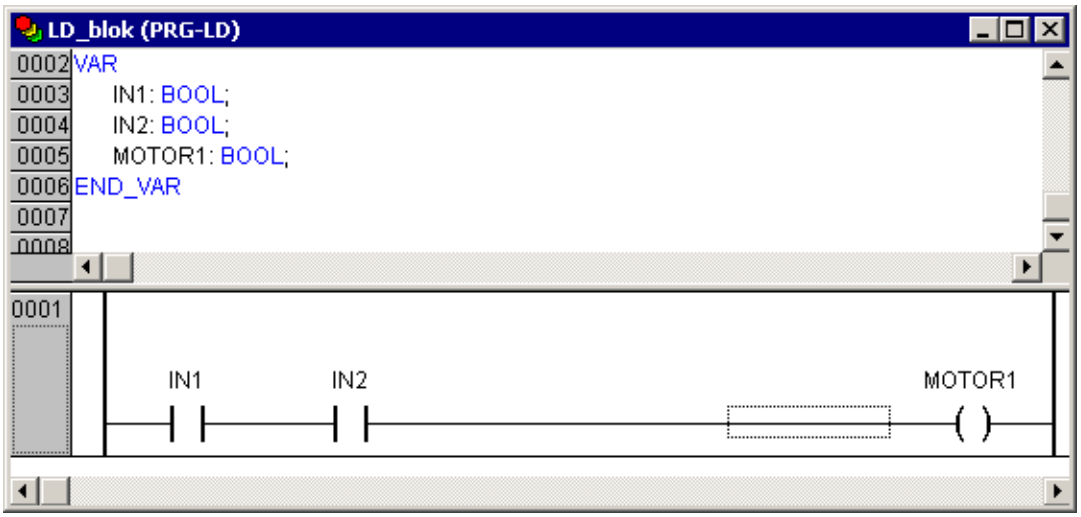


Рисунок 3.10 – Пример программы на языке LD

3.3.2.1.5 Язык «SFC» – последовательные функциональные схемы

Графический язык, используемый для структурирования приложений; состоит из шагов и переходов; действия выполняются внутри шагов. Не конвертируется в другие языки.

Пример программы на языке SFC приведен на рисунке 3.11.

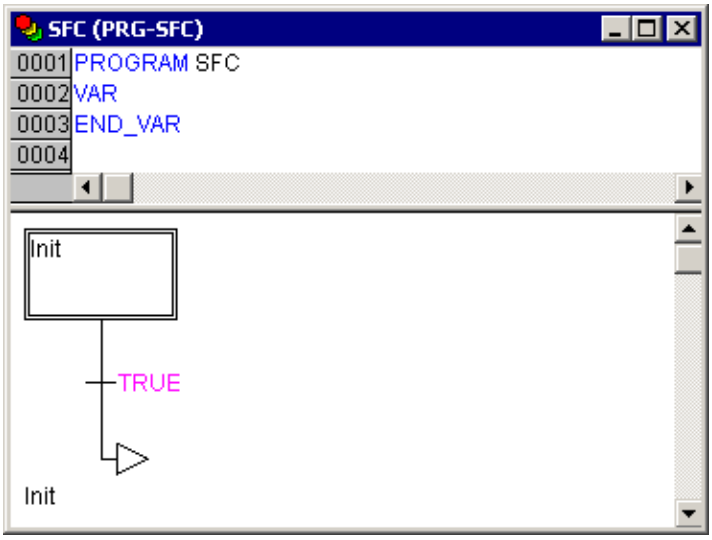


Рисунок 3.11 – Пример программы на языке SFC

3.3.2.1.6 Язык «CFC» – непрерывные функциональные схемы

Язык непрерывных функциональных схем. В отличие от FBD, не использует цепи, но дает возможность свободно размещать компоненты и соединения, что позволяет создавать, в частности, обратные связи.

Пример CFC программы приведен на рисунке 3.12.

Примечание. *Свобода размещения компонентов и соединений определяет необходимость упорядочения порядка выполнения программы. Группа команд «Порядок | Показать порядок / Упорядочить топологически / В соответствии с потоком данных / Порядок: Выше, Ниже, В начало, В конец» контекстного меню позволяет отобразить порядковые номера (по очередности выполнения) элементов программы и изменить этот порядок при необходимости. Порядковые номера элементов отображаются в затемнённом квадратике у правого верхнего угла каждого элемента (см. рисунок 3.12).*

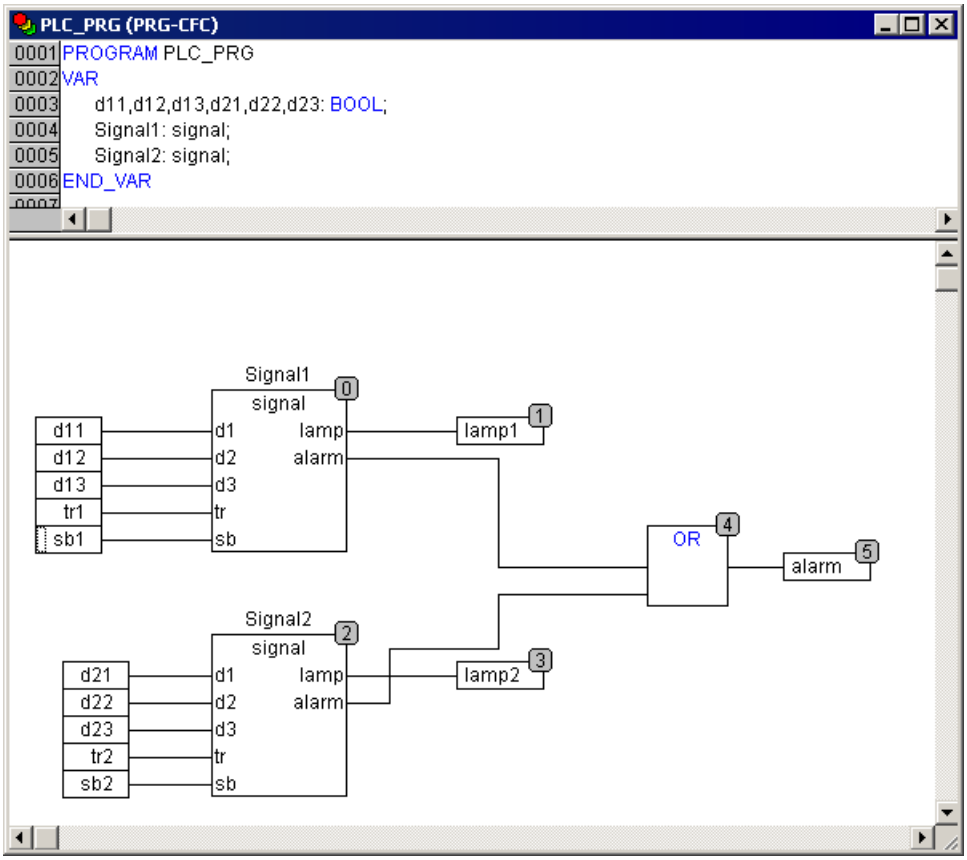


Рисунок 3.12 – Пример программы на языке CFC

3.3.2.2 Редакторы ПО CODESYS

Все режимы редактирования (далее – «редакторы») программных компонентов POU (Program Organization Units) содержат область кода (нижняя часть окна) и раздел объявлений (верхняя часть окна), см. рисунки 3.6 – 3.12.

Область кода может включать графический или текстовый редактор; раздел объявлений – это всегда текст. Разделы кода и объявлений разделены горизонтальной границей, которую можно перетаскивать мышкой.

Окно редактирования открывается при входе в режим написания программного компонента. Для входа следует перейти на вкладку «POU» организатора объектов и выбрать в дереве программных компонентов проекта, отображаемом на вкладке,

требуемый элемент. В рабочей области главного окна ПО CODESYS откроется окно редактора. Тип окна зависит от выбранного языка программирования (см. п. 3.3.2.1). В каждом окне редактора становятся доступны команды контекстного меню, содержащие основные операции, доступные в выбранном языке. Одновременно панель инструментов главного окна ПО CODESYS дополняется панелью, кнопки которой аналогично командам контексного меню вызывают основные операции, доступные в выбранном языке. Окно редактора открывается также при добавлении программного компонента.

Пример главного окна ПО CODESYS с открытым окном редактирования программного компонента на языке FBD (с дополнительной панелью инструментов и открытым контекстным меню) приведен на рисунке 3.13.

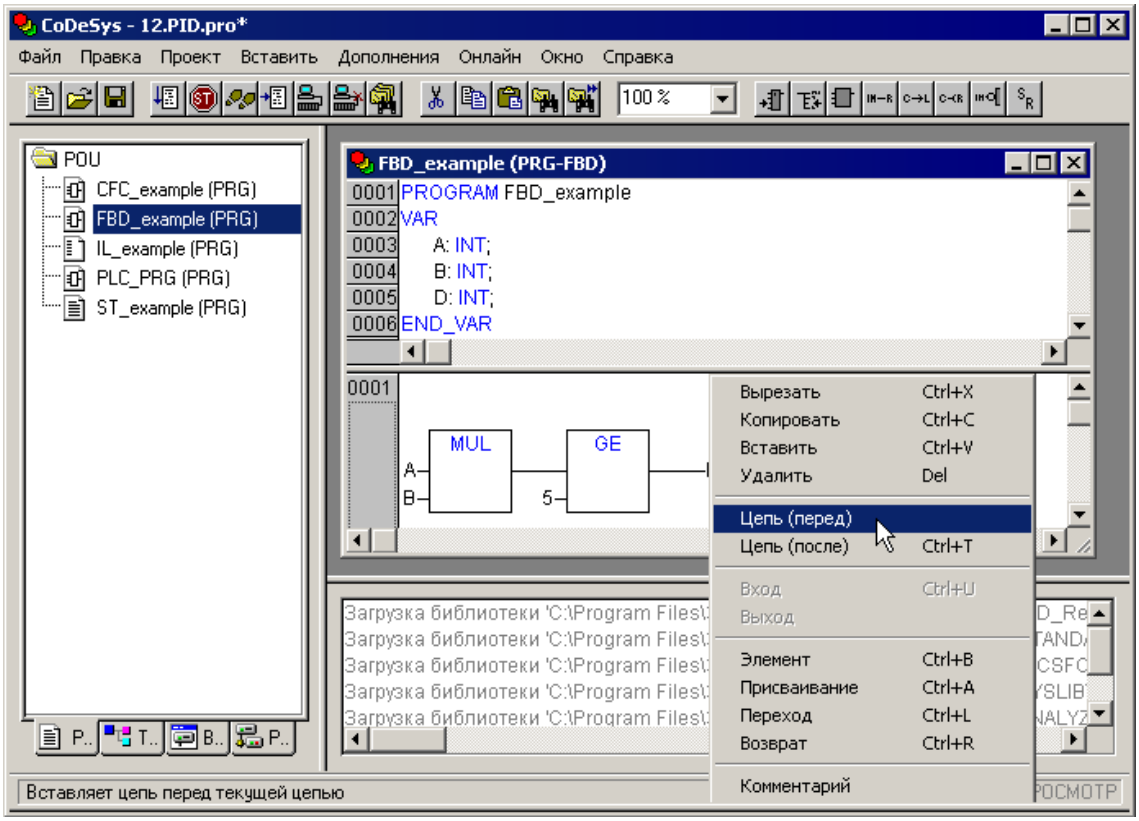


Рисунок 3.13 – Пример окна редактирования FBD программы

3.3.3 Проект. Программные компоненты (POU)

Проект создается в ПО CODESYS на любом из доступных языков программирования. Проект может состоять из одного или нескольких программных компонентов (POU, Program Organization Unit). Главная программа, выполняемая циклически, должна называться PLC_PRG.

К программным компонентам (POU) относятся функциональные блоки, функции и программы. Отдельные POU могут включать действия (подпрограммы).

Каждый программный компонент состоит из раздела объявлений и кода. Для написания всего кода POU используется только один из МЭК языков программирования (IL, ST, FBD, SFC, LD или CFC).

CODESYS поддерживает все описанные стандартом МЭК компоненты. Для их использования достаточно включить в свой проект библиотеку standard.lib (подробнее о библиотеках см. раздел 4.2.4).

POU могут вызывать другие POU, но рекурсии недопустимы.

Кроме того, в проекте могут быть явно определены несколько задач с различными условиями выполнения. Работа с задачами описана в разделе 6.7 «Конфигуратор задач (Task Configuration)» документа «Руководство пользователя по программированию ПЛК в CODESYS 2.3».

Внимание! Если в окне «Конфигурация задач (Task Configuration)» определена последовательность выполнения задач (см. раздел 4.3), то проект может не содержать PLC_PRG.
Нельзя удалять или переименовывать POU PLC_PRG в однозадачном проекте – если не используется «Конфигурация задач (Task Configuration)», то PLC_PRG является главной программой.

Приемы работы при написании программ и примеры программ представлены в разделе 3.3 и в документе «Первые шаги в CODESYS».

3.3.4 Проект. Типы данных

Тип данных определяет род информации и методы ее обработки и хранения, количество выделяемой памяти. Программист может непосредственно использовать элементарные (базовые) типы данных (логический, целочисленные, рациональные, строковые, временные; подробнее см. раздел 4.2.3) или создавать собственные (пользовательские) типы на их основе.

В разделе 4.2.3.1 описаны элементарные (базовые) типы данных.

В разделе 4.2.3.2 описаны пользовательские типы данных.

3.3.5 Проект. Установка связи с ПЛК

Возможность загружать в контроллер пользовательскую программу появляется только тогда, когда установлена связь контроллера и ПК разработчика. Для установки связи могут использоваться различные устройства (каналы) связи: последовательные порты RS-232, сетевой интерфейс Ethernet, интерфейс USB (посредством разъема USB-B). Интерфейс RS-485 предназначен только для обмена данными, но не для программирования.

Также, в некоторых случаях может потребоваться подключение компьютера разработчика к ПЛК, подключенного к другому компьютеру; компьютеры при этом соединены локальной сетью либо сетью Интернет. Такое соединение также возможно, и вариант с локальной сетью описан в данном руководстве, вариант соединения с использованием Интернет по своим настройкам немногим отличается от соединения по локальной сети.

Перед установкой связи ПО CODESYS с контроллером следует однократно настроить рабочий канал связи (используемый интерфейс и настройки обмена данными). В дальнейшем, при отладке программы, настройка связи может потребоваться только при переходе на другой канал связи.

3.3.5.1 Настройка интерфейса связи

Когда проект создан, в CODESYS в рамках проекта включается возможность настройки соединения с контроллером. Чтобы приступить к настройке соединения с контроллером, необходимо выбрать в главном меню «Онлайн» — «Настройка параметров связи». Появится окно настройки каналов связи с контроллерами (см. рис. 3.14 а).

В окне будут отображены настройки, установленные по умолчанию. В левой части окна в иерархическом списке перечислены все созданные каналы, самая верхняя строка (точнее, две строки, где вторая строка появляется при успешном

применении настроек и является ветвью первой строки дерева) – настройки текущего подключения.

В средней части окна отображается таблица настроек канала ввода-вывода, таблица состоит из колонок “Название параметра”, “Значение”, “Комментарии”. Каждый вид подключения имеет свой набор параметров; данные параметры описывают адрес или номер устройства, если в данный канал включено несколько устройств, а также свойства канала связи, например, скорость передачи данных, наличие защиты от помех, временные задержки, и т. п.

Сверху над таблицей расположены две панели, отображающие текст без возможности редактирования, в левой панели — название вида (протокола) связи, используемого в текущем канале.

В правой части окна расположены кнопки для работы со списком каналов связи: “Ok” и “Cancel” для того, чтобы применить и отменить изменения, внесенные в список каналов связи, “New” и “Remove” для того, чтобы создать новую и удалить текущую запись о канале связи, “Gateway…” для определения, с локальными, или с удаленными подключениями осуществляется работа и “Update”.

При нажатии на кнопку “Gateway” на экране появляется окно, показанное на рисунке 3.14 в.

Переключатель “Connection” устанавливает, какое соединение будет использоваться при подключении: локальное (пункт “Local”) или сетевое с использованием протокола TCP/IP (пункт “Tcp/Ip”). В случае локального соединения для настройки подключения нужно установить только параметры порта. В случае подключениня по сети с использованием протокола TCP/IP, в текущем диалоговом окне становятся доступными дополнительные настройки, которыми возможно задать адрес компьютера с подключенным к нему контроллером: либо IP-адрес, либо имя компьютера, в случае использования в сети протокола DNS, либо слово “localhost”, если, несмотря на настройки, требуется установить соединение с локально подключенным контроллером.

Также, при сетевом подключении возможно защитить контроллер помощью пароля, поэтому в настройках подключения присутствует ввод пароля на случай защищенного подключения (поле “Password:”).

По умолчанию, при использовании протокола TCP/IP, на сервере для передачи данных используется IP-сокет 1210, то же значение присутствует по умолчанию в настройках сетевого соединения (поле “Port”:).

Смена подключения с локального на удаленное (и наоборот) приводит к очистке списка подключений в таблице (см. рисунок 3.14 а).

3.3.5.1.1 Установка связи по интерфейсу RS-232 (COM-порт) и USB device

Интерфейс RS-232 (последовательный порт) в примышленной автоматике является самым распространенным и основным интерфейсом взаимодействия микропроцессорных устройств. В персональных компьютерах такие интерфейсы присутствовали и обозначались, как COM-порты (порты COM1, COM2,.. COMn). В современных конфигурациях ПК порты RS-232 нередко отсутствуют, и для подключения различных внешних устройств используют более высокоскоростные USB-порты, обладающие, однако, меньшими возможностями в плане длины соединяющих проводов (до пяти метров).

Для настройки интерфейса RS-232 перед работой с ним требуется проделать следующие действия:

1) После выбора команды «Онлайн | Параметры связи (Online | Communication parameters)» главного меню ПО CODESYS. Откроется окно «Communication parameters», см. рисунок 3.14, а.

2) Нажать кнопку «New» окна «Communication parameters». Откроется окно «Communication parameters: New Channel» (см. рисунок 3.14, б).В этом окне задается имя нового соединения (например, Owen) и выбирается из перечня интерфейс соединения:

«Serial (RS232)» для связи по интерфейсу Debug RS-232 или USB Device;
«TCP/IP (Level 2)» для связи по интерфейсу Ethernet;
«Serial (Modem)» для связи через модем, подключенный к последовательному порту RS-232 или Debug RS-232.

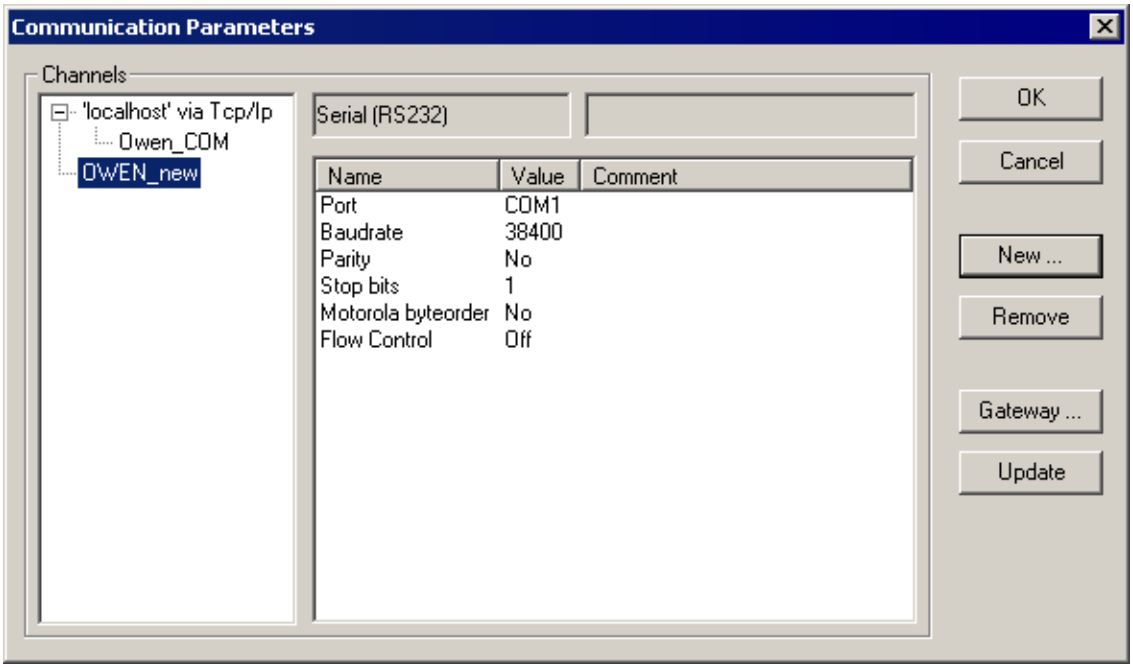
При выборе соединения «Serial (RS232)» в настройках параметров следует задать:

COM-порт (параметр Port), по которому ПЛК подключается к ПК;
скорость соединения (параметр Baudrate) 115200 бит/с;
бит четности (параметр Parity) «No».

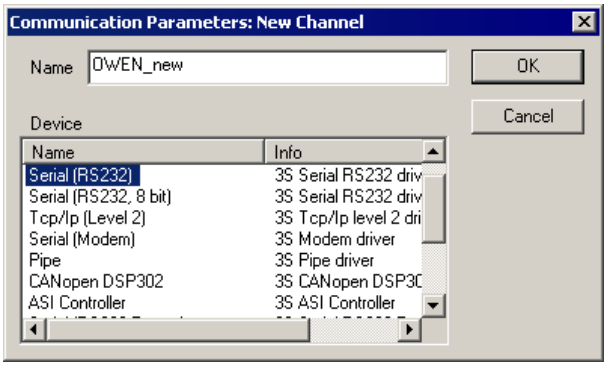
Работа с портом USB ничем не отличается от работы с обычным последовательным портом, увидеть номер данного порта можно также с помощью диспетчера устройств. По умолчанию скорость передачи составляет 115200 бит/с, размер передаваемого слова 8 бит, проверка четности выключена, количество стоповых битов 1).

Драйвер виртуального COM-порта находится на компакт-диске, входящем в комплект поставки. Для установки драйвера необходимо подключить включенный ПЛК к USB-порту ПК стандартным кабелем типа A-B (в комплект поставки не входит). После отключения питания или перезагрузки ПЛК для установки связи может потребоваться повторное отключение и подключение кабеля USB-порта для повторной инициализации драйвера.

а)



б)



в)

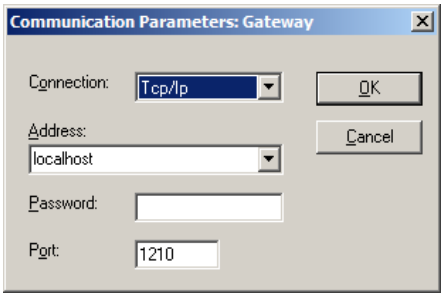


Рисунок 3.14 – Настройка интерфейса для соединения с ПЛК. Окна «Communication parameters» (а), «Communication parameters: New Channel» (б) и «Communication parameters: Gateway» (в).

3.3.5.1.2 Установка связи по интерфейсу Ethernet

Для установки соединения между ПЛК и ПК по интерфейсу Ethernet необходимо выполнение следующих условий:

- IP-адреса должны выбираться из значений, допустимых для частного пользования[2].
- IP-адрес ПК должен быть статическим;
- IP-адреса и маска подсети ПК и контроллера устанавливаются так, чтобы они находились в одной IP-подсети.

Изменение IP-адреса контроллера возможно при помощи команды «SetIP», подаваемой в режиме «ПЛК-Браузер (PLC-Browser)» (подробно о работе в режиме «ПЛК Браузер (PLC-Browser)» см. приложение Ж). При этом связь с контроллером

[2]Согласно правилам распределения IP-адресов, некоторые адреса могут использоваться свободно всеми желающими при организации подсетей с протоколом TCP/IP: адреса 10.x.y.z с маской 255.0.0.0, адреса 172.16.x.y…172.31.x.y с маской 255.240.0.0 и адреса 192.168.x. y с маской 255.255.0.0, здесь x, y, z–целые числа 0…255.

должна быть предварительно установленачерезинтерфейс DebugRS-232 илиUSB Device.
Задание дополнительного IP-адреса ПК осуществляется в свойствах протокола TCP/IP в настройках сетевого окружения Windows. При изготовлении устанавливается IP-адрес контроллера **10.2.11.119,** поэтому необходимо присвоить ПК дополнительный IP-адрес в подсети 10.2.11, отличный от адреса 10.2.11.119; маска подсети при этом задается равной 255.255.0.0. Подробно процесс присвоения дополнительного IP адреса для ПК приведен в видео-инструкции на дистрибутивном диске ПЛК.

3.3.5.1.3 Установка соединения типа «TCP/IP»

Для установки связи с помощью протокола TCP/IP по интерфейсу Ethernet следует проверить состояние сетевого подключения Ethernet на ПК: оно должно быть исправным и включенным. Контроллер и ПК соединить кросс-кабелем Ethernet; если соединение установлено успешно, на экране ПК появится соответствующее сообщение; индикаторы Ethernet-соединения на ПЛК начнут мигать.

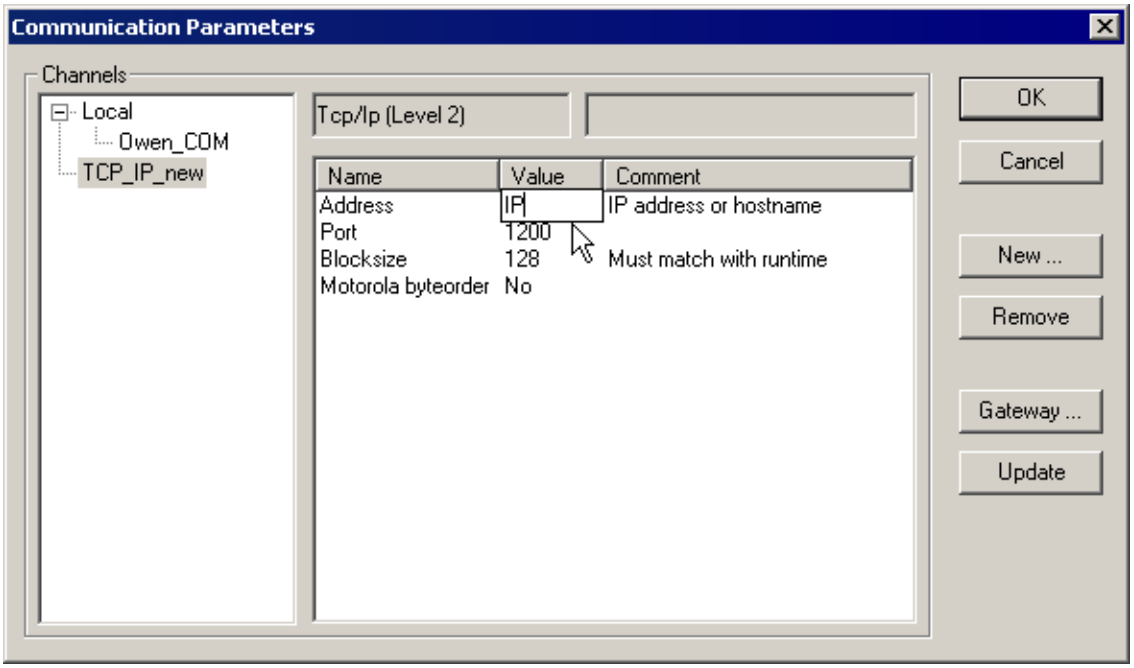
Перед тем, как начать работу с контроллером, подключенным по протоколу TCP/IP, необходимо настроить подключение контроллера к компьютеру. Во-первых, следует перенастроить то сетевое соединение, посредством которого он подключен: ввести статический IP-адрес, и маску подсети. IP-адрес контроллера, если контроллер не использовался ранее, или его настройки были полностью сброшены, имеет значение, указанное выше, в п. 3.3.5.1.2. Настроим сетевое соединение:

Для настройки интерфейса соединения с контроллером следует:

1) Выбрать команду «Онлайн | Параметры связи (Online|Communication parameters)» главного меню ПО CODESYS. Откроется окно «Communicationparameters», см. рисунок 3.15, а.
2) Нажать кнопку «New» окна «Communication parameters». Откроется окно «Communication parameters: New Channel» (см. рисунок 3.15, б).В этом окне задается имя нового соединения (например, TCP_IP_new) и выбирается из перечня интерфейс соединения: «TCP/IP (Level 2)» (обязательно – «TCP/IP (Level 2)») для связи по интерфейсу «TCP/IP».
3) В параметре «Address» – указать значение IP-адреса контроллера и нажать кнопку «Enter».

После задания параметров соединения по интерфейсу Ethernet по протоколу TCP/IP необходимо перезагрузить контроллер. Лучше всего перезагрузить контроллер путем отключения питания и повторного его включения через пять или более секунд.

а)



б)

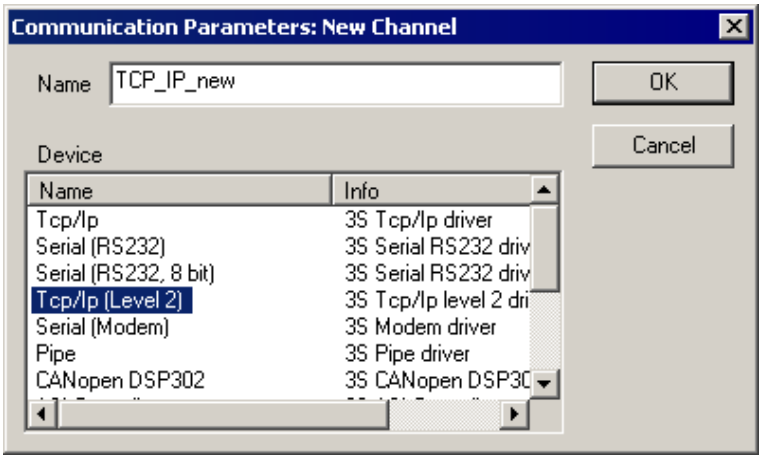
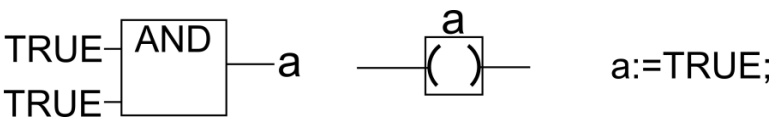


Рисунок 3.15 – Настройка интерфейса для соединения с ПЛК. Окна «Communication parameters» (а) и «Communication parameters: New Channel» (б)

3.3.5.2 Установка связи с контроллером

Для установки связи необходимо, чтобы предварительно была создана программа пользователя, хотя бы простейшая.

Примеры программ на языках FBD, LD и ST, которые можно использовать для проверки связи с контроллером, приведены на рисунке 3.16. Простейшей программой на языке ST является символ «;» (точка с запятой). Такой программы достаточно для проверки связи с контроллером.



а) б) в)

Рисунок 3.16 – Примеры программ на языках FBD (а), LD (б) и ST (в)

Длязагрузкипрограммы в контроллер следует:

1) Выбрать команду «Онлайн | Подключение (Online | Login)» главного меню, тем самым – установить связь с ПЛК. При этом должен быть снят флаг перед строкой меню «Онлайн | Режим эмуляции (Online | Simulation Mode)» (установка и снятие флага производится последовательными щелчками левой кнопкой мыши на строке).
Перед установкой связи ПО скомпилирует проект; и в случае наличия в нем ошибок – прервет установку связи.

Сразу после установки связи среда программирования предложит загрузить (см. рисунок 3.17) или обновить код пользовательской программы в оперативной памяти контроллера.

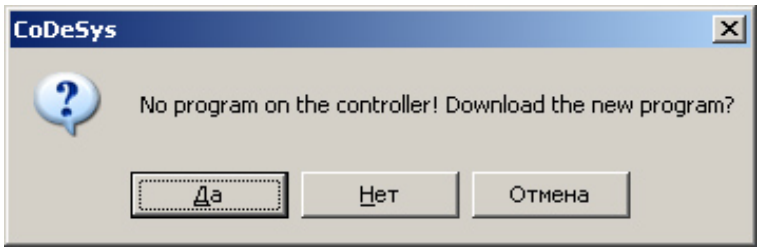


Рисунок 3.17 – Окно предложения загрузки программы

Для ПЛК110 поддерживается режим «ONLINECHANGE», позволяющий обновить выполняющуюся программу без прерывания ее работы (см. рисунок 3.18).

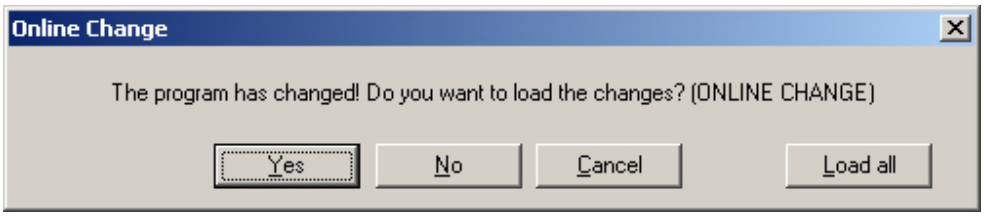


Рисунок 3.18 – Окно предложения обновления программы в режиме «ONLINECHANGE»

При компиляции программы по умолчанию используется самая последняя версия компилятора. Новые версии компиляторов разрабатываются с выполнением правил совместимости между версиями, тем не менее, если по каким-либо непонятным причинам программный код, созданный пользователем, не является работоспособным, смена версии компилятора может решить проблему.

В составе дистрибутива CODESYS присутствует несколько версий компилятора для обеспечения максимальной совместимости среды с различными типами контроллеров, версиями прошивок и приложениями. Установить версию компилятора возможно из настроек проекта среды разработки CODESYS: это можно сделать следующим образом:

1. Выбрать в главном меню CODESYS пункт “Проект” – “Опции”. На экране появится окно, в котором задаются параметры проекта, в том числе и опции компилятора (см. рисунок 3.19).

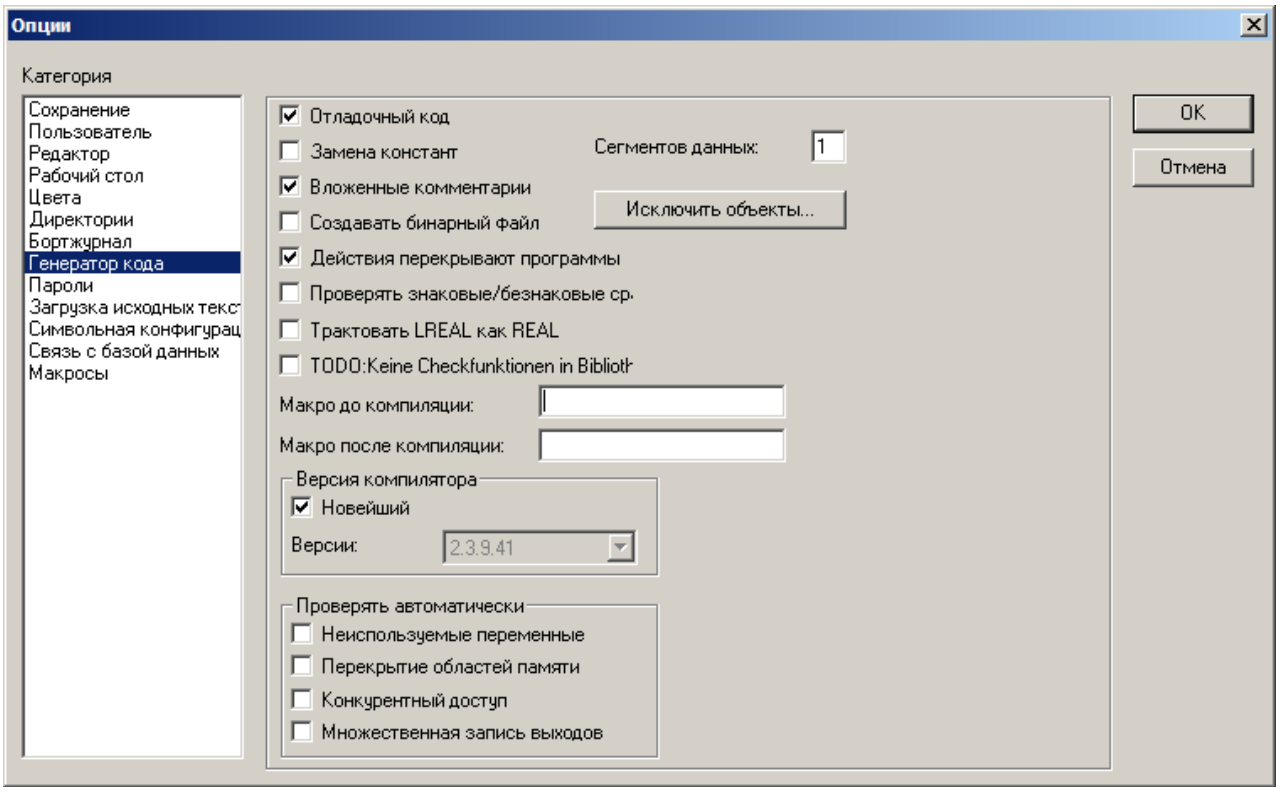


Рисунок 3.19 – Окно настройки параметров проекта с параметрами генерации кода

2. Выбрать в списке слева “Генератор кода” – в поле справа появится интерфейс для настройки параметров генерации кода, в числе параметров присутствует панель “Версия компилятора”. На панели по умолчанию присутствует чекбокс “Новейший”, который по умолчанию установлен и переключатель с выпадающим списком “Версии”, по умолчанию заблокированный.
3. Снять отметку в чекбоксе “Новейший” и выбрать версию компилятора из ставшего доступным списка “Версии” (см. рисунок 3.20).

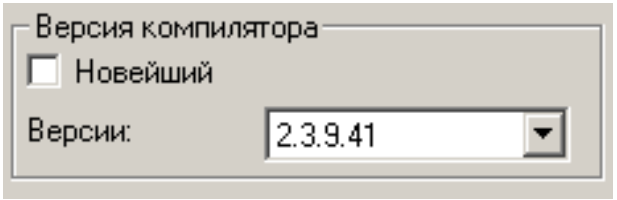


Рисунок 3.20 – Панель выбора версии компилятора

В настоящее время компания “ОВЕН” рекомендует для проектов использовать версию компилятора CODESYS 2.3.8.1.

3.4 Конфигурирование области ввода-вывода ПЛК

Перед созданием программы рекомендуется настроить конфигурацию входов, выходов и интерфейсов связи ПЛК с внешними устройствами (модулями ввода-вывода, устройствами индикации и т.д.), обмен данными с которыми будет производиться по сети. Перечисленные устройства обмениваются данными с пользовательской программой через специальную область памяти: Память ввода-вывода. Конфигурация ее задается в окне режима («Ресурса») «Конфигурация ПЛК (PLC

Configuration)» ПО CODESYS и подробно описана в разделе 7 настоящего документа.

Размер памяти ввода-вывода определяется типом лицензии CODESYS контроллера ПЛК (см. раздел 2.2).

Подробное описание процедур конфигурирования области ввода-вывода приведено в разделе 4 .

3.4.1 Расчет потребности ПЛК в памяти ввода/вывода

Наиболее простым способом, позволяющим точно проверить, хватит ли доступного объема памяти ввода/вывода для выполнения проекта, является способ создания проекта. Не приобретая контроллер, но установив на компьютере ПО CODESYS и Target-файл, можно создать проект, в котором подключить все необходимые модули. Подключение производится в режиме («Ресурс») «Конфигурация ПЛК (PLCConfiguration)», см. раздел 7 . При компиляции проекта либо компиляция пройдет успешно, либо CODESYS сообщит об ошибке, если памяти недостаточно. При этом в проекте будут учтены все особенности, в том числе особенности выравнивания адресов переменных (см. раздел 7.4.2.2). Для осуществления подобной проверки собственно программу контроллера писать не требуется.

Альтернативным, более сложным, способом потребности ПЛК в памяти ввода/вывода рассчитываются по следующей схеме:

1) Для подсчета потребности ПЛК в памяти ввода / вывода, необходимой для работы с приборами ОВЕН, следует воспользоваться данными, приведенными в таблице 3.1.

Таблица 3.1 – Потребности ПЛК в памяти ввода/вывода, требуемой для работы с некоторыми приборами производства компании ОВЕН

Прибор	Объем требуемой памяти (байт) при использовании различных протоколов передачи данных					
	ОВЕН		ModBus		DCON	
	%I, байт	%Q, байт	%I, байт	%Q, байт	%I, байт	%Q, байт
ТРМ2хх. Один аналоговый вход	-	4	-	-	-	-
ТРМ151, ТРМ148, ТРМ133. Один аналоговый вход	-	4	-	-	-	-
ИП320. Одна переменная на чтение с ПЛК	-	-	-	2	-	-
ИП320. Одна переменная на запись в ПЛК	-	-	-	2	-	-
МВ110-2А. Один аналоговый вход	4	-	4	-	4	-
МВ110-8А. Один аналоговый вход	4	-	4	-	4	-
МУ110-8И. Один аналоговый выход	-	2	-	2	-	-
МУ110-6У. Шесть аналоговых выходов	-	2	-	2	-	-

2) При использовании приборов других производителей, работающих по протоколам ModBus или DCON, следует по руководствам на эти приборы определить, сколько байт данных содержат команды, посылаемые по сети. При работе с приборами ввода количество этих байт надо прибавить к размеру области %I, при работе с приборами вывода количество надо прибавить к размеру области %Q.
3) Для дискретных модулей ввода/вывода сторонних производителей, ра-

ботающих по протоколу ModBus, как правило, значение одного входа или одного выхода кодируется одним битом. Соответственно, занимаемый размер памяти в области ввода/вывода следует считать в битах, но с учетом того, что на один модуль тратится целое число байт. Таким образом, на двенадцатиканальный модуль дискретного ввода потребуется два байта, из 16 бит которых только 12 будут значащими.

4) Для приборов и операторских панелей, работающих по протоколу ModBus, передача одного значения параметра осуществляется как минимум в двухбайтном регистре (даже если параметр – однобайтовый).
5) Дополнительно при использовании модуля архивации на каждую архивируемую переменную следует в памяти %Q зарезервировать место, равное размеру этой переменной.
6) При использовании модулей Master сетевых протоколов (т.е. модулей, организующих обмен с внешними устройствами и модулями) дополнительно следует учесть, что эти модули содержат ряд служебных переменных, также расположенных в области памяти вывода %Q. Один модуль Master одного сетевого протокола дополнительно требует от 4 до 8 байт.
7) После подсчета необходимого размера областей памяти %I и %Q следует провести проверку достаточности объема доступной памяти каждого типа. При этом следует учитывать, что часть памяти занимается собственными входами и выходами.
8) Если расчет показал, что резерва памяти нет, то следует приобретать контроллер без ограничения области памяти ввода/вывода, так как из-за принятого в CODESYS способа выравнивания адресов переменных в памяти ввода/вывода может возникнуть дополнительный расход памяти. Алгоритм выравнивания описан в разделе 7.4.2.2, но учитывать особенности выравнивания при расчете потребности в памяти ввода/вывода не рекомендуется из-за сложности расчета.

3.5 Визуализация

ПО CODESYS позволяет создать одно или несколько окон-визуализаций, в которых пользователь может располагать визуальные элементы, позволяющие графически отобразить данные из пользовательской программы. Данные в визуализацию передаются из контроллера, при установленой с ним связи (подробнее см. п. 3.3.5).

В режиме «Online» представление элементов на экране изменяется в зависимости от значений переменных.

Например, если уровень заполнения емкости жидкостью доступен в программе в виде значения некоторой переменной, то в окне визуализации он может быть изображен графическим элементом в виде полосы, которая, в зависимости от значения переменной проекта, будет изменять свою длину и/или цвет. Рядом может быть размещен текст, отображающий в виде числа текущий результат измерения. Здесь же можно разместить и, например, кнопки запуска и остановки программы.

Создание окна визуализации выполняется на вкладке «Визуализации» Организатора объектов ПО CODESYS (см. рисунок 3.21).

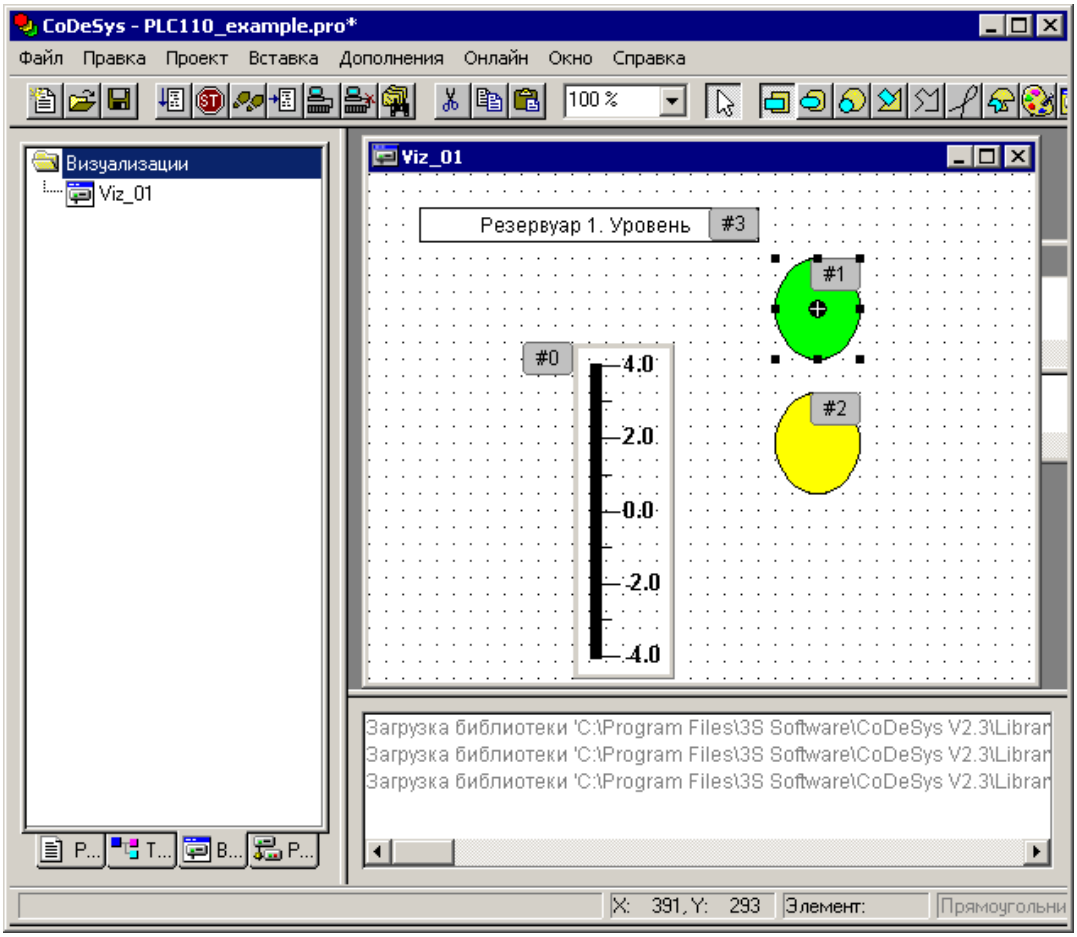


Рисунок 3.21 – Окно визуализации проекта

3.6 Сохранение проекта

Созданный проект следует сохранить в виде файла на жестком диске персонального компьютера для дальнейшей работы. Начальное сохранение проекта производится вызовом команды «Файл | Сохранить как (File | Save as)», последующие сохранения изменений – вызовом команды «Файл | Сохранить (File | Save)» или нажатием кнопки «Сохранить» ( ) панели инструментов.

Проект может быть также сохранен на встроенный в контроллер Flash-диск. Это позволяет хранить проект непосредственно в контроллере, что снижает вероятность его потери. Для загрузки проекта на встроенный Flash-диск контролера следует после установки связи с контроллером (подробнее об установке связи см. раздел 3.3.5) выбрать команду «Онлайн | Загрузка исходных текстов (Online | Sourcecode Download)» главного меню.

Проект CODESYS может быть сохранен совместно с конфигурацией, т.е. со структурой, описанной в Target-файле, загруженном при вызове проекта. Такой способ сохранения на несколько килобайт увеличивает сохраняемый файл проекта, но позволяет в дальнейшем не заботиться о совместимости проекта и версии Target-файла, установленного в системе на момент редактирования проекта. В таком режиме рекомендуется сохранять проект в случае, когда предполагается возможность редактирования проекта попрошествии значительного времени с момента его создания. Включение режима сохранения проекта совместно с конфигурацией производится в окне «Конфигурация ПЛК (PLC Configuration)», установкой флажка переключателя «Сохранять конфигурационные файлы в проекте (Save configuration file in project)», см. рисунок 3.22.

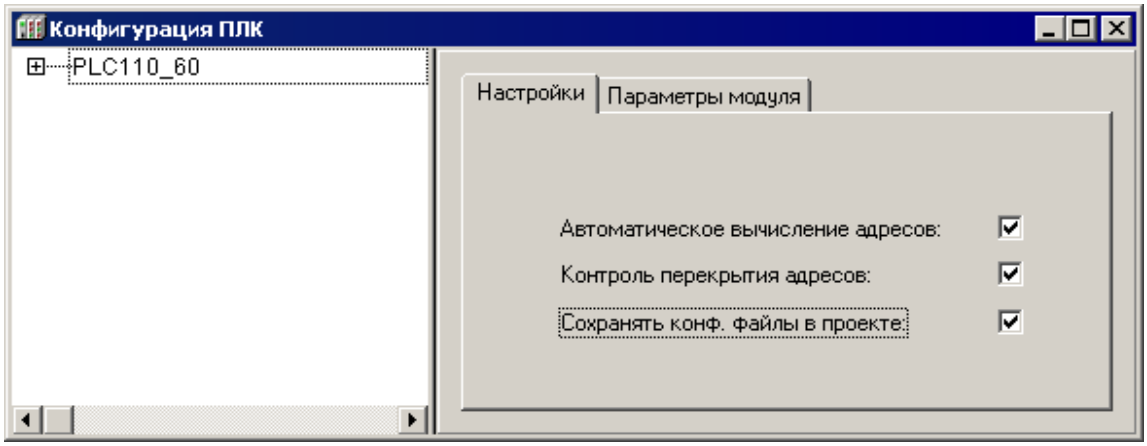


Рисунок 3.22 – Окно «Конфигурация ПЛК (PLC Configuration)». Установка опции сохранения конфигурации в файл проекта

3.7 Определения состояния микроконтроллера по сигналам светодиодных индикаторов на передней панели

Работая с контроллером (программирование, отладка, работа с flash-памятью), необходимо контролировать его состояние, обозначаемое светодиодными индикаторами, расположенными на передней панели корпуса контроллера. Таких индикаторов четыре:

- «Работа», обозначает различные режимы работы контроллера: загрузку операционной системы (ядра), успешность или неуспешность загрузки программы, ее работу;
- «Питание», светится, когда на контактах питания устройства присутствует напряжение питания, также светится две секунды после выключения контроллера;
- «Связь», светится, когда установлена связь с устройством любым способом:
- «Бат.», светится, когда в контроллер установлена батарея аварийного питания и подано питающее напряжение на вход питания.

Светодиод «Работа» в процессе работы контроллера выдает сигналы следующего содержания: *тусклое свечение* – сразу после включения, пока не загружена операционная система («ядро»), если «ядро» оказалось поврежденным (проверка его контрольных сумм успешно не завершилась), светодиод начинает *мигать*, если несколько попыток перезагрузить контроллер окончились миганием светодиода, необходимо переустановить встроенное программное обеспечение контроллера (см. гл. 10), если и это не привело к нормальной загрузке «ядра», необходимо отправить контроллер на ремонт. Если после загрузки операционной системы светодиод «Работа» светится ярко и постоянно, то загрузка прошла успешно, и контроллер готов к работе, при этом запускается установленное на него пользовательское приложение.

Если пользовательское приложение (программа, созданная в CODESYSи записанная boflash-память контроллера) загрузилось и запустилось, светодиод «Работа» продолжает ярко светиться, если же пользовательское приложениене загрузилось или не запустилось, светодиод «Работа» *гаснет.* При запуске программы.светодиод «Работа» начинает светиться.

Также на передней панели контроллера расположены светодиодные индикаторы состояний дискретных входов и выходов. Каждый индикатор соответствует одному дискретному (как быстрому, так и обычному) входу. При работе контроллера любой из индикаторов светится в случае значения логической единицы на соответ-

ствующем ему входе или выходе, и не светится, когда на входе или выходе значение логического нуля. Уровни напряжений, соответствующие логическим нулю и единице приведены в РЭ контроллера.

3.8 Запуск пользовательской программы

Для запуска загруженной программы следует выбрать команду «Онлайн | Старт (Online | Run)» главного меню; отладочный запуск программы осуществляется только таким способом.[3]

После запуска программы следует проверить ее работоспособность, эмулировав или воссоздав необходимые сигналы на входах контроллера или на подключенных модулях ввода и удостовериться в правильности управления выходами или модулями вывода.

Запуск программы после выключения контроллера (текущая, записанная в ПЗУ) или сброса контроллера из CODESYS (текущая, хранящаяся в ОЗУ, выбор пункта меню “Онлайн” - “Сброс (горячий)” или “Онлайн” - “Сброс (холодный)”) происходит, если трёхпозиционный тумблер находится в положении “Работа”. Если тумблер нахолится в положении “Стоп”, то запуска не происходит.

3.9 Сохранение программы в памяти контроллера

После написания и отладки программы ее необходимо записать во внутреннюю Flash-память контроллера, выбра вкоманду «Онлайн | Создание загрузочного проекта (Online | Create boot project)» главного меню. После этого программа сохраняется в памяти контроллера после отключения питания или перезагрузки и будет автоматически запускаться на контроллере при перезагрузкеи при включении питания.

Внимание! Ресурс встроенной Flash-памяти контроллера ограничен (около 50 000 циклов записи), поэтому не рекомендуется при отладке программы каждый раз записывать ее во Flash-память.

Если контроллер циклически перегружается из-за ошибок в программе, сохраненной во Flash-памяти, или некорректной записи программы во Flash-память, следует до или сразу после перезагрузки контроллера установить трехпозиционный тумблер в положение “Стоп”. Программа из Flash-памяти не будет автоматически запущена, что даст возможность подключиться к контроллеру и загрузить в него корректно работающую программу.

[3] Установка трехпозиционного тумблера “Работа – Стоп - Сброс”, в положение “Работа” или “Стоп” на функционирующем контроллере не приведет к запуску или остановке программы.

4 Написание программы

В данном разделе приводится пример разработки программы (проекта).

Подготовительный этап создания пользовательской программы (проекта) – установка ОС и ПО CODESYS описаны в разделе 2 .

Этапы создания пользовательской программы (проекта) описаны в ле 3 данного руководства.

Установка настроек целевой платформы описана в разделе 3.2 .

Пользовательская программа («проект») ПЛК в ПО CODESYS содержит программные компоненты (POU), типы данных, визуализации, ресурсы и библиотеки, сведения о ресурсах ПЛК и некоторую другую информацию, хранимую в одном файле («name.pro»).

4.1 Программные компоненты проекта

К программным компонентам (POU) относятся функциональные блоки, функции и программы. Отдельные POU могут включать действия (подпрограммы).

Каждый программный компонент состоит из раздела объявлений и кода. Для написания всего кода POU используется только один из МЭК языков программирования (IL, ST, FBD, SFC, LD или CFC).

CODESYS поддерживает все описанные стандартом МЭК 61131 компоненты. Для их использования достаточно включить в свой проект библиотеку standard.lib (подробнее о библиотеках см. раздел).

POU могут вызывать другие POU, но рекурсии недопустимы.

Кроме того, в проекте могут быть явно определены несколько **задач** с различными условиями выполнения. Работа с задачами описана в разделе 6.7 «Конфигуратор задач (Task Configuration)» документа «Руководство пользователя по программированию ПЛК в CODESYS 2.3».

Приемы работы при написании программ и примеры программ представлены в документе «Первые шаги в CODESYS».

4.1.1 Программы

Программа это программный компонент (POU), способный формировать произвольное число значений во время вычислений. Значения всех переменных программы сохраняются между вызовами. В отличие от функционального блока (см. ниже), экземпляров программы не существует. Программа является глобальной во всем проекте.

Нельзя вызывать программу из функции (см. ниже).

Если вызвать программу, которая изменит значения своих переменных, то при следующем вызове ее переменные будут иметь те же значения, даже если она вызвана из другого POU.

В этом заключается главное различие между программой и функциональным блоком, в котором изменяются только значения переменных данного экземпляра функционального блока. Список объявлений программы (в редакторе объявлений) начинается с ключевого слова PROGRAM и следующего за ним имени программы.

Пример записи программы на языке IL приведен на рисунке 4.1

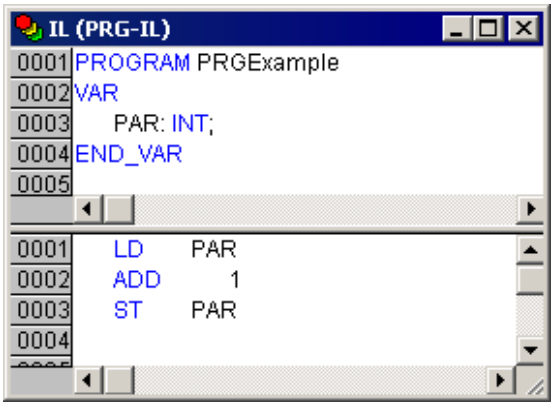


Рисунок 4.1 – Пример записи программы на языке IL

4.1.2 Функции

Функция – это программный компонент (POU), который возвращает только единственное значение (которое может состоять из нескольких элементов, если это битовое поле или структура). В текстовых языках функция вызывается как оператор и может входить в выражения.

При объявлении функции необходимо указать тип возвращаемого значения. Для этого после имени функции нужно написать двоеточие и тип (см. рекомендации по наименованию в приложении J документа «Руководство пользователя по программированию ПЛК в CODESYS 2.3»). Правильно объявленная функция выглядит следующим образом: **FUNCTION Fct: INT**;

Имя функции используется как выходная переменная, которой присваивается результат вычислений.

Пример функции, написанной на языке IL, использующей три входных переменных (**par1** – **par3**) целочисленного типа (INT; диапазон изменения – от минус 32768 до 32767) и возвращающей результат деления произведения первых двух на третью.Список объявлений функции (в редакторе объявлений) начинается с ключевого слова FUNCTION и следующего за ним имени функции, за которым, отделенное двоеточием, указывается название типа возвращаемого значения.

Пример записи функции на языке IL приведен на рисунке 4.2.

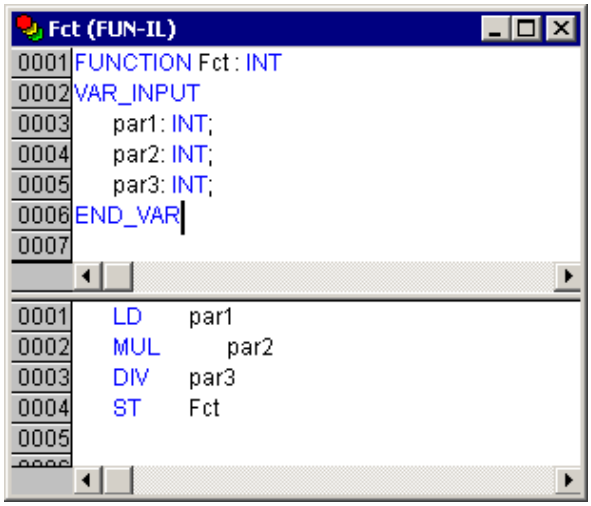


Рисунок 4.2 – Пример записи функции на языке IL

В языке ST вызов функции может присутствовать в выражениях как операнд.
В SFC функция вызывается только из шага или перехода.

Примечание. *Функция не имеет внутренней памяти, но CODESYS допускает использование в функциях глобальных переменных. Это является отклонением от требований стандарта МЭК 61131-3, в соответствии с которыми выходное значение функции должно зависеть исключительно от входных параметров. Т.е. функция с одними и теми же значениями входных параметров всегда должна возвращать одно и то же значение.*

4.1.3 Функциональный блок

Функциональный блок – это программный компонент (POU), который принимает и возвращает произвольное число значений. В отличие от функции (см. ниже), функциональный блок не формирует возвращаемое значение.

Список всех объявлений функционального блока (в редакторе объявлений) начинается с ключевого слова FUNCTION_BLOCK и следующего за ним имени блока.

Функциональный блок может имеет один или несколько экземпляров (копий).

На рисунке 4.3 приведен пример функционального блока, написанного на IL, который имеет две входных и две выходных переменных. Значение выходной переменной MULERG равно произведению значений двух входных переменных, а значение VERGL определяется в результате сравнения значений входных переменных.

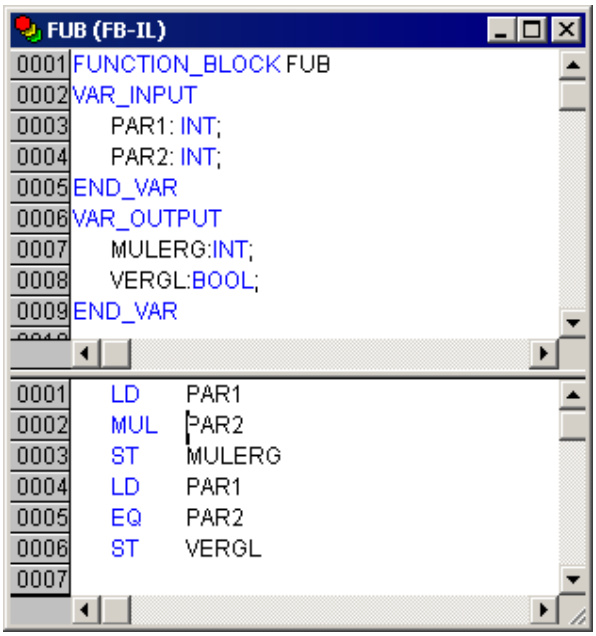


Рисунок 4.3 – Пример записи функционального блока на языке IL

4.2 Использование переменных

Программные компоненты (POU) проекта обрабатывают переменные – величины, значения которых могут меняться в ходе выполнения программы (в частных случаях переменные, обрабатываемые программой, могут быть и константами). Переменные могут использоваться для хранения и передачи промежуточных результатов выполнения логических операций, значений состояний входов или выходов функциональных блоков программы, значений состояний входов или выходов ПЛК и др. Каждая переменная имеет идентификатор.

4.2.1 Типы переменных

Переменные, используемые в ПО CODESYS, могут принадлежать к нескольким типам.

Во-первых, переменные могут относитьмся к **локальным** или к **глобальным** переменным. Локальные переменные могут использоваться только в рамках текущего компонента POU, глобальные переменные – могут использоваться в рамках всего проекта (во всех программных компонентах, входящих в его состав). Локальные переменные задаются в редакторе объявлений (см. раздел 3.3.2), глобальные – в аналогичном редакторе, вызываемом выбором объекта «Глобальные переменные (GlobalVariables)» .

Кроме того, используемые переменные могут относиться к **входным** или **выходным**, а также к переменным, одновременно являющимся **входными и выходными**.

При необходимости сохранять значения переменных они могут быть объявлены как **перманентные переменные** – такие переменные сохраняют свои значения при определенных сбоях в системе. Они бывают сохраняемые и постоянные.

Сохраняемые переменные обозначаются при объявлении ключевым словом RETAIN. Эти переменные сохраняют свои значения, даже если произошла авария питания (выключение и включение) контроллера, что равносильно команде «Сброс» (Онлайн | Сброс (Online | Reset')). Значения RETAIN переменных сохраняются в энергонезависимой памяти контроллера.

Постоянные переменные обозначаются ключевым словом PERSISTENT. В отличие от сохраняемых переменных эти переменные сохраняют свои значения только при загрузке кода новой программы, но не при выключении питания или любом сбросе. Значения постоянных переменных размещаются вне энергонезависимого ОЗУ.

Подробнее о типах переменных см. Руководство пользователя ПО CODESYS.

4.2.2 Объявление переменных

Для использования в POU переменная должна быть **объявлена**. Объявление переменных производится в «Редакторе объявлений» (см. раздел 3.3.2).

«Редактор объявлений» отображается в верхней части окна редактирования POU, открывающегося при выборе существующего или при добавлении нового программного компонента (в дереве программных компонентов на вкладке «POU» Организатора объектов).

Редактор объявлений используется для объявления переменных POU, глобальных переменных, описания типов данных.

В разделе объявлений зарезервированные слова, типы данных и сами переменные автоматически выделяются разными цветами (см. рисунок 4.4).

Наиболее важные команды можно найти в контекстном меню, которое появляется по щелчку правой кнопки мыши или по нажатию сочетания клавиш <Ctrl>+<F10>.

Локальные переменные POU объявляются в разделе объявлений редактора программного компонента. Такими переменными могут быть входные и выходные переменные, переменные, одновременно являющиеся входными и выходными, локальные переменные, сохраняемые переменные и константы.

Синтаксис, используемый при объявлении переменных, соответствует стандарту МЭК61131-3.

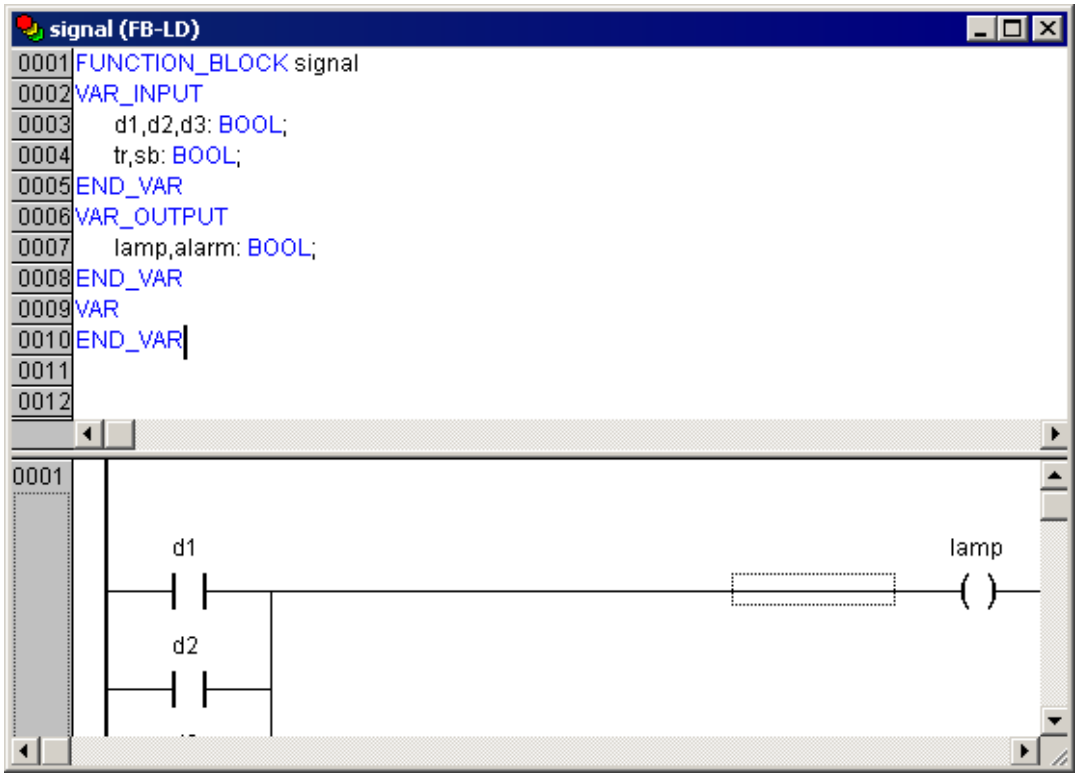


Рисунок 4.4 – Окно редактора объявлений (верхняя часть окна ркедактора POU)

Переменные объявляются следующим образом:

<Идентификатор> {AT <Адрес>}:<Тип> {:=<начальное значение>};

Части, заключенные в фигурных скобках, не обязательны.

Имена переменных не должны содержать пробелов и специальных символов, должны объявляться только один раз и не должны совпадать с зарезервированными словами. Регистр букв в имени переменной не имеет значения (т.е. переменные Var1, VAR1 и var1) не различаются. В именах переменных допустим знак подчеркивания: (переменные A_BCD и AB_CD) считаются разными. Идентификатор не должен содержать подряд более одного символа подчеркивания. Длина индентификатора не ограничена, все символы являются значимыми.

Все переменные и типы данных можно инициализировать. Для этого используется оператор «=». Переменные простейших типов инициализируются константами. По умолчанию все переменные инициализируются нулем.

Пример:

iVar1:INT:=12; (*Переменная типа INT, инициализируемая числом 12*).

Если требуется поместить переменную по определенному адресу, то следует объявить ее с ключевым словом **AT**.

4.2.2.1 Методы объявления переменных

В ПО CODESYS применяются три метода объявления переменных: текстовый (описан выше, в разделе 4.2.2), табличный и автоматический.

4.2.2.2 Табличное объявление переменных

Табличный способ объявления переменных позволяет ускорить процедуру объявления переменных. Для вызова окна табличного объявления следует выбрать команду «Объявления в форме таблицы (Declarations as Tables)» контекстного меню окна редактора объявлений. Окно редактора объявлений примет вид, изображенный

на рисунке 4.5. На вкладках окна редактора отображаются списки переменных различных типов. В ячейках таблицы списки переменных могут быть дополнены новыми переменными. Значения атрибутов переменных могут быть введены или отредактированы также в ячейках таблицы. Кроме того, требуемые переменные могут быть не только отредактированы, но и удалены из списков.

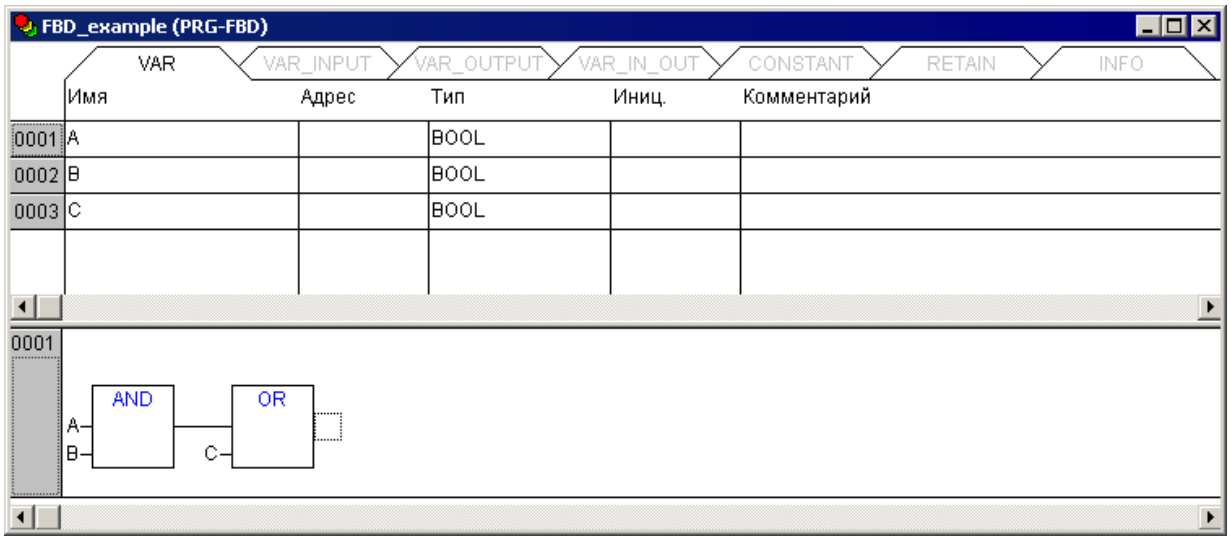


Рисунок 4.5 – Окно редактора объявлений (в табличной форме) (верхняя часть окна редактора POU)

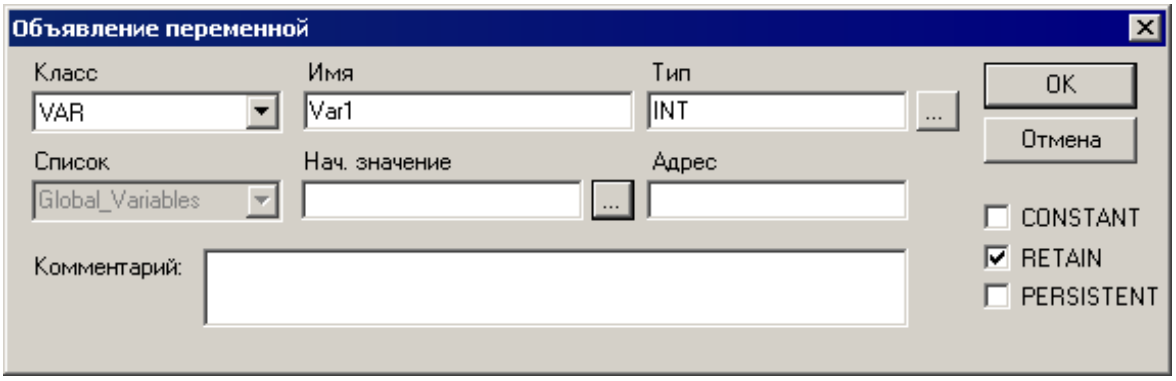
4.2.2.3 Автоматическое объявление переменных

Автоматическое объявление переменных позволяет автоматизировать ввод значений ряда атрибутов переменной, что позволяет ускорить и упростить процедуру ввода и одновременно избежать ошибок, возможных при ручном вводе.

Для вызова окна автоматического объявления переменных следует выбрать команду «Автообъявление (Auto Declare)» контекстного меню окна редактора объявлений.

В открывшемся окне (см. рисунок 4.6, а) задается имя добавляемой переменной (в поле «Имя»). В других полях окна значения задаются выбором либо из раскрывающегося списка, либо из списков, отображаемых в специальных окнах. Например, выбор требуемого типа переменной производится в окне «Ассистент ввода» (см. рисунок 4.6, б), которое открывается по нажатию кнопки с тремя точками, размещенной у правого края поля «Тип».

а)



б)

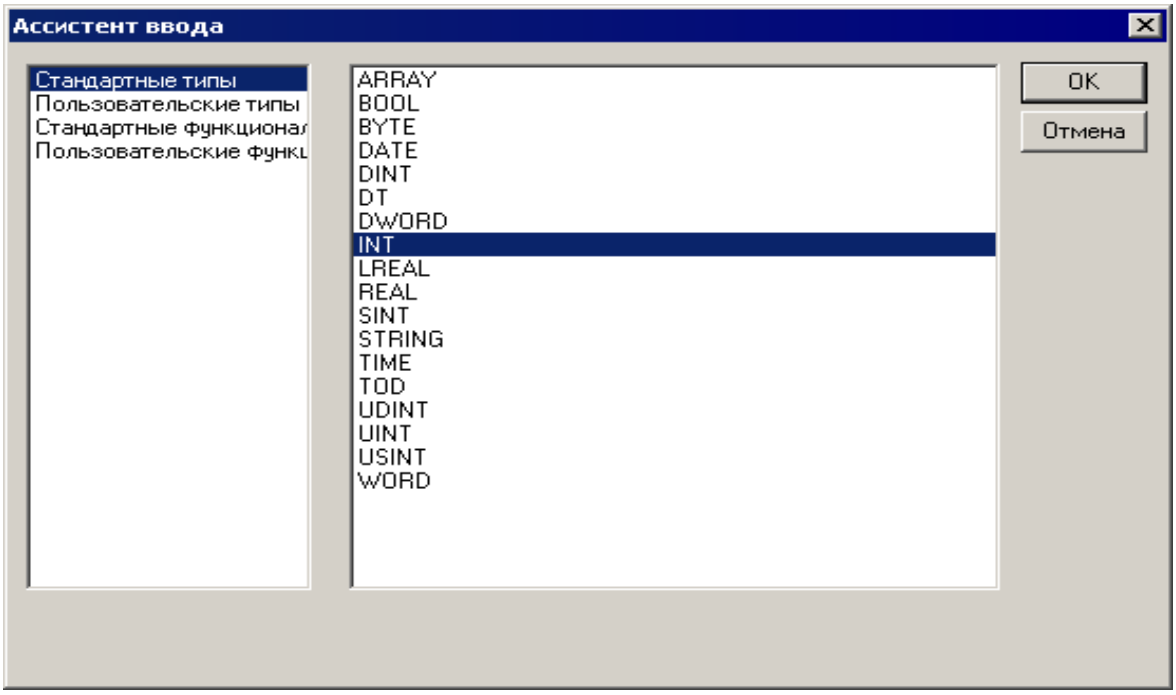


Рисунок 4.6 – Окна «Объявление переменной» (а) и «Ассистент ввода» (б)

4.2.3 Типы данных

Тип данных определяет род информации, методы ее обработки и хранения, количество выделяемой памяти. Программист может непосредственно использовать элементарные (базовые) типы данных или создавать собственные (пользовательские) типы на их основе.

4.2.3.1 Базовые типы данных

4.2.3.1.1 Логический (BOOL)

BOOL – логический тип данных. Переменная может принимать 2 значения: ИСТИНА (TRUE) или ЛОЖЬ (FALSE). Занимает 8 бит памяти (если не задан прямой битовый адрес).

4.2.3.1.2 Целочисленные

BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT и **UDINT** – целочисленные типы данных.Они отличаются диапазонами сохраняемых данных и требованиями к памяти. Подробно их характеристики приведены в таблице 4.1.

Таблица 4.1 – Характеристики целочисленных типов данных

Тип	Нижний предел	Верхний предел	Размер памяти
BYTE	0	255	8 Бит
WORD	0	65535	16 Бит
DWORD	0	4294967295	32 Бит
SINT	-128	127	8 Бит
USINT	0	255	8 Бит
INT	-32768	32767	16 Бит
UINT	0	65535	16 Бит
DINT	-2147483648	2147483647	32 Бит
UDINT	0	4294967295	32 Бит

4.2.3.1.3 Рациональные

REAL и **LREAL** – данные в формате с плавающей запятой, используются для сохранения рациональных чисел. Для типа REAL необходимо 32 бита памяти, для LREAL – 64 бита.

Диапазон значений REAL: от [1.175494351e-38] до [3.402823466e+38].

Диапазон значений LREAL: от [2.2250738585072014e-308] до [1.7976931348623158e+308].

4.2.3.1.4 Строки

Строковый тип STRING представляет строки символов. Максимальный размер строки определяет количество резервируемой памяти и указывается при объявлении переменной. Размер задается в круглых или квадратных скобках. Если размер не указан, принимается размер по умолчанию – 80 символов.

Длина строки в CODESYS не ограничена, но строковые функции способны обращаться со строками от 1 до 255 символов.

Пример объявления строки размером до 35 символов:

str:STRING(35):='Просто строка";

4.2.3.1.5 Время и дата

TIME представляет длительность интервалов времени в миллисекундах. Максимальное значение для типа TIME: 49d17h2m47s295ms (4194967295 ms).

TIME, **TIME_OF_DAY** (сокр. **TOD**) содержит время суток, начиная с 0 часов (с точностью до миллисекунд). Диапазон значений **TOD:** от 00:00:00 до 23:59:59.999.

DATE содержит календарную дату, начиная с 1 января 1970 года. Диапазон значений от: 1970-00-00 до 2106-02-06.

DATE_AND_TIME (сокр. DT) содержит время в секундах, начиная с 0 часов 1 января 1970 года. Диапазон значений от: 1970-00-00-00:00:00 до 2106-02-06-06:28:15.

Типы **TIME, TOD, DATE** и **DATE_AND_TIME** (сокр. **DT**) сохраняются физически как **DWORD**.

4.2.3.2 Пользовательские типы данных

Кроме стандартных типов данных (см. раздел 4.2.3.1), в проектах можно использовать определяемые пользователем сложные типы данных: массивы, перечисления, структуры и некоторые другие (см. раздел 5).

4.2.4 Подключение дополнительных программных модулей

Подключение требуемых дополнительных программных модулей (библиотек) производится в окне «Менеджер библиотек (Library Manager)» ПО CODESYS (см. рисунок 4.7).

Вызов окна производится выбором команды Окно | Менеджер библиотек (Window | Library Manager)» или выбором пункта «Менеджер библиотек (Library Manager)» в дереве ресурсов (на вкладке «Ресурсы» Организатора объектов).

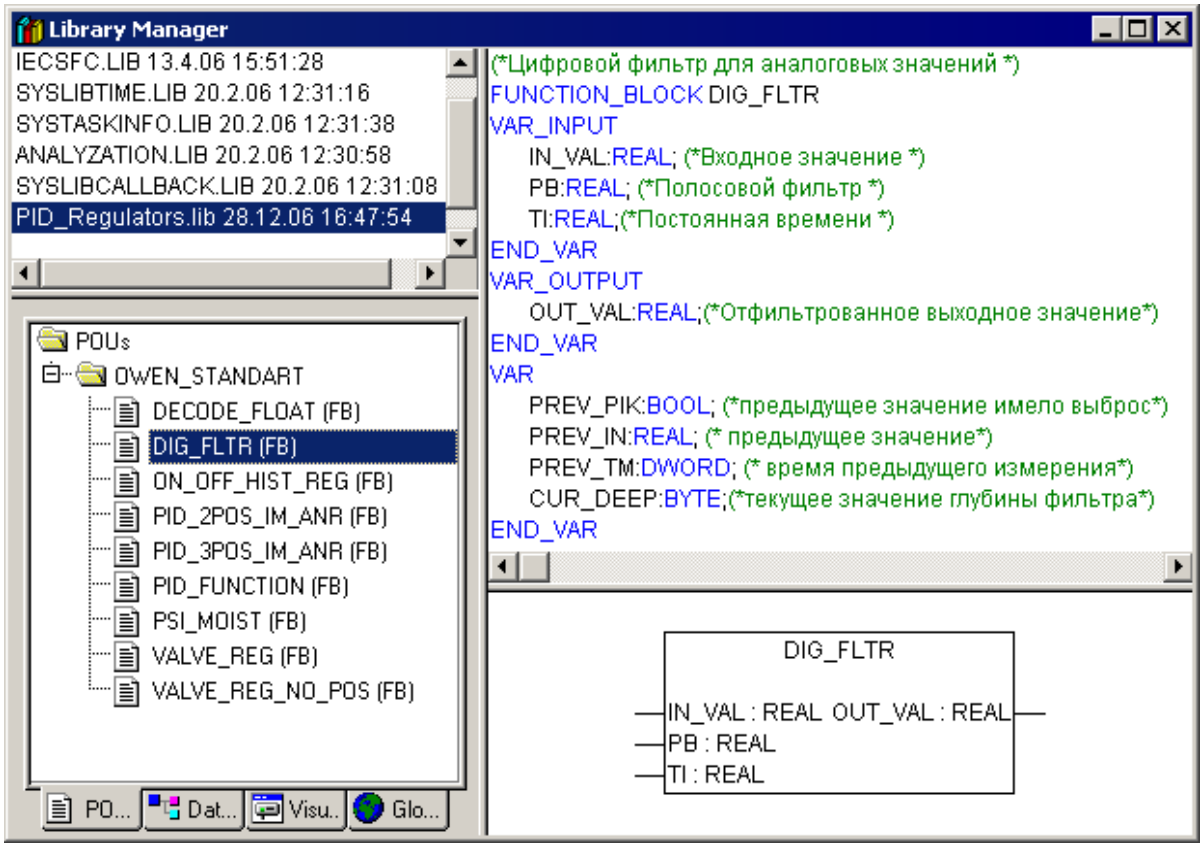


Рисунок 4.7 – Окно «Менеджер библиотек (Library Manager)»

Для подключения библиотеки следует:

1) Выбрать команду «Добавить библиотеку (Add library)» контекстного меню списка подключенных библиотек (отображаемого в верхней левой области окна режима) или команду «Вставка | Добавить библиотеку (Insert | Additional Library)» главного меню.
2) В открывшемся окне выбора файлов следует выбрать файл требуемой библиотеки и нажать кнопку «Открыть». Выбранная библиотека будет подключена к проекту. Ее наименование отобразится в списке установленных библиотек (в верхней левой области окна режима).

Для удаления подключенной библиотеки следует:

1) Выделить требуемую запись в списке подключенных библиотек (отображаемого в верхней левой области окна режима).
2) Выбрать команду «Удалить (Delete)» контекстного меню списка. Выделенная библиотека будет отключена от проекта.

Для того, чтобы включить в проект дополнительный программный модуль (то есть модуль, содержащийся в подключенной к проекту библиотеке), следует выполнить следующие операции:

1) Перейти на вкладку «POU» Организатора объектов.
2) В дереве программных компонентов объекта выбрать требуемый.
3) Выбрать команду «Правка | Ассистент ввода (Edit | Input Assistant)» главного меню или команду «Ассистент ввода (Input Assistant)» контекстного меню редактора объявлений.
4) В открывшемся окне «Ассистент ввода» (см. рисунок 4.8), в левой части, где отображается перечень доступных типов добавляемых объектов, – выделить требуемый тип (в данном случе – «Стандартные функциональные блоки»).
   В правой части окна при этом отобразится перечень доступных объектов выбранного типа. При этом: если флажок переключателя «Структурно» в нижней части окна установлен, то перечень отображается в виде иерархического структурированного списка, см. рисунок 4.5; если флажок не установлен, то перечень отображается в виде отсортированного по алфавиту линейного списка.
5) В перечне доступных объектов (в правой части окна) – выбрать требуемый объект и нажать кнопку «ОК» окна. Выбранный объект (в данном случае – стандартный функциональный блок) будет вставлен в редактируемый программный компонент проекта. Для отказа от добавления блока – нажать кнопку «Отмена» окна.

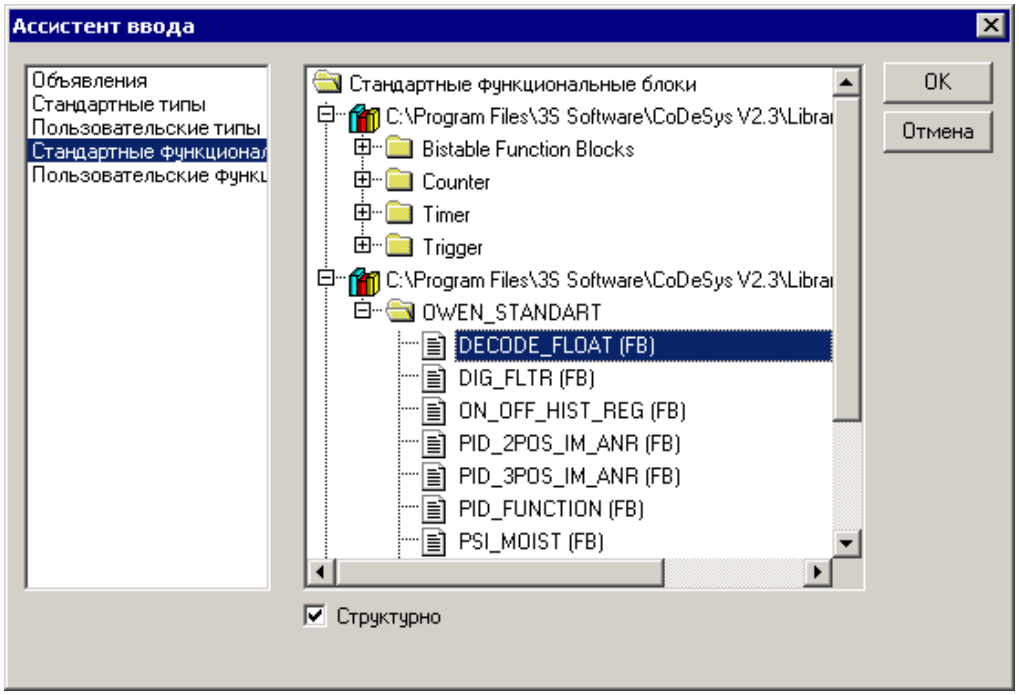


Рисунок 4.8 – Окно «Ассистент ввода (Input Assistant)»

4.2.4.1 Доступные дополнительные программные модули

В дистрибутив ПЛК включен ряд дополнительных программных модулей (библиотек).

Подробное описание библиотек ОВЕН приведено в документе «Библиотеки программных компонентов для ПЛК. Руководство по применению».

Описания системных библиотек CODESYS доступны на сайте компании 3S Software и на дистрибутивном диске ПЛК.

ПЛК110 поддерживают следующие библиотеки программных компонентов:

- Standart.lib
- Util.lib
- SysLibCallback.lib
- SysLibSockets.lib
- SysLibCom.lib
- SysLibTime.lib
- SysLibProjectInfo.lib
- SysLibMem.lib
- SymLib.lib
- SysLibPorts.lib
- OwenLibFileAsync.lib
- OwenLibUSBSerial.lib
- OwenLibNetControl.lib
- OwenLibFactorySetups.lib
- UNM.lib
- PID_Regulators.lib
- Pid_reg2.lib
- Timer.lib.

4.2.4.2 Модуль работы с файлами

Встроенное программное обеспечение контроллера в новой версии предусматривает работу с файловыми системами: поддерживаются функции сохранения в файлы произвольных данных, чтение, удаление, копирование, переименование файлов и др., данные функции реализованы в отдельном модуле – **OwenLibFileAsync.lib .**

Имеется возможность сохранять файлы на три устройства хранения:

- внешний накопитель, подключенный к порту USB-host (например, flash-память, или жесткий диск),
- внутренний flash-накопитель,
- внутренний виртуальный диск – RAM Диск – 64кБайт.

На носителях поддерживаются подкаталоги, а также доступны операции с файлами в директориях и просмотр содержимого директорий.

Устройство, на которое файл будет записываться, задается с помощью префикса к имени файла: “ffs:” для внутренней flash-памяти, “ram:” для виртуального диска и “usb:” для внешнего накопителя, подключенного к порту USB-host. Например, чтобы записать файл с именем “file1.txt” на разные носители, необходимо преобразовать его в “ffs:file1.txt” для записи во внутреннюю flash-память, “ram:file1.txt” – для записи на виртуальный диск и “usb:file.txt” – для записи на внешний накопитель.

На ПЛК запущен TFTP сервер для передачи данных по протоколу TFTP (RFC 1350 - THE TFTP PROTOCOL (REVISION 2) http://tools.ietf.org/html/rfc1350). Особенностью является : только RAM диск доступен для TFTP протокола, а также отсутствие функции разграничения доступа. Сервер доступен на всех интерфейсах ПЛК по соответствующим IP адресам на порту 69.

Внимание! Не рекомендуется использовать встроенную Flash-память для записи часто переписываемых файлов, так как ее ресурс ограничен (≈50 000 циклов записи). В создаваемых программах для контроллера рекомендуется программировать сохранение файла с предпочтением внешнего накопителя при его наличии.

Внимание! Виртуальный диск расположен в оперативном запоминающем устройстве контроллера, поэтому все файлы, записанные под именами с префиксом “ram:”, будут храниться в контроллере до первого его выключения. Чтобы сохранить эти файлы, необходимо их копировать на flash-носители – внутренний или внешний.

Внимание! Допускается работа программы пользователя одновременно не более чем с пятью файлами.

При работе с USB Host следует учесть следующие особенности:

– Поддержка USB Hub-ов. Однако суммарное потребление хабом и подключёнными устройствами не должно превышать 0,5 А.

– USB HOST имеет функцию защиты от перегрузки и короткого замыкания. Срабатывание защиты приводит к выключению питания на USB HOST с последующими периодическими попытками восстановления питания.

– К ПЛК могут быть подключены USB MassStorage и USB HID устройства. Общее число MSD и HID устройств не должно превышать 1 для каждого типа. Остальные устройства игнорируются. Инициализация устройств в порядке подключения. Если, к примеру, 2 USB MSD уже подключены и подаётся питание на ПЛК — порядок их инициализации непредсказуем.

– Стек USB HOST поддерживает опрос не менее 1 устройства класса USB HID.

– Используя библиотеку OwenLibFileAsync.lib пользователь может получать сообщения от HID устройства, например мыши и/или клавиатуры.

– В ПЛК поддерживается класс USB MSD. Поддерживается только файловая система FAT (12, 16, 32). Ограничения на размер накопителя налагаются только ограничениями файловой системы FAT.

Рекомендуется использование файловой системы FAT 32!

Запись на USB MSD происходит без кэширования, т.о. для безопасного удаления накопителя необходимо:

- Завершить все процедуры записи (остановить запись из программы через библиотеку OwenLibFileAsync и закрыть открытые файлы, дождаться завершения всех системных загрузок файлов, остановить работу модуля(ей) архивации, остановить опрос файлов через 0x20 функцию ModBus slave)
- Дождаться прекращения активности на накопителе (если индикация активности присутствует) или выждать не менее 3 секунд.
- Удалить накопитель

Не рекомендуется подключать USB MSD устройства на базе жёстких дисков и SDD без дополнительного внешнего питания, т.к. это может привести к перегрузке по питанию, и циклическому включению/выключению внешнего диска.

Внимание! Не гарантируется корректная работа ПЛК с USB устройствами, если последние обратно не совместимы с протоколом USB 1.1

В приборе реализован асинхронный механизм доступа к файлам из программы пользователя. Асинхронный доступ гарантирует отсутствие задержек в выполнении программы пользователя при доступе к файлам, в т.ч. расположенным на внешних носителях. Библиотека **OwenLibFileAsync.lib** предоставляет интерфейс для создания и обработки запросов к файловой системе на открытие, чтение, запись, модифицирование и т.п. файлов в асинхронном режиме. Основной особенностью использования данной библиотеки является – выполнение функций в два этапа. А именно:

- на первом этапе необходимо подать команду для работы с файлом;
- на втором этапе проверяется завершенность выполнения указанной команды и разрешается давать следующие команды для дальнейших операции с файлом.

Библиотека OwenLibFileAsync.lib рекомендуется для всех новых разработок.

В имеющейся версии библиотеки реализованы функции работы с файлами, перечисленные в таблице 4.2.

Таблица 4.2 – Функции работы с файлами в асинхронном режиме

Название функции	Краткое описание функции
OwenSysFileCloceAllOpenAsync	Данный функциональный блок закрывает все открытые файлы. Не нужно сообщать имена или дескрипторы файлов, поскольку они все уже известны системе.
OwenSysFileCloseAsync	Используется для закрытия файла. После закрытия файл освобождается для других процессов, дескриптор более не имеет значения.
OwenSysFileWriteAsync	Используется для записи данных в файл. Файл должен быть предварительно успешно открыт с помощью SysFileOpenAsync.
OwenSysFileReadAsync	Используется для чтения данных из файла. Файл должен быть предварительно успешно открыт с помощью SysFileOpenAsync
OwenSysFileDeleteAsync	Удаление файла с заданным именем.
OwenSysFileGetPosAsync	Возвращает позицию (смещение от начала файла в байтах) записи и чтения в файл.
OwenSysFileSetPosAsync	Задает позицию записи и чтения в файл.
OwenSysFileEOFAsync	Возвращает TRUE, если текущая позиция чтения-записи находится в конце файла, иначе возвращает FALSE.
OwenSysFileGetSizeAsync	Возвращает размер файла с заданным именем.
OwenSysFileGetTimeAsync	Возвращает время создания, последнего доступа и последнего изменения файла с заданным именем.
OwenSysFileCopyAsync	Копирование файла с заданным именем в файл с другим именем.
OwenSysFileRenameAsync	Переименование (перенос) файла с заданным именем.
OwenFileOpenAsync	Используется для открытия существующего или создания нового файла. Выход hFile (DWORD) сообщает дескриптор файла. Он используется другими функциональными блоками для работы с данным файлом.

4.2.4.2.1 Функция SysFileOpenAsync

Функция возвращает значение типа DWORD, используется для открытия существующего или создания нового файла. Возвращаемое значение – дескриптор файла, либо '0' в случае ошибки. Дескриптор файла используется для доступа к открытому файлу другими функциями библиотеки. В некоторых случаях сообщение об ошибке имеет вид 16#FFFFFFFF (в десятичной системе это 4 294 967 295 при интерпретации как беззнакового числа или -1 при интерпретации, как числа со знаком).

Входные переменные:

- **FileName** типаSTRING – имя файла;
- **Mode** типа STRING – режим работы с файлом, может имеет следующие значения:
  “w”, если требуется открыть файл только для записи, при этом, если файл существовал до начала записи, то он будет стерт и создан пустой файл с заданным именем;
  “r”, если требуется открыть файл только для чтения;
  “rw”, если требуется открыть файл и для записи и для чтения;
  “a”, аналогично “w”, но если файл существовал до начала операции, данные будут дописываться в конец файла.

Пример открытия файла «a» и разрешение на переход к дальнейшим операциям с файлом на языке ST:

```
0:
res:=OwenFileOpenAsync(filename,'a',ADR(handle));
IF res=ASYNC_WORKING THEN
    state:=1;
END_IF

1:
res:=OwenFileOpenAsync(filename,'a',ADR(handle));
IF res=ASYNC_DONE THEN
    IF handle<>0 THEN
        state:=2;
    ELSE
        state:=0;
    END_IF
ELSIF res<0 THEN
    state:=0;
END_IF
```

4.2.4.2.2 Функция SysFileCloseAsync

Функция закрывает файл, открытый ранее функцией SysFileOpenAsync, возвращает значение типа BOOL, которое равно TRUE при успешном закрытии файла, иначе (например, если файл не был открыт) FALSE. Входным параметром является дескриптор закрываемого файла.

Входные переменные:

- **hFile** типа DWORD - дескриптор файла, число, которое возвратила функция SysFileOpenAsync

Пример закрытия файла «a» и разрешение на переход к дальнейшим операциям программы на языке ST:

```
– res:=OwenFileCloseAsync(handle,ADR(result));
– IF res=ASYNC_WORKING THEN
    ▪ state:=7;
– ELSE
    ▪ state:=0;
– END_IF

– 7:
– res:=OwenFileCloseAsync(handle,ADR(result));
– IF res=ASYNC_DONE THEN
    ▪ IF result=0 THEN
      • state:=8;
    ▪ ELSE
      • state:=8;
    ▪ END_IF
– ELSIF res<0 THEN
    ▪ state:=8;
– END_IF
```

4.2.4.2.3 Функция SysFileWriteAsync

Функция записи данных в файл, открытый с помощью SysFileOpenAsync, возвращает значение типа DWORD – количество записанных байт данных.

Входные переменные:

- **File** – типа DWORD – дескриптор файла, число, которое возвратила функция SysFileOpenAsync;
- **Buffer** – адрес буфера, содержащего данные, которые необходимо записать в файл, число, которое возвратила функция ADR с аргументом – именем переменной-буфера; тип – массив, например, массив байт, или строка.
- **Size** – типа DWORD – размер буфера в байтах, можно использовать функцию SIZEOF с аргументом – именем переменной-буфера.

Пример на рисунке 4.9 языке ST, записывающая данные в файл «a» значение «buffout».

```
– 2:
–     res:=OwenFileWriteAsync(handle,ADR(bufout),14,ADR(result));
–     IF res=ASYNC_WORKING THEN
–         state:=3;
–     ELSE
–         state:=6;
–     END_IF
–
–     3:
–     res:=OwenFileWriteAsync(handle,ADR(bufout),14,ADR(result));
–     IF res=ASYNC_DONE THEN
–         IF result=14 THEN
–             state:=4;
```

```
–                ELSE
–                        state:=6;
–                END_IF
–        ELSIF res<0 THEN
–                state:=6;
–        END_IF
```

Рисунок 4.9 – Использование функции записи в файл

4.2.4.2.4 Функция SysFileReadAsync

Функция чтения данных из файла, открытого с помощью SysFileOpenAsync, возвращает значение типа DWORD– количество считанных байт данных.

Входные переменные:

- **File** – типа DWORD – дескриптор файла, число, которое возвратила функция SysFileOpenAsync;
- **Buffer** – адрес буфера, содержащего данные, которые необходимо записать в файл, число, которое возвратила функция ADR с аргументом – именем переменной-буфера; тип – массив, например, массив байт или строка.
- **Size** – типа DWORD – размер буфера в байтах, можно использовать функцию SIZEOF с аргументом – именем переменной-буфера.

Использование функции – аналогично SysFileWriteAsync (см. рисунок 4.9).

4.2.4.2.5 Функция SysFileDeleteAsync

Функция удаления файла, возвращает значение типа BOOL: TRUE при успешном удалении, FALSE при ошибках.

Входная переменная – **FileName –** типа STRING – имя удаляемого файла.

4.2.4.2.6 Функция SysFileGetPosAsync

Функция возвращает число типа DWORD – фактическое смещение в байтах от начала до текущей позиции в открытом файле с заданным дескриптором; число определяет «место» в файле, откуда будет считана или куда будет записана информация если операция чтения или записи будет произведена без дополнительного предварительного смещения позиции (см. п. 4.2.4.2.7).

Входная переменная – **File** – типа DWORD– дескриптор открытого файла.

4.2.4.2.7 Функция SysFileSetPosAsync

Функция устанавливает для открытого файла с заданным дескриптором позицию чтения (записи) с помощью заданного смещения; возвращает значение типа BOOL. Если позиция была установлена, то возвращается значение TRUE, иначе возвращается FALSE.

Входные переменные:

- **File** – типа DWORD – дескриптор открытого файла, в котором необходимо задать текущую позицию чтения (записи);
- **Pos** – типа DWORD – позиция чтения (записи), заданная смещением в байтах от начала файла.

4.2.4.2.8 Функция SysFileEOFAsync

Функция, определяющая достигнут ли коней файла. Возвращает значение типа BOOL, равное TRUE, если текущим положением позиции чтения (записи) является конец файла, иначе FALSE.

Входной переменной функции является **File** – типа DWORD – дескриптор открытого файла, в котором и проверяется достижение текущей позицией конца.

4.2.4.2.9 Функция SysFileGetSizeAsync

Функция возвращает значение типа DWORD – размер файла в байтах; входной переменной функции является **FileName** типа STRING –имя файла.

4.2.4.2.10 Функция SysFileGetTimeAsync

Функция определяет значения даты и времени создания, последнего изменения и последнего доступа к файлу. Возвращает значение типа BOOL, которое равно TRUE при успешном завершении выполнения функции и FALSE в случае любой ошибки (например, доступа к файлу). Для хранения трех значений времени в одной переменной используется структура FileTime, которую можно описать так:

```
TYPE FILETIME
    STRUCT
        dtCreation:DT; (* Дата и время создания файла *)
        dtLastAccess:DT;(* Дата и время последнего доступа *)
        dtLastModification:DT; (* Дата и время последнего изменения файла *)
    END_STRUCT
END_TYPE
```

Входные переменные:

- **FileName** – типа STRING – имя файла;
- **ftFileTime** – типа POINTER TO FILE TIME – адрес структуры, в которую будут сохраняться считанные данные о времени создания файла, последнего доступа к нему и его последней модификации. Извлекается с помощью функции ADR.

Пример на рисунке 4.10 – программа на языке ST, считывающая структуру со значениями времени создания файла.

```
VAR
        file_time:POINTER TO FILETIME;
        returnvalue:POINTER TO DWORD;
        w: FILETIME;
END_VAR

    8:
        OwenFileGetTimeAsync(filename, file_time, returnvalue);
        w:=file_time;

    ELSE
        state:=0;
    END_CASE
```

Рисунок 4.10 – Использование функции возврата значений времени

4.2.4.2.11 Функция SysFileCopyAsync

Функция копирования одного файла в другой, файлы задаются именами. Возвращает значение типа UDINT, в котором содержится количество действительно скопированных байт.

Входные переменные:

- **FileDest** – типа STRING – имя копии файла (файл-приемник);
- **FileSource** – типа STRING – имя копируемого файла (файл-источник).

4.2.4.2.12 Функция SysFileRenameAsync

Функция переименования файла. Возвращает значение типа BOOL, равное TRUE в случае успешного переименования, или FALSE в случае ошибки. Ошибка может возникать, например, если попытаться переименовать открытый файл.

Входные переменные:

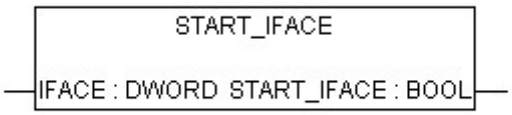
- **FileOldName** – типа STRING – текущее имя файла;
- **FileNewName** – типа STRING – новое имя файла.

4.2.4.3 Модуль контроля выхода в интернет

Библиотека **OwenLibNetControl.lib** предназначена для и управления сетевыми устройствами в обновленном ПЛК. Библиотека позволяет включать/выключать интерфейсы (если это разрешено), получать статус интерфейса, его тип, атрибуты и адреса интерфейса (MAC, IP, APN, телефонный номер и т.п.). Подключение библиотеки производится аналогично стандартным библиотекам среды CodeSys.

В ПЛК есть 2 интерфейса — Ethernet и PPP. Они имеют, соответственно, номера 0 и 1.

4.2.4.3.1 Функция START_IFACE



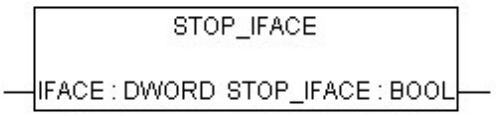
Входные переменные :

- ***IFACE*** – DWORD – номер запускаемого интерфейса

Выходные переменные:

- ***START_IFACE*** – BOOL – значение указывающие на успешность выполнения команды. При возвращении значения «**0**» - Запуск произошёл успешно.

4.2.4.3.2 Функция STOP_IFACE



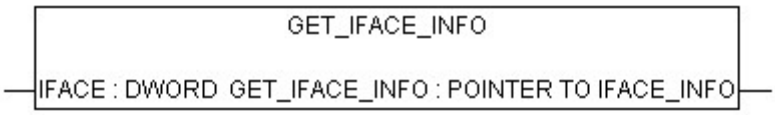
Входные переменные:

- ***IFACE*** – DWORD – номер закрываемого интерфейса

Выходные переменные:

- ***STOP_IFACE*** – BOOL – значение указывающие на успешность выполнения команды закрытия интерфейса. При возвращении значения «**0**» - закрытие произошло успешно.

4.2.4.3.3 Функция GET_IFACE_INFO



Входные переменные:

- ***IFACE*** – DWORD – номер интерфейса

Выходные перменные:

- ***GET_IFACE_INFO*** – POINTER TO IFACE_INFO – возвращает указатель на структуру, содержащую информацию о интерфейсе:

```
TYPE IFACE_INFO :
STRUCT
    name: STRING(79);                    (*Имя интерфейса*)
    itype: IFACE_TYPES;                  (*Тип интерфейса*)
    addreses: ARRAY[0..9] OF ARRAY[0..31] OF BYTE; (*Все адреса интерфейса (MAC, IP, имя порта) список зависит от типа*)
    atributes: IFACE_ATRIBUTES;   (*Атрибуты интерфейса*)
END_STRUCT
END_TYPE
```

IFACE_TYPES — типы интерфейсов

Поле структуры параметра **GET_IFACE_INFO**. Поле может принимать следующие значения:

```
NO_IFACE 0          - нет интерфейса
ETHERNET_IFACE 1    - интерфейс - Ethernet
PPP_IFACE 2         - использование модема
```

IFACE_ATRIBUTES

Поле структуры параметра **GET_IFACE_INFO**. Поле может принимать следующие значения:

```
HAVE_STATUS 1       - Возвращает статус
AUTOSTART 2         - Запускается автоматически
 USE_DHCP 4         - Настроен в режиме DHCP
```

IFACE_ADDRESS_TYPES

Массив из 10 полей типа Array [0..31] of BYTES

Каждый элемент массива содержит информацию об определённых адресах интерфейса:

IFACE_MAC_ADDRESS 0 -массив содержащий MAC-адрес
IFACE_IP_ADDRESS 1 -массив содержащий IP-адрес
IFACE_IP_MASK 2 -массив содержащий маску сети
IFACE_IP_GATE 3 -массив содержащий шлюз сети
IFACE_TEL_NUMBER 4 -массив содержащий телефонный номер текущего дозвона
IFACE_APN_NAME 5 -массив содержащий адрес APN- оператора.

4.2.4.3.4 Функция GET_IFACE_STATUS

Входные переменные:

- ***IFACE*** – DWORD – номер интерфейса

Выходные переменные:

- ***GET_IFACE_STATUS*** – IFACE_STATUS – возвращает статус модема, битовые поля которого имеют следующие значения, по битам:

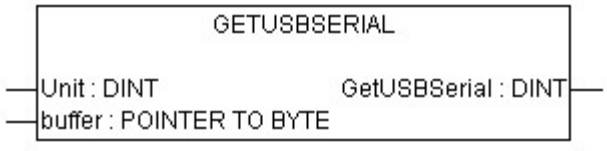
IFACE_NOT_PRESENT - если возвращается 0 — интерфейс не запущен,
IFACE_PRESENT :0 Интерфейс запущен
IFACE_CONFIGURED :1 Интерфейс настроен (есть настройки)
IFACE_CONNECTED :2 Есть физическое соединение
IFACE_WORKING :3 Есть логическое соединение
IFACE_SEND_DATA :4 В течении 5 секунд была отправлена посылка
IFACE_READ_DATA :5 В течении 5 секунд была получена посылка
IFACE_HAVE_ERROR :6 Есть ошибки
IFACE_NO_IFACE - если возвращается 16#FFFF — такого интерфейса нет

4.2.4.4 Модуль получения серийного номера USB-устройства

Библиотека OwenLibUSBSerial.lib предназначена для получения статуса подключённых к USB HOST устройств и доступа к их серийному номеру.

Библиотека содержит только одну функцию – GETUSBSERIAL.

4.2.4.4.1 Функция GETUSBSERIAL



Входные параметры:

- Unit – DINT - Номер устройства, обычно 0
- Buffer - POINTER TO BYTE – Указатель на массив из 24байт, куда будет записан серийный номер.

Выходные параметры:

- GetUSBSerial – DINT - Возвращает значения:
  - 0: Чтение серийного номера прошло успешно;
  - (-1): Устройство USB не подключено;
  - (-2): Чтение серийного номера произошло с ошибкой

Массив содержащий серийный номер USB расшифровывать следующим образом:

Offset	Field	Size	Value	Description
0	bLength	1	N+2	Size of this descriptor in bytes
1	bDescriptorType	1	Constant	STRING Descriptor Type
2	wLANGID[0]	2	Number	LANGID code zero
……				
N	wLANGID[x]	2	Number	LANGID code x

Или строка (вместо серийного номера)

Offset	Field	Size	Value	Description
0	bLength	1	Number	Size of this descriptor in bytes
1	bDescriptorType	1	Constant	STRING Descriptor Type
2	bString	N	Number	UNICODE encoded string

4.2.5 Создание и использование дополнительных программных модулей

При необходимости дополнительные программные модули могут быть разработаны и применены пользователем. Такая необходимость может возникнуть в том случае, если применяемая программа должна содержать алгоритмы, которые не могут быть написаны с использованием готовых программных модулей.

Для создания пользовательского программного модуля следует проделать следующие действия.

1) Создать новый проект (см. раздел 3.3.2).
2) В рамках проекта – создать объект типа «Функциональный блок» (например, см. рисунок 4.11.

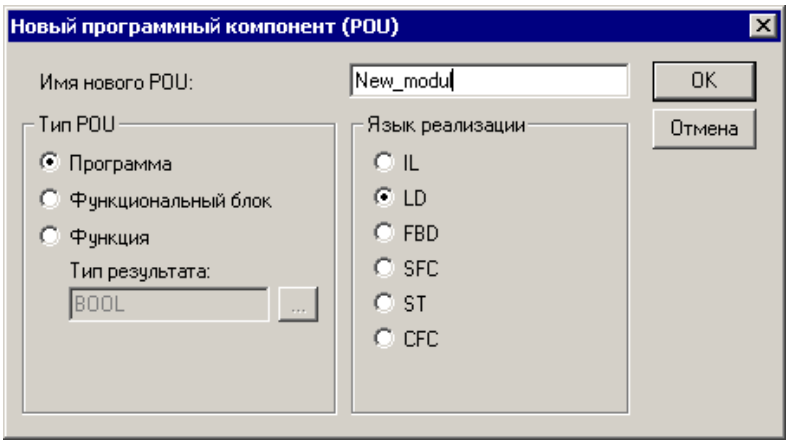


Рисунок 4.11 – Окно «Новый программный компонент (NewPOU)»

3) Написать программу этого функционального блока, которую предполагается использовать в качестве пользовательского программного модуля (например, см. рисунок 4.12).

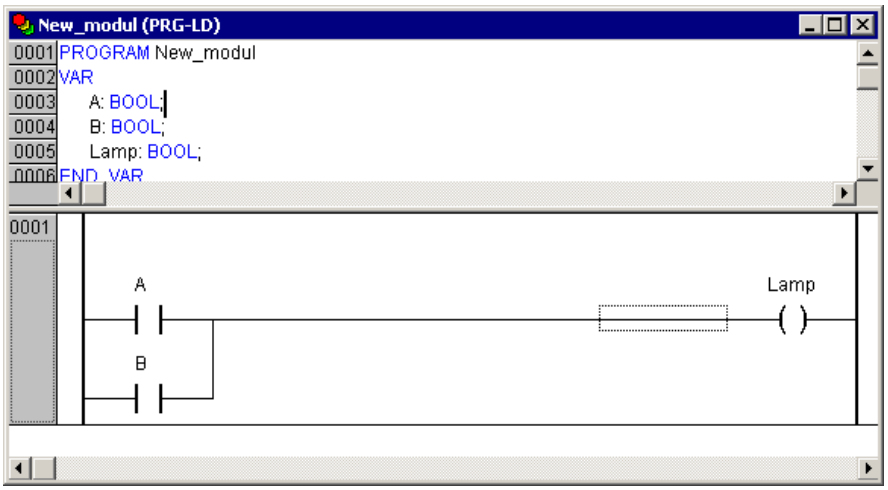


Рисунок 4.12 – Пример текста нового программного компонента

4) Удалить из проекта программный компонент PLC_PRG (команда «Удалить объект» контекстного меню объекта в дереве объектов), оставив в нем только требуемый функциональный блок.
5) Сохранить POU командой «Файл | Сохранить как», задав в открывшемся окне, в поле «Тип файла» тип файла – «Внешняя библиотека (*.lib)» и нажав кнопку «Сохранить».
6) Сохраненный таким образом функциональный блок следует затем подключить к разрабатываемому проекту аналогично тому, как подключаются готовые программные модули (см. раздел 4.2.4)
7) После этого новый функциональный блок отобразится в перечне доступных стандартных функциональных блоков окна «Ассистент ввода» и может быть добавлен в текущий проект аналогично тому, как это выполняется для функциональных блоков из состава поставляемых библиотек (см. раздел 4.2.4), см. рисунок 4.13.

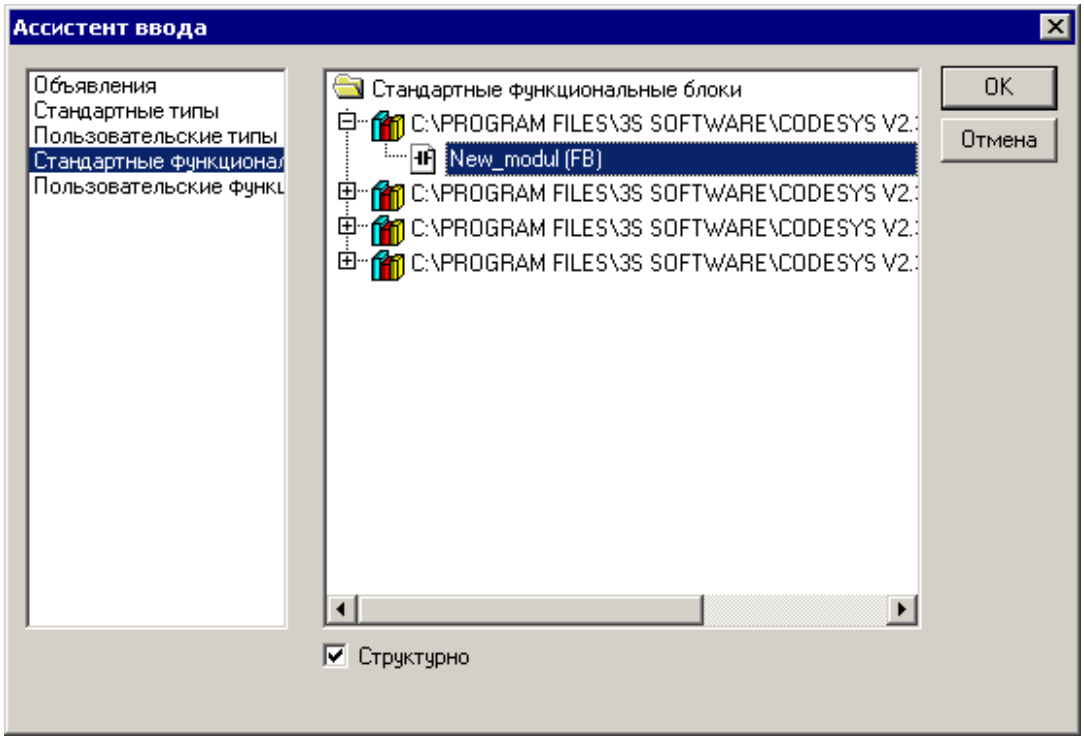


Рисунок 4.13

4.3 Многозадачность

По умолчанию в проекте всегда создается единственная «главная» программа PLC_PRG, выполняемая циклически (см. раздел 3.3.3).

Кроме того, в проекте можно явно определить несколько **задач** с различными условиями выполнения. Задача – это единица обработки программы.

Каждая задача имеет название, приоритет и тип:

- Название служит идентификатором задачи.
- Тип определяет условие вызова задачи. Условием может служить время (циклическое или свободное freewheeling выполнение) или событие, внутреннее или внешнее (например, превышение заданного порога глобальной переменной или прерывание в контроллере).
- Приоритет задается числом (от 1 до 15) и в сочетании заданными условиями вызова задачи определяет хронологический порядок выполнения задач.

Для каждой задачи назначается ряд программ, которые будут в ней выполняться. Если задача выполняется в текущем цикле, то это означает, что выполняются включенные в неё программы (по одному циклу каждая).

Порядок выполнения задач определяется комбинацией приоритетов и условий вызова задач.

Кроме того, для каждой задачи можно задать контроль времени выполнения («сторожевой таймер»). Возможности его использования и настройки определяются целевой платформой.

Выполнение каждой задачи можно разрешить или запретить независимо от других.

4.3.1 Конфигурирование задач

Определение задач в рамках редактируемого проекта производится в окне «Конфигурация задач (Task Configuration)», открывающемся при выборе строки «Конфигурация задач (Task Configuration)» в дереве объектов («ресурсов») на вкладке Ресурсы (Resources) Организатора объектов.

Окно «Конфигурация задач» (см. рисунок 4.14) разделено на две части.

В левой части окна отображается перечень задач текущего проекта в виде дерева конфигурации. В корневой позиции обязательно присутствует элемент «Конфигурация задач (Task Configuration)». Под ним раскрывается список конкретных задач, представленных по именам.

Для добавления в проект новой задачи следует выбрать команду «Вставка | Вставить задачу» главного меню или команду «Вставить задачу» контекстного меню дерева задач.

При добавлении задачи в проект она включается в дерево задач и снабжается пиктограммой, отображающей тип задачи:

- – задачи, выполняемые по системным событиям (Старт, Стоп, Сброс).
- – циклически выполняемые задачи;
- – задачи, выполняемые по времени (свободному);
- – задачи, выполняемые по событию (связанному с глобальными переменными проекта);
- – задачи, выполняемые по внешнему событию.

В правой части окна отображаются поля задания атрибутов текущей (выбранной в дереве задач) задачи. Набор полей соответствует атрибутам задачи выбранного типа.

Задавая значения атрибутов, можно конфигурировать свойства задач (Task properties), вызова программ (Program call), задавать связи с системными событиями (System events). Эта возможность зависит от выбора целевой платформы. Она должна быть поддержана в системе исполнения и разрешена в опциях целевой системы. Если стандартный набор настроек расширен специфическими параметрами, они будут представлены на отдельной вкладке «Parameter» в правой части окна.

Внимание!Если в Конфигураторе задач (Task Configuration) определена последовательность выполнения задач, то проект может не содержать PLC_PRG.
Но если ‘Конфигурация задач (Task Configuration) не используется в проекте, то удалять или переименовывать POU PLC_PRG нельзя: PLC_PRG является главной программой в однозадачном проекте.

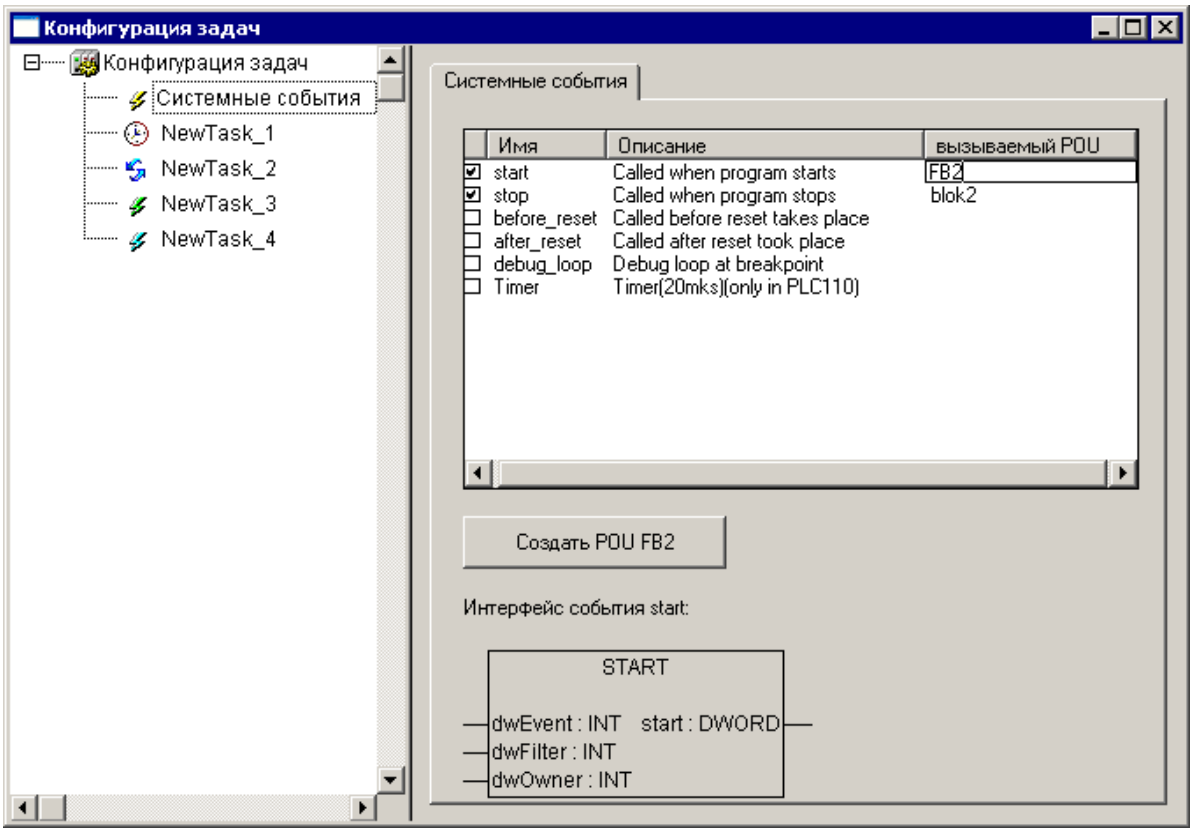


Рисунок 4.14 – Окно «Конфигурация задач (Task Configuration)». Конфигурирование задач типа «Системные события»

Конфигурирование задач типа «Системные события» (то есть выполняемых по одному из возможных системных событий включает (см. рисунок 4.14):

- Указание требуемого события – установкой флажка в поле переключателя в требуемой строке списка системных событий.
- Указание имени программного компонента (POU), который должен выполняться по наступлению события (ввод с клавиатуры в столбце «Вызываемый POU» в требуемой строке списка системных событий). Если POU с указанным именем не существует в текущем проекте, то активируется кнопка «Создать POU <Имя POU>». По нажатию этой кнопки автоматически формируется программный компонент проекта, имеющий указанное имя. Он может редактироваться так же, как другие POU.

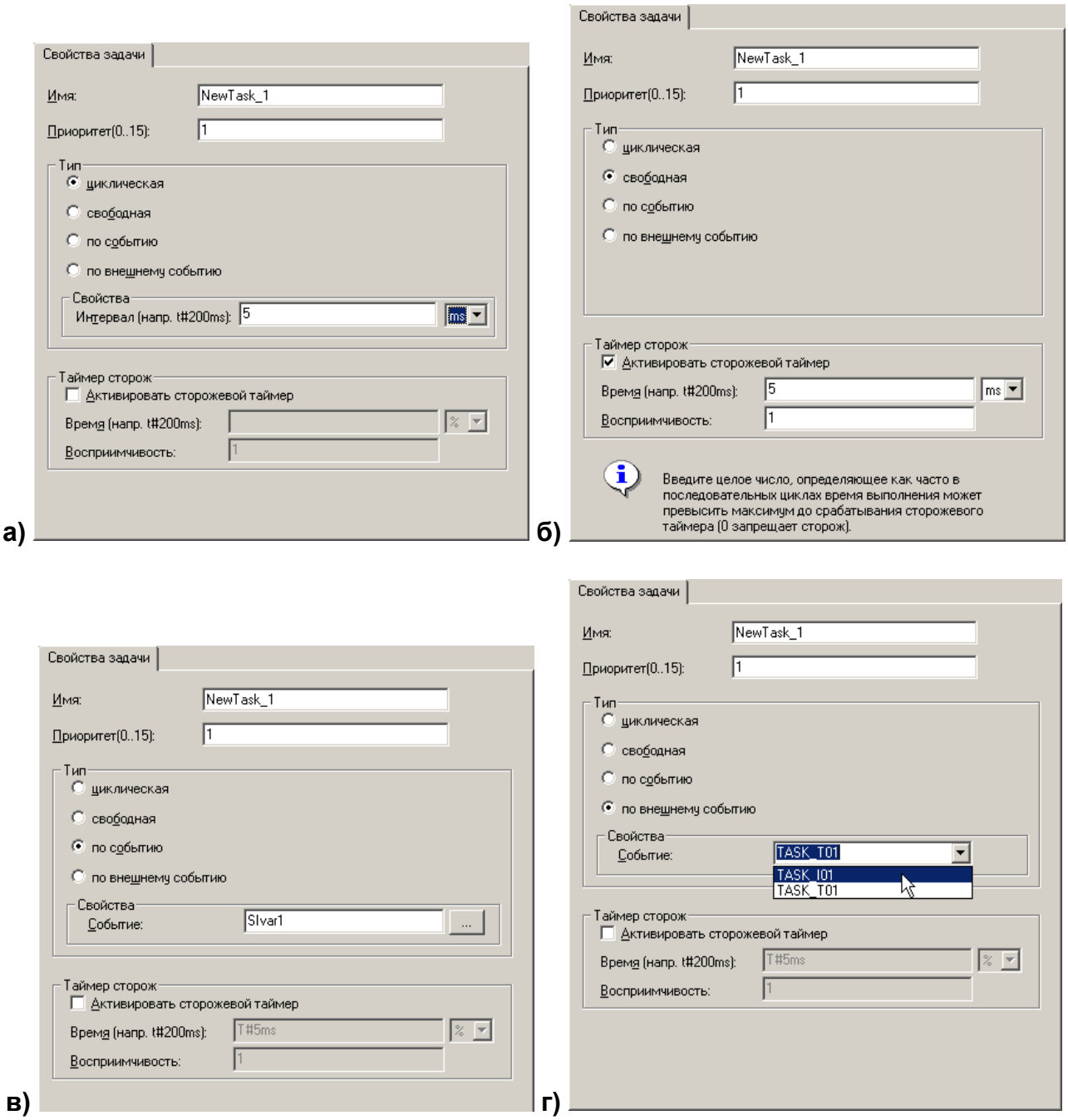


Рисунок 4.15 – Конфигурирование задач

Конфигурирование задач других типов включает (см. рисунок 4.15):

- Указание типа задачи («Циклическая / Свободная / По событию / По внешнему событию).
- Для задачи циклического типа – указание интервала выполнения (см. рисунок 4.15, а);
- Для задачи, выполняемой по событию – указание события (см. рисунок 4.15 в) – по нажатию кнопки у правого края поля открывается окно «Ассистент ввода», в котором можно выбрать требуемую переменную
- Для задачи, выполняемой по внешнему событию – указание события (см. рисунок 4.15, г) – по нажатию кнопки у правого края поля открывается список задач, в котором можно выбрать требуемую задачу.

– Для задач любого типа задать параметры контроля времени выполнения («сторожевого таймера») – если эта опция доступна в используемом ПЛК (см. рисунок 4.15, б).

***Примечание**. Не следует использовать одни и те же строковые функции в разных задачах, это может привести к ошибкам перезаписи данных.*

В режиме «Онлайн» выполнение задач можно наблюдать в виде графической диаграммы.

4.3.2 Обработка событий

1. Системные события, которые контроллер, запрограммированный посредством CODESYS, способен обрабатывать, перечислены в таблице на рисунке 4.14:Событие “start” – вызов POU (функции) если программа начала выполняться (после загрузки в ОЗУ автоматически, либо по команде пользователя);функция выполняется до начала выполнения первого цикла программы.
2. Событие “stop” – вызов функции если программа была остановлена. Функция выполняется сразу после получения системой команды остановки программы (из отладочной среды или с помощью управляющего тумблера на передней панели).
3. Событие “before_reset” – вызов функции если был произведен горячий рестарт системы с помощью трехпозиционного тумблера (тумблер удерживался пять и более секунд в положении «Сброс»). Функция выполняется до перезагрузки контроллера.
4. Событие “after_reset” – вызов функции если был произведен горячий рестарт системы с помощью трехпозиционного тумблера (тумблер удерживался пять и более секунд в положении «Сброс»). Функция выполняется после перезагрузки контроллера.
5. Событие “debug_loop” – вызов функции если включен режим отладки и выполнение программы дошло до отладочной точки останова.
6. Событие “Timer” – вызов функции каждые 20 микросекунд.

Функции обработки событий должны иметь такие параметры, как указано в закладке «Системные события» окна «Конфигурация задач». В случае с контроллером ПЛК-110 функции обработки событий имеют одинаковый набор параметров: входными параметрами являются dwEvent, dwFilter и dwOwner типа INT, при этом функция выдает значение типа WORD. (см. Помощь по CODESYS – «Система программирования CODESYS» – «Ресурсы» – «Конфигуратор задач» – «Системные события»)

7. Отладка проекта

ПО CODESYS располагает несколькими инструментами отладки проекта. Их краткие описания приведены ниже.

4.3.3 Цифровая трассировка[4] (Sampling Trace)

«Цифровая трассировка» (Sampling Trace) позволяет отображать последовательные значения переменных, записанные периодически, с возможной привязкой к событию, заданному так называемым триггером трассировки (см. рисунок 4.16). Цифровая трассировка обладает способностью трассировать до 20 переменных одновременно. Для каждой переменной сохраняются 500 последних значений.

[4] Подробнее о цифровой трассировке можно узнать из встроенной помощи к системеCODESYS: запустить CODESYS, затем вызвать справку (F1 или меню – «Справка» – «Содержание…»), затем выбрать в содержании слева «Система программирования CODESYS» – «Трассировка» – «Трассировка» – «Трассировка в CODESYS».

Настройка, выполнение и просмотр результатов трассировки производится в режиме, вызываемом выбором строки «Трассировка» дерева ресурсов проекта на вкладке «Ресурсы» организатора объектов CODESYS.

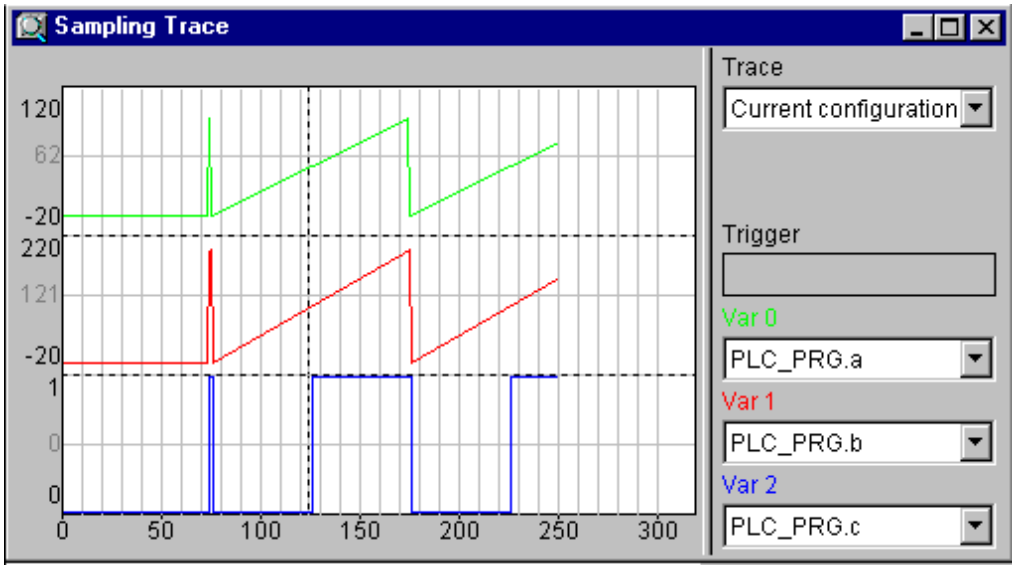


Рисунок 4.16 – Окно трассировки

4.3.4 Отладка

Опция отладки ПО CODESYS заставляет компилятор формировать дополнительный код, упрощающий поиск ошибок. Опция «Отладочный код (Debugging)» включается установкой флажка переключателя «Отладочный код (Debugging)» в окне «Опции (Options)», вызываемом командой «Проект | Опции (Project | Options)» главного меню, на вкладке «Генератор кода (Build)» (см. рисунок 4.17).

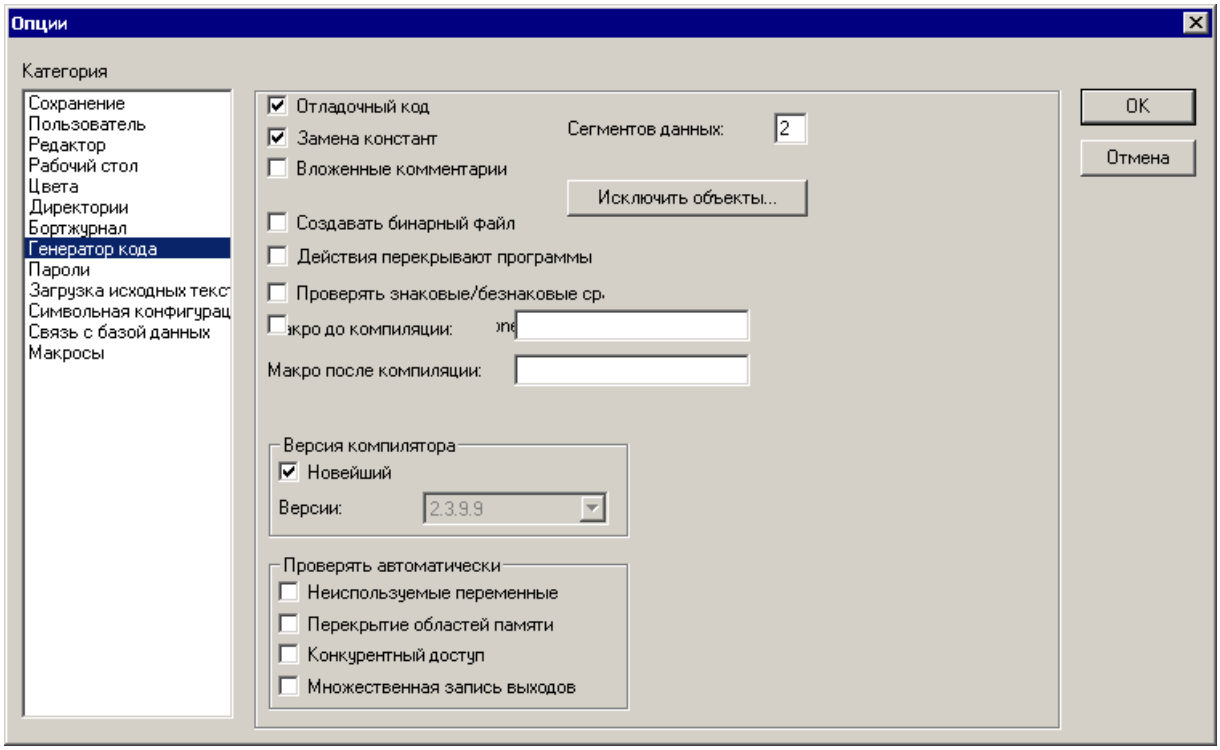


Рисунок 4.17 – Окно «Опции (Options)», вкладка «Генератор кода»

4.3.5 Точки останова

Точки останова – это места, в которых выполнение программы будет приостанавливаться, что позволяет просмотреть значения переменных на определенном этапе работы программы. Точки останова можно задавать во всех редакторах. В текстовом редакторе точка останова устанавливается на номер строки, в FBD и LD – на графический элемент, в SFC – на шаг.

Внимание! Система исполнения CODESYS SP32 Bit Full автоматически деактивирует сторожевой таймер задачи, если она выходит на точку останова.

4.3.6 Пошаговое выполнение[5]

Пошаговое выполнение позволяет проверить логическую правильность программы.

Под «шагом» подразумевается:

- В IL: Выполнить программу до следующего оператора CALL, LD или JMP.
- В ST: Выполнить следующую инструкцию.
- В FBD, LD: Выполнить следующую цепь.
- В SFC: Продолжить действие до следующего шага.

4.3.7 Выполнение по циклам

Команда «Онлайн | Один цикл (Online | Single Cycle)» выполняет один рабочий цикл и останавливает контроллер после выполнения.

4.3.8 Эмуляция

Режим эмуляции последовательно включается и отключается выбором команды «Онлайн | Режим эмуляции (Online | Emulation)» главного мню. Включенный режим маркируется установленным флажком в строке главного меню и записью «Эмул.» в строке состояния главного окна.

Во время эмуляции созданная программа выполняется не в ПЛК, а в компьютере, на котором запущено ПО CODESYS. В этом режиме допустимы все функции онлайн, что позволяет проверить логическую правильность программ, не используя контроллер.

Внимание!
В режиме эмуляции функции внешних библиотек не выполняются.

[5] Подробнее о пошаговом выполнении можно узнать из встроенной помощи к системе CODESYS: запустить CODESYS, затем вызвать справку (F1 или меню – «Справка» – «Содержание…»), затем выбрать в содержании слева «Система программирования CODESYS» – «Что есть что в CODESYS» – «Отладка и Online-функции».

4.3.9 Бортжурнал (Log)

«Бортжурнал (Log)» хронологически записывает действия пользователя, внутренние сообщения системы исполнения, изменения состояния и исключения в режиме онлайн. Это позволяет анализировать условия возникновения ошибки при отладке программы.

Просмотр записей «Бортжурнала (Log)» производится в режиме, вызываемом выбором строки «Бортжурнал (Log)» дерева ресурсов проекта на вкладке «Ресурсы» организатора объектов CODESYS.

5 Использование сложных структур данных

Кроме стандартных типов данных (см. раздел 4.2.3.1), в проектах можно использовать определяемые пользователем сложные типы данных (массивы, перечисления, структуры и некоторые другие): объекты (переменные или постоянные) которые имеют внутреннюю структуру, доступную программисту. Их использование позволяет произвольно конструировать требуемые структуры данных из небольшого набора предопределённых типов.

Чем адекватнее используемая в программе структура данных реальному объекту автоматизации, тем безошибочнее и долговечнее будет функционировать разработанная программа.

5.1 Пользовательские типы данных

5.1.1 Массивы

Элементарные типы данных могут образовывать одно-, двух-, и трехмерные массивы. Массивы могут быть объявлены в разделе объявлений POU или в списке глобальных переменных.

Путем вложения массивов можно получить многомерные массивы, но не более 9-мерных ("ARRAY[0..2] OF ARRAY[0..3] OF …").

Синтаксис:

<Имя_массива>:ARRAY [<ll1>..<ul1>,<ll2>..<ul2>] OF <базовый тип>

Здесь ll1, ll2, ll3 указывают нижний предел индексов; ul1, ul2 и ul3 указывают верхние пределы. Индексы должны быть целого типа. Нельзя использовать отрицательные индексы.

5.1.2 Перечисления

Перечисление – это определяемый пользователем тип данных, задающий несколько строковых псевдонимов для числовых констант.

Перечисление доступно в любой части проекта, даже при локальном его объявлении внутри POU. Поэтому рационально создавать все перечисления на вкладке «Типы данных» Организатора Объектов.

Объявление должно начинаться с ключевого слова TYPE и заканчиваться строкой END_TYPE.

Синтаксис:

TYPE <Имя_перечисления>:(<Элемент_0> ,< Элемент _1>, ...
…, < Элемент_n>); END_TYPE

Переменная типа <Имя_перечисления> может принимать только перечисленные значения. При инициализации переменная получает первое значение из заданного списка. Если числовые значения элементов перечисления не указаны явно, то им присваиваются последовательно возрастающие числа, начиная с 0. Фактически элемент перечисления – это число типа INT, и работать с ними можно точно так же. Можно напрямую присвоить число переменной типа перечисление.

Элемент, уже включенный в перечисление, нельзя повторно включать в другое перечисление.

5.1.3 Структуры

Структуры создаются командой «Добавить объект (AddObject)» контекстного меню вкладки «Типы данных» Организатора Объектов.

Новый объект отображается в дереве объектов, окно задания параметров объекта открывается в рабочей области главного окна ПО CODESYS (см. рисунок 5.1).

Объявление должно начинаться с ключевых слов TYPE и STRUCT и заканчиваться строками END_STRUCT и END_TYPE.

Синтаксис:

TYPE <Имя _структуры>:
STRUCT
<Объявление переменной 1>
.
.
<Объявление переменной n>
END_STRUCT
END_TYPE

<Имя _структуры> образует новый тип данных, который может быть использован в любой части проекта наряду с базовыми типами.

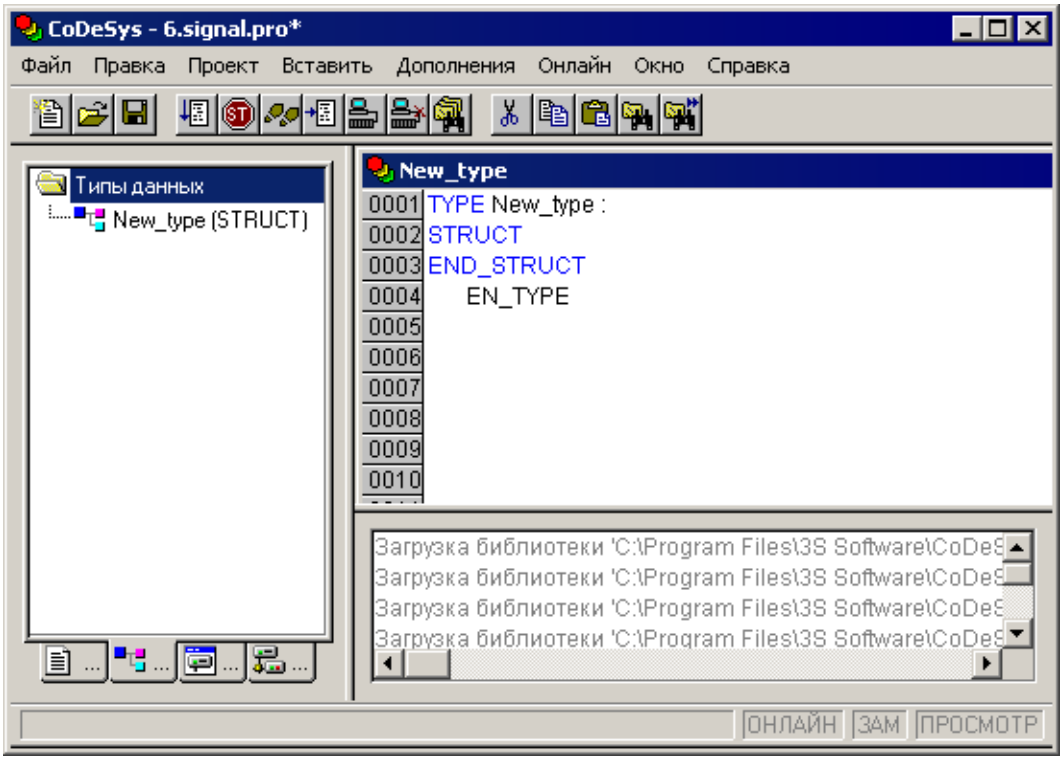


Рисунок 5.1 – Вкладка «Типы данных»

Допускаются вложенные структуры, но запрещено размещение элементов структуры по прямым адресам (в частности, недопустимы **AT** объявления).

Для доступа к элементам структуры используется следующий синтаксис:

<Имя_структуры>.<Имя_компонента>

Например, если структура "Week" содержит компонент "Monday", то обращение к нему будет выглядеть так: **Week.Monday.**

5.1.4 Указатели

Указатели позволяют работать с адресами переменных или функциональных блоков.

Синтаксис:

<Имя_указателя>: POINTER TO <Тип данных/Функциональный блок>;

Указатели применимы для всех базовых типов данных или функциональных блоков, включая определяемые пользователем.

6 Визуализация проекта

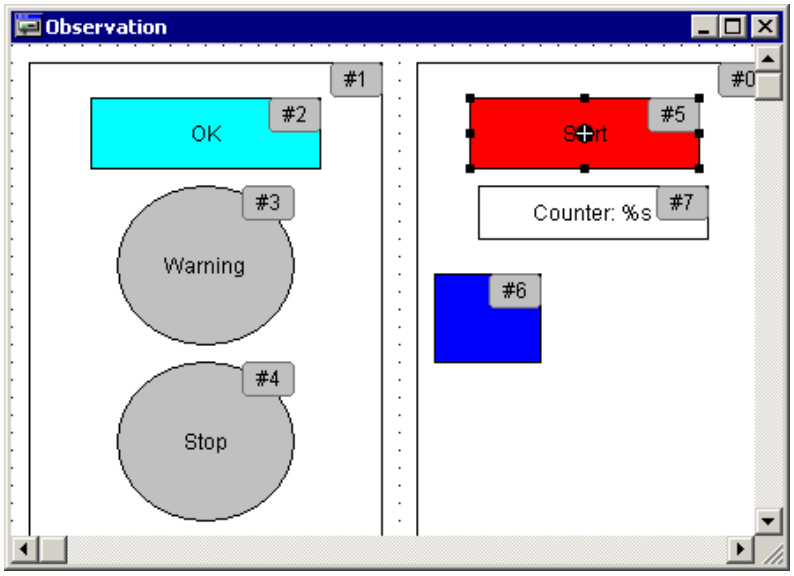
Визуализация проекта предназначена для графического представления объекта управления и непосредственно связана с созданной в ПО CODESYS программой контроллера. Редактор визуализации CODESYS предоставляет набор готовых графических элементов, которые могут быть связаны требуемым образом с переменными проекта.

Например, если в программе доступна переменная, связанная с уровнем заполнения некоторой емкости, то в визуализации ее можно изобразить графическим элементом в виде полосы, которая, в зависимости от значения переменной проекта, будет изменять свою длину и/или цвет.

В Online режиме представление элементов визуализации на экране изменяется в зависимости от значений переменных.

Свойства отдельных элементов визуализации, а также визуализации в целом устанавливаются в соответствующих диалогах конфигурации и диалоге свойств объекта (см. рисунок 6.1, а, б). Здесь определяется начальный вид элементов и выполняется привязка динамических свойств к значениям переменных проекта.

а)



б)

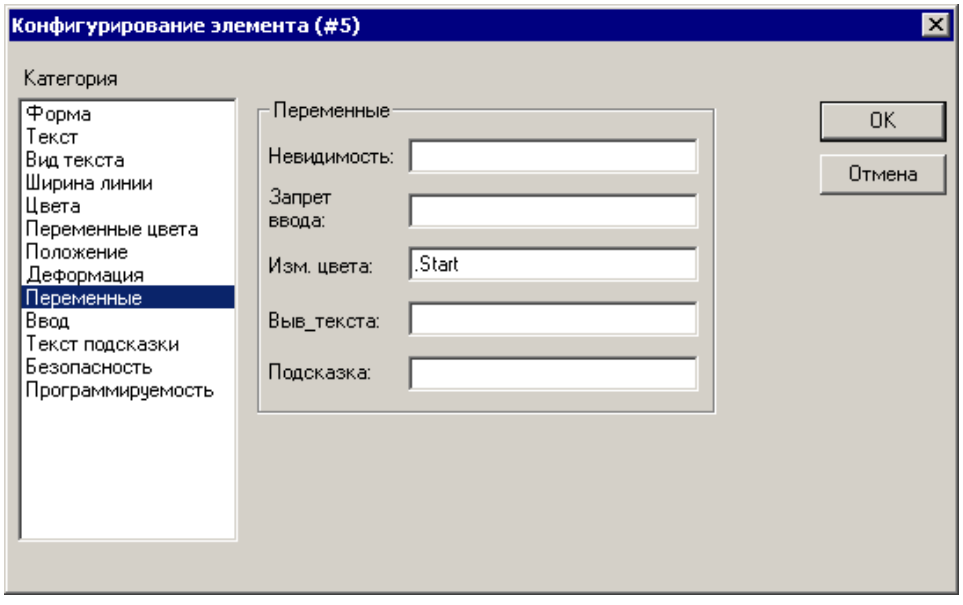


Рисунок 6.1 – Окно визуализации проекта (а) и окно конфигурирования элемента визуализации (б)

Визуализация проекта может использоваться как пользовательский интерфейс для контроля и управления работой ПЛК программы в рабочем режиме. В определенных ситуациях – при необходимости, например, исключить возможность вмешательства оператора в программу работы – как единственный пользовательский интерфейс.

В этом случае ввод данных для программы должен выполняться исключительно посредством элементов визуализации. Такую возможность обеспечивают специальные возможности ввода, задаваемые в процессе конфигурации. Кроме того, предусмотрено создание клавиш быстрого ввода для каждой конкретной визуализации.

Созданная в CODESYS визуализация может использоваться еще несколькими способами:

- Программа Win32 CODESYS HMI отображает формы визуализации на ПК в полноэкранном режиме. В отличие от ПО CODESYS, эта программа не бесплатна.
- Web-визуализация отображает данные и предоставляет возможность удаленного управления через Интернет.
- Для контроллеров со встроенным дисплеем доступна целевая визуализация.

Подробнее о создании окон визуализации см. документ «Визуализация CODESYS. Дополнение к руководству пользователя по программированию ПЛК в CODESYS 2.3».

6.1 CODESYS HMI

CODESYS HMI – это система исполнения визуализаций созданных в среде программирования CODESYS.

Если проект содержит визуализацию, то при запуске CODESYS HMI она будет воспроизводиться в полноэкранном режиме. Пользователь сможет использовать заданные в программе функции управления и отображения при помощи мыши и клавиатуры, причем, даже если проект CODESYS защищен от чтения.

Возможность редактирования программ, меню и панели инструментов CODESYS не доступны пользователю. Поэтому все необходимые функции управления и отображения данных должны быть сопоставлены соответствующим элементам визуализации. Для этого в диалоге конфигурации элементов визуализации предусмотрены специальные возможности ввода для CODESYS HMI.

6.2 Web визуализация

Web визуализация – это технология, позволяющая наблюдать и управлять CODESYS визуализацией посредством Web-браузера на любой аппаратной платформе.

CODESYS может формировать описания объектов визуализации проекта в формате XML и загружать их в контроллер. Web-сервер обрабатывает данные контроллера и также в формате XML создает постоянно обновляемую визуализацию. Таким образом, она будет отображаться в Web-браузере на любом подключенном через Интернет компьютере независимо от платформы (например, с целью удаленного управления).

Web-сервер может динамически переключаться между несколькими контролерами. Для этого PLCHandler будет использоваться как базовый компонент визуализации. Элементы визуализации могут быть сконфигурированы для переключения целевой системы.

7 Конфигурирование контроллера

7.1 Конфигурация памяти ввода / вывода

В процессе создания и отладки проекта необходимо настроить конфигурацию входов, выходов и интерфейсов связи ПЛК с внешними модулями ввода-вывода, устройствами индикации или иными устройствами, обмен данными с которыми будет производиться по сети (см. раздел 3.4).

Внешние устройства обмениваются данными с пользовательской программой ПЛК через специальную область памяти ПЛК: область памяти ввода / вывода ПЛК (**%I** и **%Q**). Она включает дискретные и аналоговые входы и выходы, модули расширения функционала (в том числе организующие обмен информацией между ПЛК и отдельными приборами и устройствами, связанными по сети с ПЛК).

Размер памяти ввода / вывода определяется типом лицензии CODESYS контроллера ПЛК (см. раздел 2.2).

Настройка конфигурации выполняется в окне редактора «Конфигурация ПЛК (PLC Configuration)» ПО CODESYS.

Для входа в режим редактирования конфигурации ПЛК следует перейти на вкладку «Ресурсы» Организатора объектов. В «дереве ресурсов» следует выбрать пункт «Конфигурация ПЛК (PLC Configuration)». В рабочей области главного окна откроется окно редактора (см. рисунок 7.1).

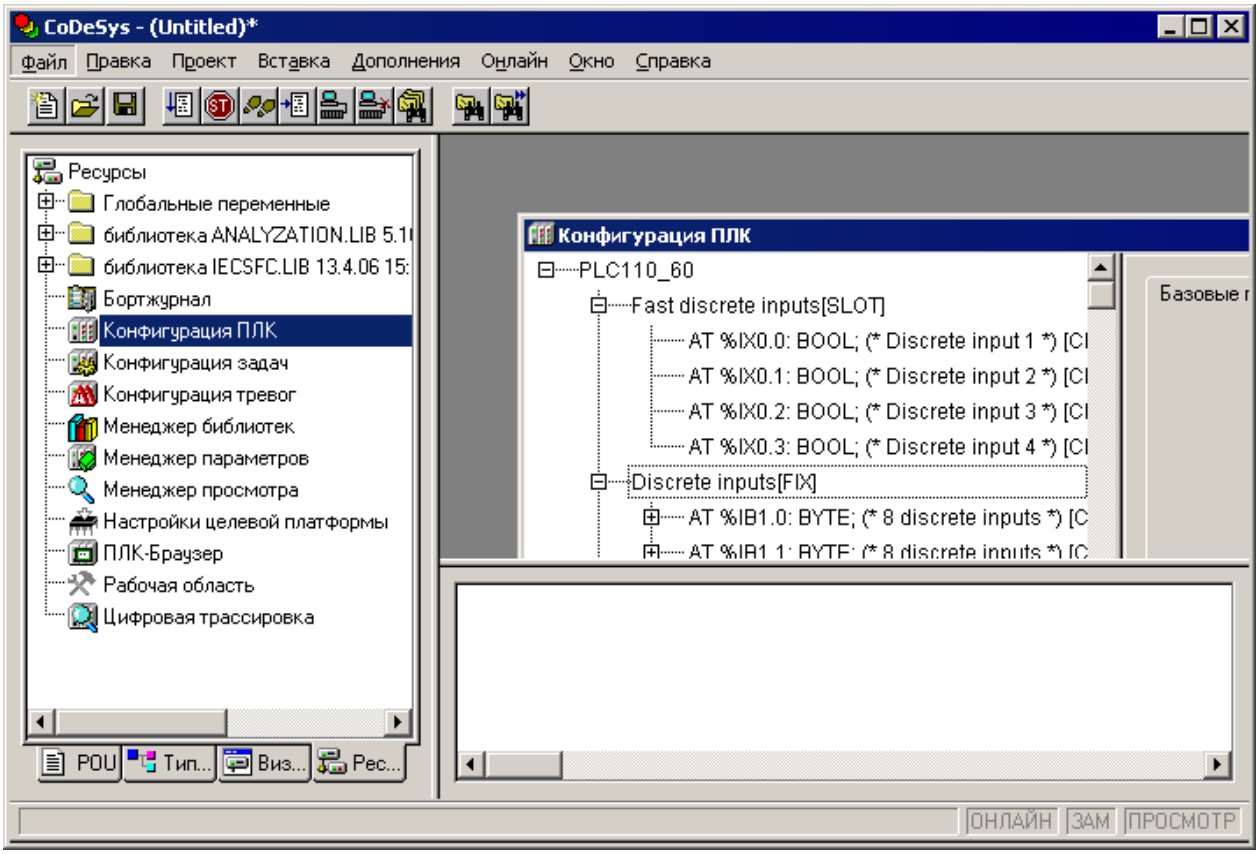


Рисунок 7.1 – Вход в режим «Конфигурация ПЛК (PLC Configuration)»

Окно редактора конфигуратора ПЛК разделено на две части. В левой части окна отображается дерево конфигурации, древовидная структура, отображающая ресурсы контроллера. Структура и компоненты дерева определяются файлом настроек целевой платформы (см. раздел 3.2) конфигурации, но могут быть изменены пользователем CODESYS. В правой части окна отображаются диалоги конфи-

гурации, доступные для текущего (выделенного) элемента дерева конфигурации. Диалоги отображаются в виде одной или нескольких табличных вкладок (см. рисунок 7.2). В полях, расположенных на вкладках диалогов, задаются требуемые значения параметров канала или модуля. Значение параметра устанавливается интерактивно до компиляции проекта. Оно передается в ПЛК и влияет на работу аппаратуры.

Примечание. *Правая часть окна видна по умолчанию, но может быть скрыта выбором команды меню «Дополнения (Extras) | Свойства (Properties)» главного меню: последовательные щелчки левой кнопкой мыши на строке команды включают (в строке при этом отображается «галочка») и отключают («галочка» отсутствует) отображение.*

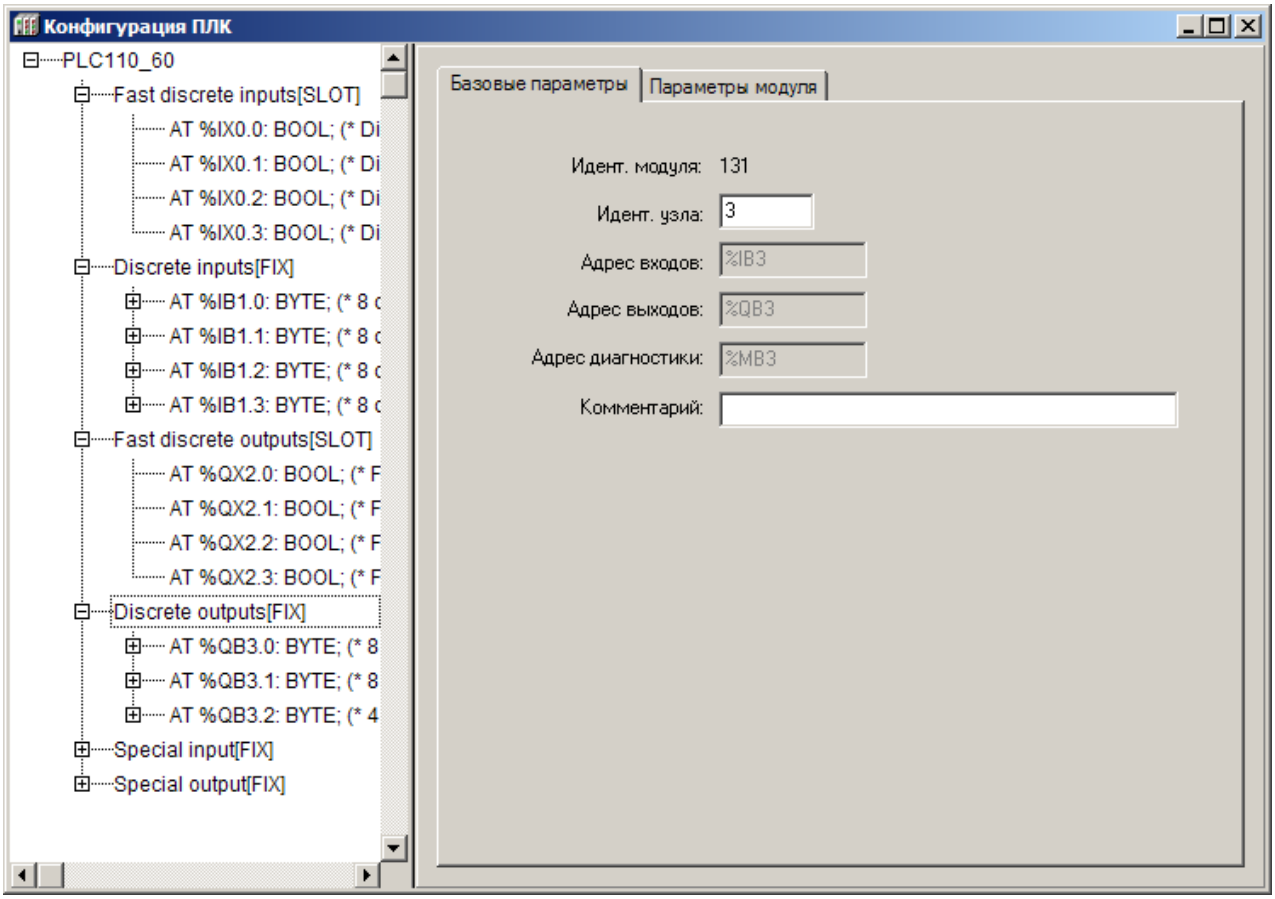


Рисунок 7.2 – Окно режима «Конфигурация ПЛК (PLC Configuration)»

Корневой элемент конфигурационного дерева определяется используемым файлом настроек целевой платформы. Если проект создается без установки настроек целевой платформы, или если в процессе создания проекта файл настроек целевой платформы был заменен другим (т.е. был совершен переход на другую платформу), то вместо отображения дерева конфигурации может отобразиться запись «Not found (Не найдено)». В этом случае следует выбрать команду «Дополнения | Стандартная конфигурация (Extras | StandartConfiguration)» главного меню, и в окне отобразится дерево конфигурации, соответствующее текущему файлу настроек целевой платформы.

Конфигурация ПЛК определяет аппаратные средства системы. В дереве конфигурации задается распределение адресов входов / выходов контроллера, что определяет привязку проекта к аппаратным средствам. На основе описания конфигурации ПЛК CODESYS проверяет правильность задания МЭК адресов, используе-

мых в программах, на их соответствие фактически имеющимся аппаратным средствам.

В дереве конфигурации отображаются следующие элементы:

- **Модуль** (элемент конфигурации)**:** независимая единица аппаратных средств. Модуль включает набор **каналов** ввода-вывода. Модуль (как и каждый отдельный канал) может иметь параметры. Каждый тип модуля имеет уникальный идентификатор. Могут иметь вложенные **подмодули** (подэлементы конфигурации).
- **Канал:** это собственно данные ввода-вывода. Как правило, модуль имеет фиксированный набор каналов или подмодулей. Каждый канал имеет определенный МЭК тип и адрес. Для каждого канала автоматически выделяется определенное пространство памяти. Каждый канал имеет уникальный в пределах данной конфигурации ПЛК идентификатор.
- **Битовый канал:** идентификатор отдельного бита в многобитном канале.

В конфигурации присутствуют модули, отвечающие за структурирование областей ввода и/или вывода, каждый из которых может содержать вложенные подэлементы (субмодули и каналы). Для каналов могут быть назначены символические имена. Прямые МЭК адреса отображаются в конфигурации для каждого символического имени.

Определение адресов каналов в области ввода / вывода ПЛК рекомендуется выполнять в автоматическом режиме. Для этого следует установить флажок переключателя «Автоматическое вычисление адресов (Automatic calculation of addresses)» на вкладке «Настройки (Settings)». В этом случае при изменении положения модуля адреса его каналов соответствующим образом смещаются. Альтернативой может служить фиксированная адресация. В этом случае для каждого модуля отводится фиксированное адресное окно, которое определяется физическим расположением (номером слота) модуля. Например: %QB0, %IB26, %MW4. Подробнее см. раздел «Конфигуратор ПЛК (PLC Configuration)» документа «Руководство пользователя по программированию ПЛК в CODESYS 2.3».

Некоторые элементы конфигурации пользователь настраивает самостоятельно. Настройка может заключаться в добавлении и/или удалении модулей и подмодулей, а также в задании требуемых значений параметрам элементов конфигурации.

Внимание!

1) Добавление и удаление модулей конфигурации, а также настройка их параметров осуществляются при контроллере, отключенном от ПО CODESYS. Для отключения контроллера следует вызвать команду «Онлайн | Отключение (Online | Logout)» главного меню или нажать кнопку «Отключение (Logout)» () панели инструментов.

2) При конфигурировании ПЛК следует иметь в виду, что можно изменять значения только переменных, лежащих в области вывода. Значения переменных из области ввода можно только считывать.

Если в процессе создания программы требуется изменить используемый ПЛК (сменить настройки целевой платформы), то следует:

1) В окне «Настройки целевой платформы» (вкладка «Ресурсы» Организатора объектов) открыть настройки целевой платформы, и выбрать новый Target-файл (соответствующий новому ПЛК).

2) Перейти в режим «Конфигурация ПЛК» и выбрать команду Дополнения | Стандартная конфигурация (Extras | Standard Configuaration)» главного меню.

При этом, если предполагается переход от одного типа контроллера к другому, то переменные следует задавать в режиме («ресурсе») «Глобальные переменные (Global Variables)». Связано это с тем, что при задании стандартной конфигурации («Standard Configuaration») переменные, заданные в редакторе «Конфигурация ПЛК (PLC Configuration)», пропадают, и ранее созданное распределение и именование переменных теряется. При объявлении и глобальных переменных их имена не будут потеряны, и при переходе к другомуTarget-файлу достаточно только скорректировать адреса.

Внимание!
Все переменные, привязанные к каналам конфигурации ПЛК, автоматически объявляются глобальными переменными.

Для объявления глобальной переменной следует:

1) Войти в режим «Глобальные переменные (GlobalVariables)», в открывшемся окне режима (редакторе) – выбрать команду «Автообъявление» контекстного меню (см. рисунок 7.3).

2) В открывшемся окне «Автообъявление переменной» (см. рисунок 7.4) – задать значения параметров вводимой переменной.

Подробнее об объявлении и применении глобальных переменных см раздел 6.2 «Глобальные и конфигурационные переменные, файл комментариев» документа «Руководство пользователя по программированию ПЛК в CODESYS 2.3.

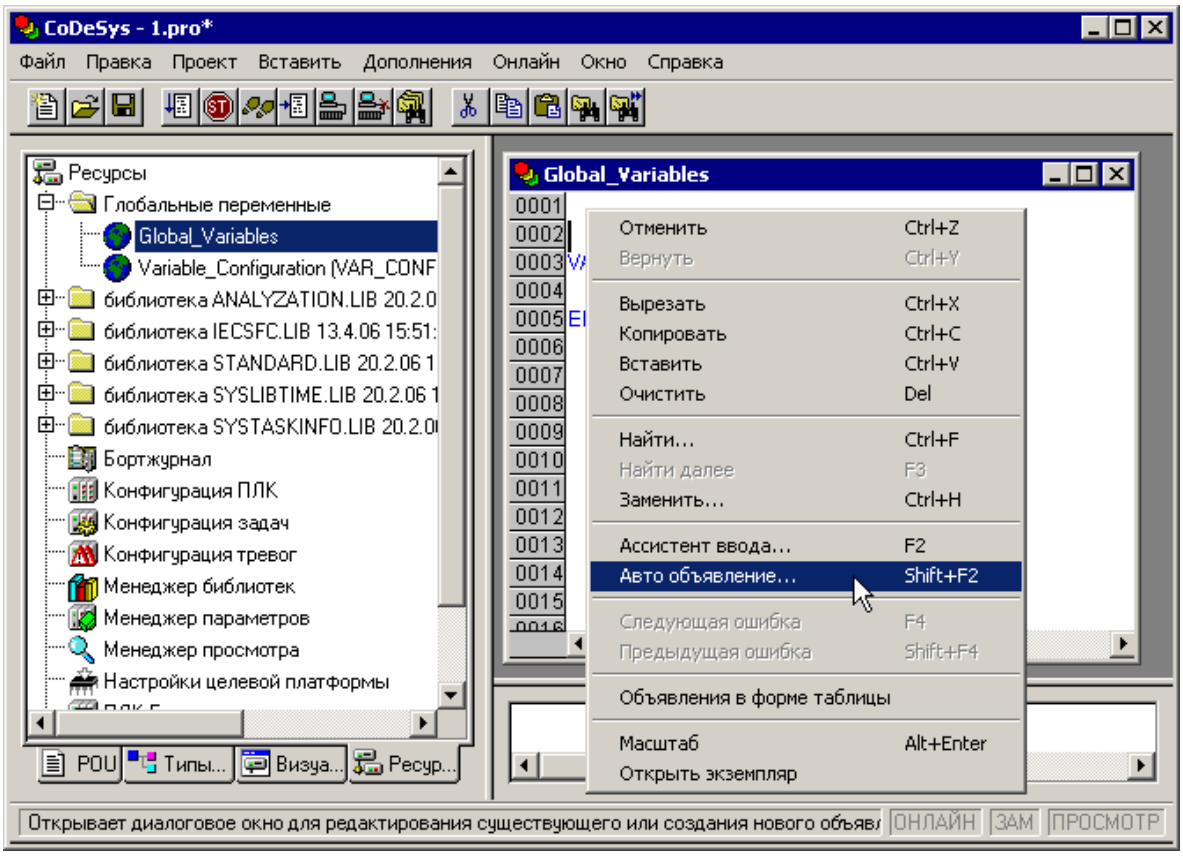


Рисунок 7.3 – Глобальные переменные. Вызов окна «Объявление переменной»

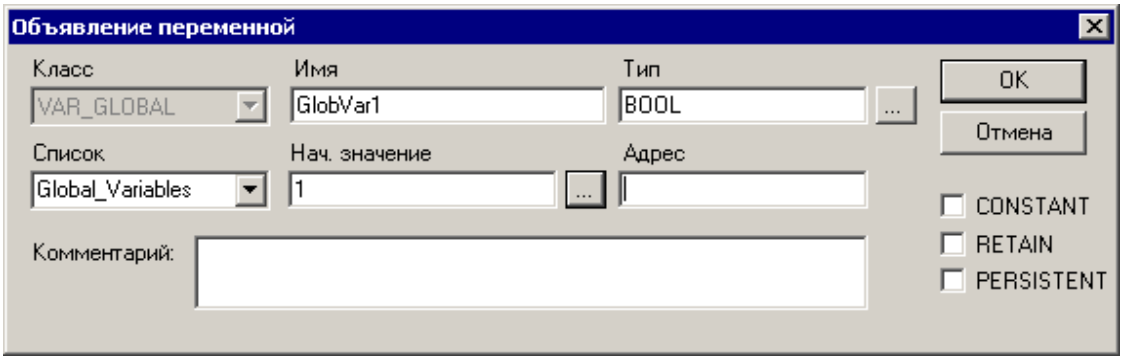


Рисунок 7.4 – Окно «Объявление переменной»

7.1.1 Приемы редактирования конфигурации ПЛК

Начальный вид конфигурации ПЛК задает файл конфигурации (*.cfg) ПЛК, расположенный в директории, определенной в установленном целевом файле (Target-файле) и считываемый при открытии проекта в ПО CODESYS.

Редактирование элементов конфигурации ПЛК заключается в выполнении операций над элементами «дерева» конфигурации ПЛК, отображаемого в левой части окна режима (добавлении, замене и удалении модулей, подмодулей и каналов) и редактировании значений параметров элементов «дерева» конфигурации ПЛК (в правой части окна «Конфигурация ПЛК»).

7.1.1.1 Типы и виды модулей в конфигурации

Существует два вида модулей:

- **Фиксированные модули** – заданы обязательно, и не могут быть удалены или заменены. Доступно только редактирование их параметров.
- **Добавляемые модули** – добавляются (заменяются, удаляются) пользователем в процессе конфигурирования. Подразделяются на два типа:

  Тип«SLOT» – означает, что для модуля зарезервировано место, которое может быть занято или оставлено пустым. На одно зарезервированное место может быть установлен один модуль.

  Тип «VAR» (свободный) – означает возможность установить любое количество модулей (с учетом физических возможностей области ввода / вывода).

7.1.1.2 Добавление подмодулей (подэлементов)

К модулям конфигурации могут быть добавлены подмодули («подэлементы»), которые расширяют функционал или изменяют алгоритм работы модуля.

Чтобы **добавить** подмодуль (подэлемент) в текущую конфигурацию, следует:

1) Либо: Выделить требуемый модуль (элемент) конфигурации и нажатием правой кнопки мыши вызвать контекстное меню.

Выбрать в контекстном меню требуемую команду: «Добавить Подэлемент (Append Subelement) | <Имя Подэлемента>».

Выбранный подэлемент будет добавлен в редактируемую конфигурацию.

2) Либо: Выделить требуемый элемент (модуль) конфигурации и нажатием левой кнопки мыши выбрать команду «Вставка | Добавить Подэлемент (Append Subelement) | <Имя Подэлемента>» главного меню.

Выбранный подэлемент будет добавлен в редактируемую конфигурацию.

7.1.1.3 Замена модулей (элементов)

Чтобы **заменить** модуль (элемент) в текущей конфигурации, следует:

1) Либо: Выделить требуемый модуль (элемент) конфигурации и нажатием правой кнопки мыши вызвать контекстное меню.

Выбрать в контекстном меню требуемую команду: «Заменить Элемент | <Имя элемента>».

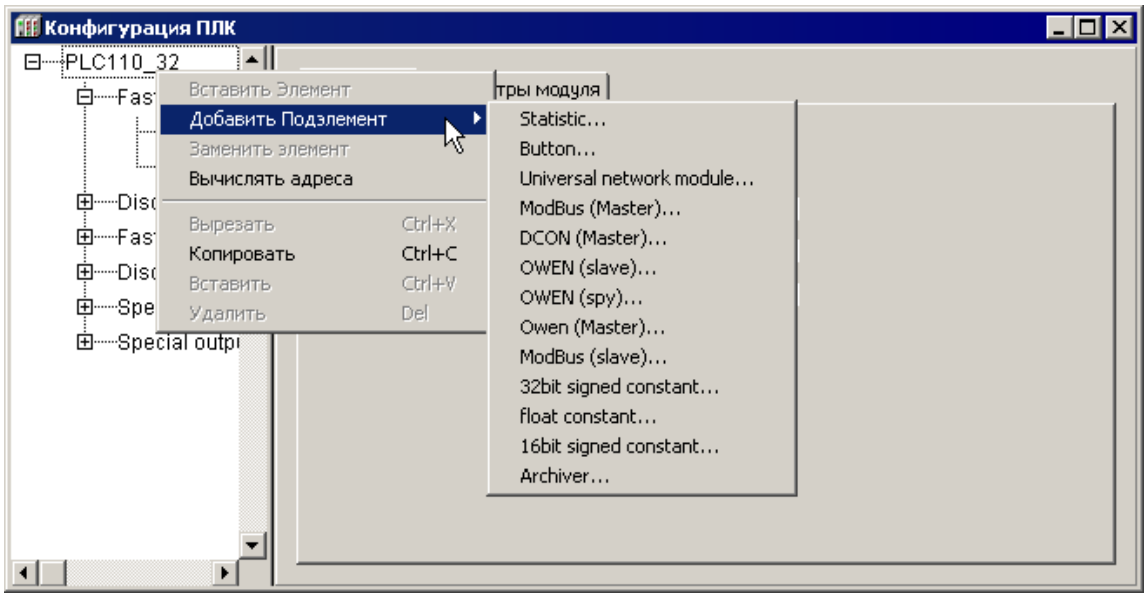
Выбранный элемент заместит собою выделенный модуль (элемент) редактируемой конфигурации.

2) Либо: Выделить требуемый модуль (элемент) конфигурации и нажатием левой кнопки мыши выбрать команду «Дополнения | Заменить элемент | <Имя элемента>» главного меню.

Выбранный элемент заместит собою выделенный модуль (элемент) редактируемой конфигурации.

Процедуры подключения к модулю подчиненного подэлемента и замены элемента продемонстрированы на рисунке 7.5.

а)



б)

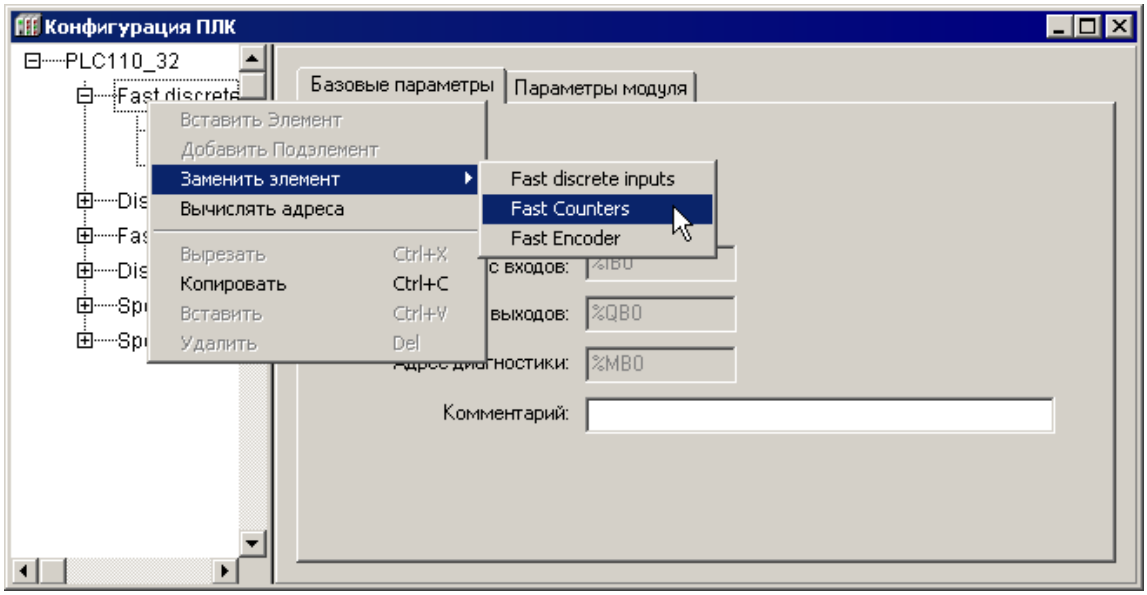


Рисунок 7.5 – Подключение (а) и замена (б) подчиненного подмодуля

7.1.1.4 Удаление подмодулей (подэлементов)

Чтобы **удалить модуль** (элемент) из текущей конфигурации (удалены могут быть только добавляемые модули, фиксированные модули не могут быть удалены из конфигурации), следует:

1) Выделить требуемый модуль в дереве конфигурации и выбрать команду **Удалить (Delete)** контекстного меню дерева конфигурации.

2) В открывшемся окне запроса подтверждения операции – нажать кнопку «Да» для подтверждения операции (или кнопку «Нет» для отказа от завершения операции удаления). Выделенный модуль будет удален из дерева конфигурации.

7.1.1.5 Параметры модулей

Параметры текущего (выделенного в дереве конфигурации) модуля отображаются на вкладках в правой части окна режима «Конфигурация ПЛК (PLC Configuration)».

7.1.1.5.1 Вкладка «Базовые параметры» модуля

В полях вкладки «Базовые параметры (Base parameters)» отображаются значения параметров:

- **«Идент(ификатор) модуля (Modul id)»** – идентификационный номер модуля.
- **«Идент(ификатор) узла (Node id)»** – определяет положение модуля на его уровне иерархии в общей конфигурации –. Это значение можно редактировать, в таком случае аналогичные идентификаторы других модулей одного уровня иерархии будут сдвигаться;
- **«Адрес входов (Input Addres), Адрес выходов (Output Addres), Адрес диагностики (Diagnostic Addres)** – отображаются адреса областей ввода-вывода (приводятся конкретные номера). Они могут использоваться для обращения при программировании; значения недоступны для редактирования.
- **«Комментарий (Comment)»** – произвольный текст комментария.

На рисунке 7.6, на примере модуля дискретных выходов, представлено окно режима «**Конфигурация ПЛК (PLC Configuration)**» с вкладкой базовых параметров модуля в правой части экранной формы.

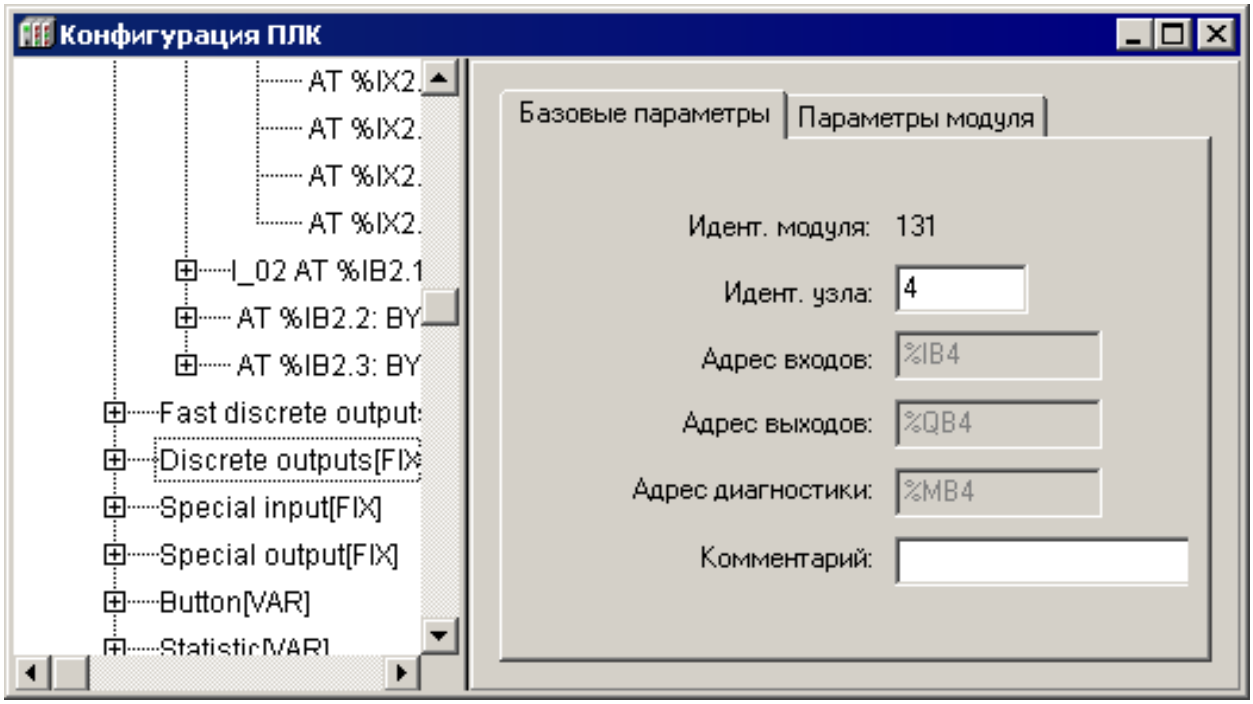


Рисунок 7.6 – Окно режима «Конфигурация ПЛК (PLC Configuration)». Модуль дискретных выходов. Вкладка «Базовые параметры»

7.1.1.5.2 Вкладка «Параметры модуля»

На вкладке «Параметры модуля (Module parameters)» отображаются значения параметров, представленные в виде таблицы, содержащей столбцы:

- индекс (**Index**),
- имя (**Name**),
- значение (текущее) (**Value**),
- (значение) по умолчанию (**Default**),
- минимальная (**Min**) величина диапазона возможных значений,

– максимальная (**Max**) величина диапазона возможных значений.

Примечание. *Значения параметров по умолчанию, минимальные и максимальные значения – опциональные и не всегда присутствуют во вкладках параметров модулей.*

Для редактирования цифровых или символьных значений параметров следует щелкнуть левой кнопкой мыши на требуемом значении, после чего запись переключается в режим редактирования, и ввести требуемое значение параметра с клавиатуры.

Для редактирования значений параметров, которые могут принимать определенное значение из списка значений, следует щелкнуть левой кнопкой мыши кнопку с треугольной стрелкой, отображаемую рядом со значением параметра. Нажатие этой кнопки раскрывает список допустимых значений параметра, в котором следует щелкнуть левой кнопкой мыши на требуемом значении. Оно будет подставлено в перечень параметров. После щелчка левой кнопкой мыши в любой другой области окна выбранное значение сохраняется в списке (см. рисунок 7.7).

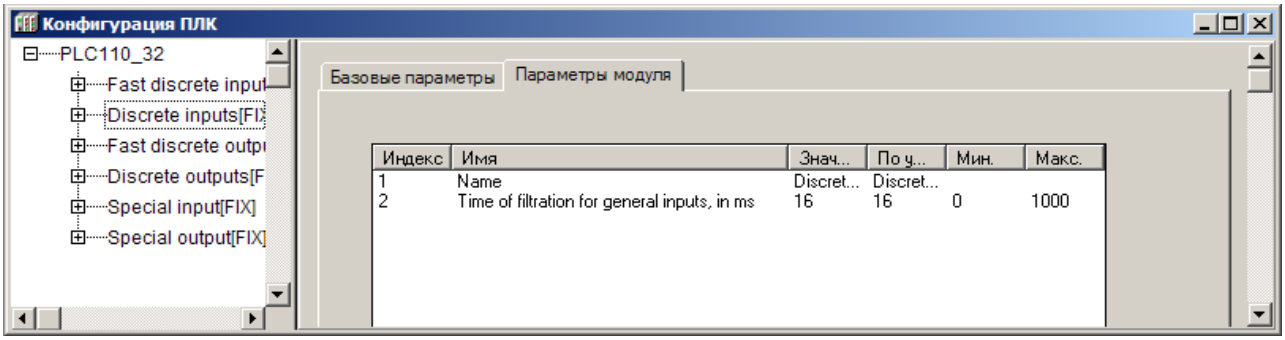


Рисунок 7.7 – Окно режима «Конфигурация ПЛК (PLC Configuration)». Модуль дискретных входов. Вкладка «Параметры модуля»

7.1.1.6 Каналы модуля

В состав модуля входят **каналы** (битовые, байтовые, каналы для данных типа REAL или STRING).

Каждый канал – это транслятор данных от внешнего оборудования в область памяти ввода-вывода ПЛК и, если требуется, их преобразователь в цифровой вид. Через канал передается значение входов и выходов (физических или сетевых), также в канале указывается, в каком месте памяти области ввода-вывода хранится данное значение (каждому каналу соответствует переменная в области ввода-вывода).

Каналу и соответствующей ячейке памяти может быть присвоено имя. При этом следует соблюдать правила именования переменных:

1) Имя может состоять из латинских букв, цифр и знака «_» (подчеркивание).
2) Имя должно начинаться с буквы или знака «_».
3) Имя должно быть уникальным.
4) В некоторых случаях редактирование имен каналов может быть запрещено.

По присвоенному имени к переменной можно обращаться из программы. Возможен также вызов переменной канала из программы по тому адресу, который установлен у нее аппаратно (например, **%IX 0.0.1**).

Получать данные из канала можно с тремя способами:

1) Задать параметру в канале имя, и в программе обращаться к этим данным по имени. Этот способ рекомендуется использовать, если текущая конфигурация ПЛК проста (например, включает стандартную конфигурацию контроллера и несколько модулей ввода / вывода).

2) Объявить необходимое количество глобальных переменных (в окне ресурса «Глобальные переменные (Global variables)» и связать добавленную переменную с областью ввода / вывода через МЭК адрес, указывающий, в какой области памяти ввода / вывода хранится значение переменной.

Например: дискретный вход1: **DI1 AT %IX0.0.1: bool**;

где: **DI1** – имя переменной, задаваемое пользователем; **AT** – указатель, что свое значение переменная будет получать из памяти ввода / вывода; **%IX0.0.1** – указание на ячейку хранения значения в области памяти ввода\вывода.

***Примечание**: данный адрес однозначно прописан в соответствующем канале конфигурации ПЛК. Этот способ рекомендуется использовать, если текущая конфигурация ПЛК сложна.*

3) Обращаться к значениям переменных в программе непосредственно через МЭК-адрес переменной в области ввода \ вывода. Этот способ использовать не рекомендуется.

Окно, представленное на рисунке 7.8, иллюстрируют процесс именования канала – появление поля ввода символов.

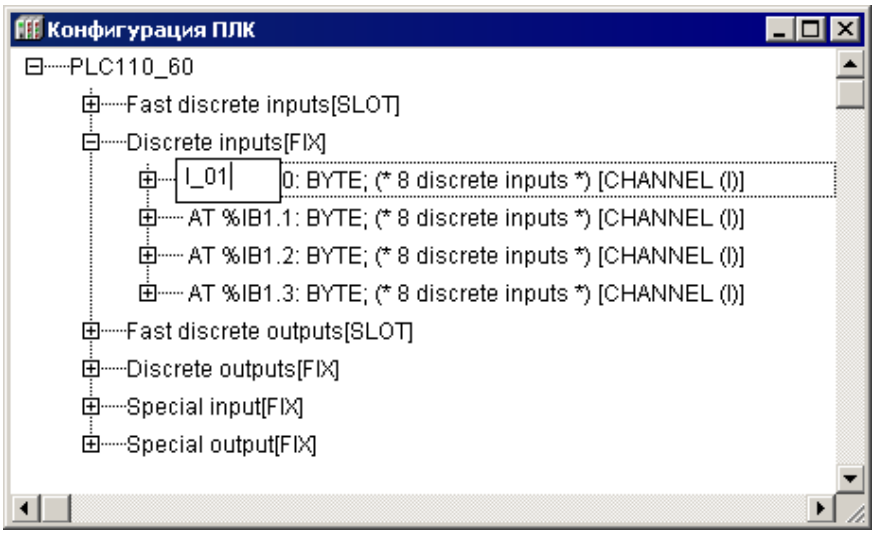


Рисунок 7.8 –Ввод и редактирование имени переменной канала

Данные, отображаемые в полях вкладки «Базовые параметры», носят информационный характер и (за исключением текста комментария) не редактируются. Для байтового канала отображаются следующие данные:

- **комментарий** – характеристика канала (например, для модуля дискретных входов – «8 discrete inputs» = «8 дискретных входов»);
- **ID канала** – идентификационный номер канала в общем списке;
- **размер** – в битах»

Для битового канала программа выводит только комментарий с номером битового канала, например, «Bit 3».

7.2 Задание времени цикла ПЛК

Для изменения параметров времени цикла ПЛК, следует:

1) В дереве конфигурации выделить корневой элемент (например, «PLC110-60», см. рисунок 7.9).
2) В области задания параметров – перейти на вкладку «Параметры модуля (ModuleParameters)», щелчком левой кнопки мыши в требуемой ячейке списка параметров, переведя запись значения, установленного по умолчанию, в режим редактирования (см. рисунок 7.9).

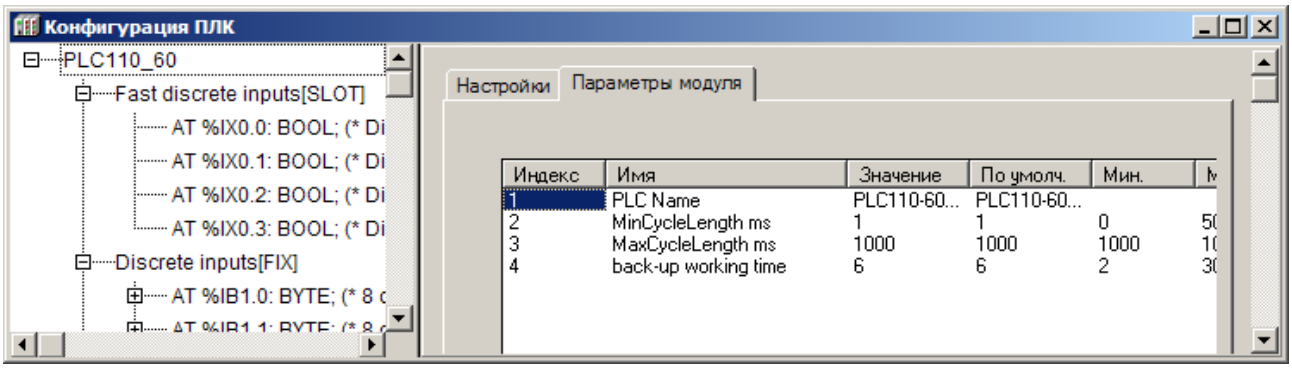


Рисунок 7.9 – Окно настройки параметров работы ПЛК

3) Задать требуемые значения параметров времени цикла ПЛК:

PLCName – символическое имя контроллера, используемое в текущем проекте. Максимальная длина имени – 80 символов, имя может содержать любые русские и латинские буквы и знаки и предназначено для хранения описаний элементов конфигурации ПЛК и их назначения непосредственно в ПЛК. Данные значения могут быть использованы, например, для идентификации контроллера в сети из нескольких или вывода на ЖК-панель.

MinCycleLength, ms (Минимальное значение цикла работы ПЛК, в мс) – параметр определяет минимальный период, с которым ПЛК выполняет полный цикл своей работы. Диапазон значений от 0 до 50 мс, значение по умолчанию – 1 мс.

Программная реализация ПЛК обеспечивает вызов цикла ПЛК не чаще, чем 1 раз в установленное число миллисекунд. Поэтому, задавая значение параметра**,** следует учесть, что после выполнения цикла ПЛК (т.е. после выполнения операции ввода данных, выполнения пользовательской программы и вывода данных) выполняется еще ряд сервисных функций (обеспечивающих сетевой обмен, работу с файлами и т.д.), на выполнение которых также требуется процессорное время. И если пользовательская программа ПЛК выполняется за время, превышающее 70-80% от значения, заданного в параметре «MinCycleLength», то на выполнение сервисных операций контроллеру не остается времени. При этом возможны сбои, замедление или прекращение сетевого обмена с модулями ввода-вывода, сбои в записи архивов и т.д. Для исправления некорректной ситуации следует увеличить значение параметра.

Узнать о времени выполнения пользовательской программы, сервисных функций и о времени простоя процессора можно в модуле «Statistic» (см. п. 7.4.7).

Для нормальной работы рекомендуется, чтобы время простоя процессора составляло не менее 20% от значения, заданного в параметре «MinCycleLength».

Значение параметра «MinCycleLength» может быть задано равным нулю. Тогда в контроллере отключается контроль времени вызова цикла ПЛК. После выполнения предшествующего цикла и после выполнения всех сервисных функций происходит вызов следующего цикла ПЛК. При этом не гарантируется строгое выполнение цикла через равные промежутки времени, т.к. длительность выполнения сервисных функций может изменяться от цикла к циклу.

НЕ РЕКОМЕНДОВАНО УСТАНАВЛИВАТЬ ВРЕМЯ ЦИКЛА РАВНЫМ НУЛЮ!

MaxCycleLength, ms (Максимальное значение цикла работы ПЛК, в мс) – параметр определяет максимально допустимое время, за которое ПЛК выполняет полный цикл своей работы. Если в процессе работы ПЛК заданная величина будет превышена (при зависании программы или при выполнении бесконечного цикла), то ПЛК будет принудительно перезагружен. Т.е., параметр «MaxCycleLength» задает время ожидания **сторожевого таймера** («WatchDog Timer»).[6] Диапазон значений от 1000 до 10000 мс, значение по умолчанию – 1000 мс.

Backup working time – параметр определяет, какое время после пропадания питания должна выполняться пользовательская программа, измеряется в секундах. Диапазон значений от 2 секунд до 30 секунд (2 секунды – минимум времени, необходимого контроллеру для корректного окончания работы). Параметр позволяет подключать или отключать функцию выполнения программы в случае резервного питания от батареи при пропадании напряжения основного источника питания, либо для отключения влияния на работу ПЛК кратковременных пропаданий питания.

7.3 Фиксированные модули (элементы) конфигурации. Входы и выходы

7.3.1 Fast Discrete inputs (Быстрые дискретные входы)

Модуль быстрых (высокочастотных) дискретных входов (Fast Discrete input) отображает в области ввода/вывода значения, характеризующие состояния дискретных быстрых (высокочастотных) входов ПЛК.

Модуль имеет несколько битовых каналов – по числу быстрых входов.

Параметры модуля (см. рисунок 7.10):

– **Name** – символическое имя элемента конфигурации ПЛК, используемое в текущем проекте. Максимальная длина имени – 80 символов, имя может содержать любые русские и латинские буквы и знаки и предназначено для хранения описаний элементов конфигурации ПЛК и их назначения непо-

[6] Если для проекта важно строгое ограничение времени цикла, то значение MaxCycleLength должно быть значительно меньше значения по умолчанию. Значение следует увеличивать в зависимости от загруженности портов ввода-вывода ПЛК: RS-232, RS-485 и Ethernet. При неиспользуемых портах ввода-вывода значение можно задать не более 20 мс, что для несложных приложений будет достаточным. С каждым занятым портом значение рекомендуется увеличивать на 50 мс, то есть, если заняты три порта, то рекомендуемое значение ограничения по времени цикла – 150 мс.

средственно в ПЛК. Данные значения могут быть использованы, например, для вывода на ЖК-экран или панель.

– **Time of filtration fast inputs, mks (Время фильтрации быстрых входов, мкс)** – период опроса состояния одного дискретного входа; задается в микросекундах (1 ед. = 1 мкс). Диапазон значений от 0 до 65535, значение по умолчанию – 10000 (10 мс).
Принцип действия фильтрации:

  - в сдвиговом регистре в драйвере каждого дискретного входа накапливаются значения четырех последних состояний, полученных в результате опроса с периодом, заданным в параметре «Время фильтрации»;
  - если состояние битового канала дискретного входа равно **1 (TRUE)**, а количество единиц в сдвиговом регистре менее двух, то битовый канал переключается на **0 (FALSE);**
  - если состояние битового канала равно **0 (FALSE)**, а количество единиц в сдвиговом регистре три и более, то битовый канал переключается на **1 (TRUE);**
  - если количество единиц в сдвиговом регистре равно 2, то состояние битового канала дискретного входа не меняется.

Режим фильтрации может быть отключен установлением в параметре «Время фильтрации» значения «**-1**». Отключение фильтрации необходимо при работе с подчиненными модулями энкодеров для того, чтобы не пропускать высокочастотные сигналы, а также в тех случаях, когда ПЛК функционирует без ограничения цикла по частоте, т.е. на максимально возможной частоте.

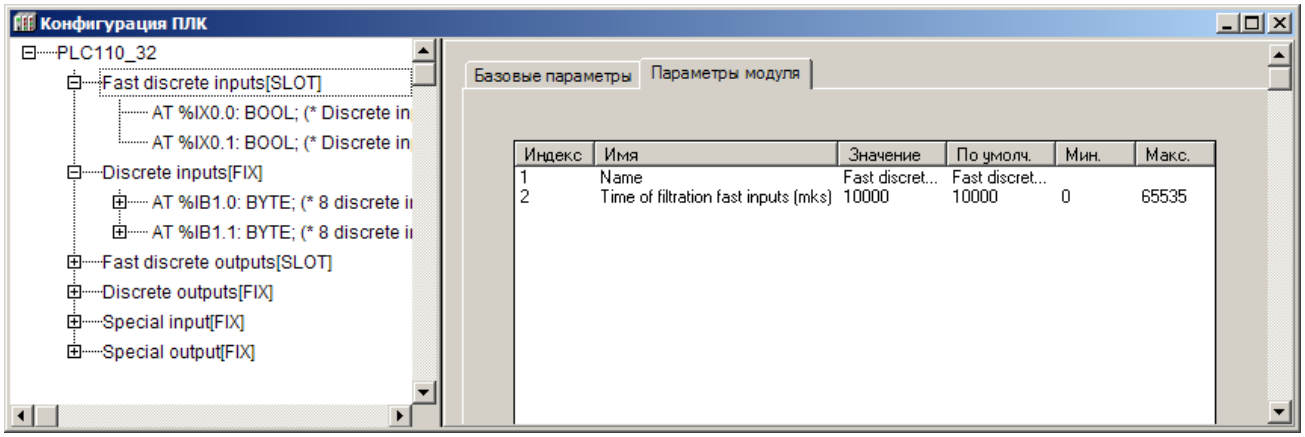


Рисунок 7.10 – Параметры модуля быстрых (высокочастотных) дискретных входов (Fast Discrete input)

7.3.2 Замещающие элементы (модули)

Процедура замещения модуля описана в разделе 7.1.1.3.

Список замещающих модулей модуля «Быстрые дискретные входы (Fast Discrete input)» (см. рисунок 7.11):

– Fast Counters (Высокочастотный Счетчик) (см. п.7.3.2.1),
– Fast Encoder (Высокочастотный энкодер) (см. п.7.3.2.2),
– Fast Z-encoder (Высокочастотный Z-энкодер), только для ПЛК110-60 (см. п. 7.3.2.3)

– Fast discrete inputs – direct control (Прямое управление дискретными быстрыми входами), только для ПЛК110-60 (см. п. 7.3.2.4).

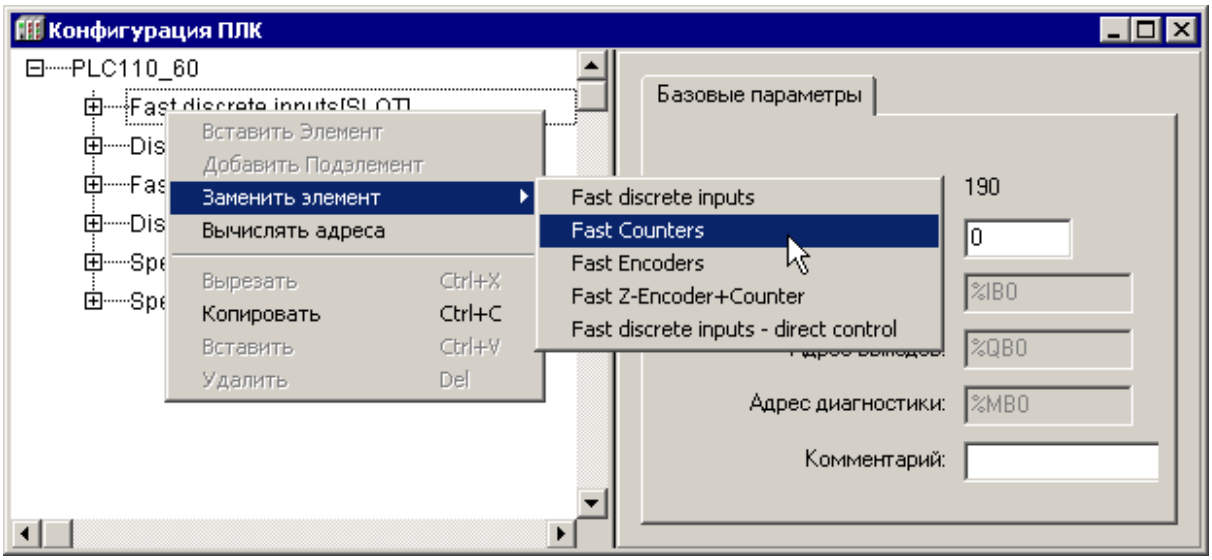


Рисунок 7.11 – Замещающие модули
модуля «Быстрые дискретные входы (Fast Discrete input)»

7.3.2.1 Fast Counters (Высокочастотный Счетчик)

Высокочастотный счетчик – программный модуль, ведущий учет входных импульсов, поступающих на быстрые дискретные входы ПЛК и экспортирующий учетные данные в программу ПЛК, в соответствующее место в области памяти входов / выходов.

Модуль **«Высокочастотный Счетчик» (Fast Counter)** является модулем, замещающим модуль быстрых дискретных входов.

Значение быстрого счетчика каждый программный цикл увеличивается на количество импульсов на входе, которое он зарегистрировал в течение цикла. Счетчик обнуляется при достижении значения большего, чем 65535 ($FFFF_{16}$), либо, если происходит не увеличение, а уменьшение значения счетчика, после того, как будет достигнут ноль, значение счетчика станет равно 65535 ($FFFF_{16}$). Подсчет импульсов осуществляется по переднему (возрастающему) фронту импульса. Таким образом, для использования и обработки значения данного канала, необходимо считывать его каждый раз в начале цикла пользовательской программы. Например, ввести в программу дополнительную переменную, и в начале цикла передавать в эту переменную значение переменной, привязанной к счетчику.

Модуль имеет два (для ПЛК110-30 и ПЛК110-32) или четыре (для ПЛК110-60) шестнадцатибитовых каналов, по числу быстрых входов.

Модуль не имеет параметров.

7.3.2.2 Fast Encoder (Высокочастотный Энкодер)

Высокочастотный энкодер – программный модуль, позволяющий осуществлять подключение на двух быстрых дискретных входах ***относительного энкодера*** для получения с его помощью данных о вращении или линейном перемещении контролируемого механизма с последующей передачей информации в цифровой форме в программу ПЛК. Для работы с механическими энкодерами необходимо включать режим фильтрации дребезга сигналов в параметре **Time of filtration in**

mks. Схемы подключения энкодеров к быстрым входам ПЛК приводятся в руководствах по эксплуатации.

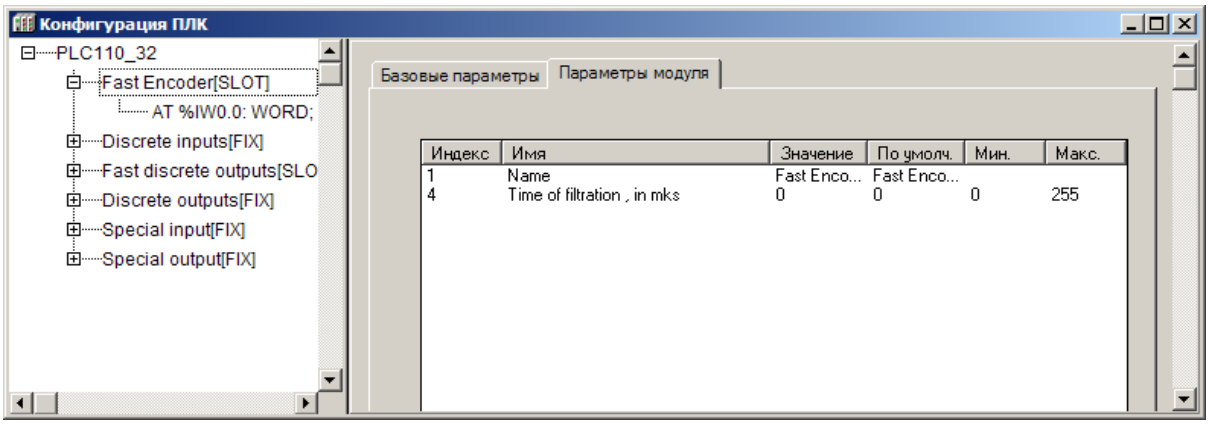
Модуль является замещающим для модуля Быстрых дискретных входов (Fast discrete input).

Модуль имеет один или два 16-ти битовых канала (формат WORD), по максимальному числу подключаемых к контроллеру энкодеров.

Параметры модуля (см. рисунок 7.12):

- **Name** – символическое имя элемента, см. п. 7.3.1.
- **Time of filtration in mks (Время фильтрации, в микросекундах)** – задает время установления сигнала, значения от 0 до 255 (1 единица = 1 мкс).
  Параметр задает время, в течение которого контроллер игнорирует дребезг контактов механического энкодера. Отсчет времени начинается с момента переключения выхода энкодера в противоположное положение. Если по истечении заданного времени сигнал выхода энкодера
  не изменяется, то считается, что он установился и дребезг контакта закончился.
  Значение параметра задается в сотнях микросекунд (т.е. 1 единица равна 100 мкс, 10 ед. = 1 мс).
  Для механических энкодеров рекомендуется устанавливать значение параметра в диапазоне от 20 до 40 (от 2 до 4 мс).
  Для отключения фильтрации (при применении оптических энкодеров) следует задать параметру значение 0.
  Если в модуле имеется один параметр, то его действие распространяется на все энкодеры, подключенные к контроллеру.
- **Visibility (Видимость)** – задает видимость параметров модуля в протоколе «Gateway» (в частности, в программе «EasyWorkPLC» разработки ПО «ОВЕН»). Значения выбираются из списка **«yes»** и **«no»,** значение по умолчанию – **«no».**

а)



б)

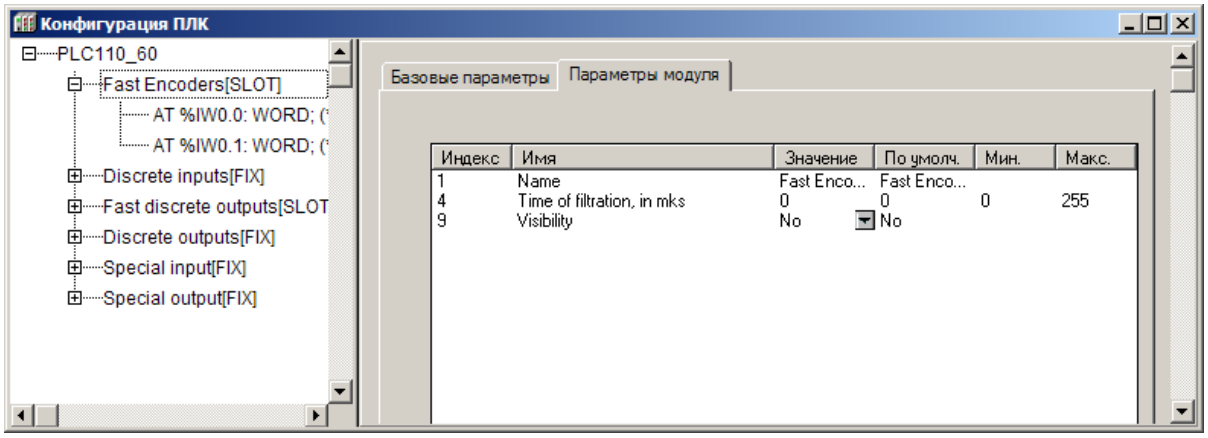


Рисунок 7.12 – Параметры модуля Высокочастотный Энкодер (FastEncoder) для ПЛК110-32 (а) и ПЛК110-60 (б)

7.3.2.3 Fast Z-Encoder + Counter (Высокочастотный Z - Энкодер + Счетчик)

Высокочастотный Z-энкодер – программный модуль, позволяющий осуществлять подключение на трёх быстрых дискретных входах **относительного энкодера с отметкой пересечения нуля** для получения с его помощью данных о вращении или линейном перемещении контролируемого механизма, с последующей передачей информации в цифровой форме в программу ПЛК. Для работы с механическими энкодерами необходимо включать режим фильтрации дребезга сигналов в параметре **Time of filtration in mks**.

Модуль является замещающим для модуля **Быстрых дискретных входов (Fast discrete input).**

Неиспользуемый при подключении Z-энкодера дискретный вход может быть использован в качестве высокоскоростного счетчика.

Модуль имеет два шестнадцатибитовых канала (формат WORD), по числу подключаемых к контроллеру Z- энкодеров и счетчиков.

Модуль доступен только в ПЛК110-60.

Параметры модуля (см. рисунок 7.13):

- **Name** – символическое имя элемента, см. п. 7.3.1.
- **Time of filtration in mks (Время фильтрации, в микросекундах)** – задает время установления сигнала, значения от 0 до 255, 1 единица = 1 мкс (см. п. 7.3.2.2)
- **Visibility (Видимость)** – задает видимость параметров модуля в протоколе «Gateway» (в частности, в программе «EasyWorkPLC» разработки ПО «ОВЕН»). Значения выбираются из списка **«yes»** и **«no»,** значение по умолчанию – **«no».**

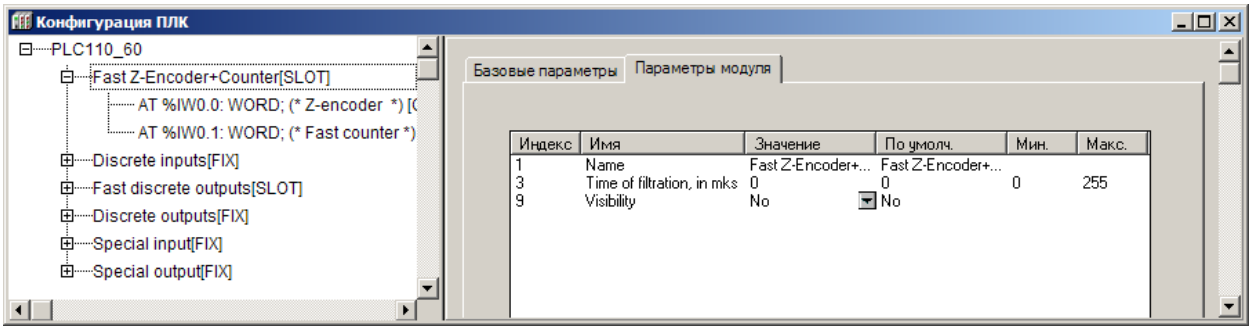


Рисунок 7.13 – Параметры модуля Высокочастотный Z-Энкодер+Счетчик (FastZ-Encoder+Counter)

Особенностью подключения Z – энкодера являеться порядок подключения выходов энкодера. Контроллер требует следующего подключения:

На первый быстрый вход – DI1 - сигнал A.

На второй быстрый вход – DI2 – сигнал B.

На третий быстрый вход – DI3 – сигнал Z/

Четвертый быстрый вход – DI4 – может использоватся в качестве высокоскоростного счетчика.

7.3.2.4 Fast discrete inputs – direct control (Прямое управление быстрыми дискретными входами)

Модуль переводит быстрые дискретные входы в режим прямого управления из функций стандартной библиотеки SysLibPort. При этом входы не отображаются в пространстве области ввода (%I).

Прямое управление быстрыми входами из библиотеки SysLibPort необходимо для управления быстрыми входами из процедуры обработки прерываний высокочастотного таймера, т.к. вызов обработчика по таймеру происходит чаще, чем заканчивается цикл ПЛК и, соответственно, чаще, чем происходит обращение к памяти вывода.

Модуль является замещающим для модуля **Быстрых дискретных входов (Fast discrete input).**

При установке модуля все быстрые входы переключаются на режим прямого управления из библиотеки SysLibPort и не реагируют на изменение значений каналов в памяти ввода.

Модуль доступен в ПЛК110-60.

Модуль не имеет параметров и каналов.

Работа с высокочастотным таймером подробно описана в разделе 9 .

7.3.3 Discrete inputs (Дискретные входы)

Фиксированный модуль «Discrete inputs (Дискретные входы)» отображает в области ввода/вывода значения, характеризующие состояния дискретных входов ПЛК.

Модуль имеет два (для ПЛК110-30 и ПЛК110-32) или четыре (для ПЛК110-60) восьмибитовых канала.

Параметры модуля (см. рисунок 7.14):

– **Name** – символическое имя элемента, см. п. 7.3.1.

– **Time of filtration general inputs in ms (Время фильтрации основных входов в миллисекундах)** – период опроса состояния одного дискретного входа; задается в сотнях микросекунд (1 ед. = 1 мс). Диапазон значений от 0 до 1000, значение по умолчанию – 10.

Принцип действия фильтрации:

в сдвиговом регистре в драйвере каждого дискретного входа накапливаются значения восьми последних состояний, полученных в результате опроса с периодом, заданным в параметре «Время фильтрации»;

если состояние битового канала дискретного входа равно **1 (TRUE)**, а количество единиц в сдвиговом регистре менее двух, то битовый канал переключается на **0 (FALSE);**

если состояние битового канала равно **0 (FALSE)**, а количество единиц в сдвиговом регистре 7 и более, то битовый канал переключается на **1 (TRUE);**

если количество единиц в сдвиговом регистре от 2 до 6, то состояние битового канала дискретного входа не меняется.

Режим фильтрации может быть отключен установлением в параметре «Время фильтрации» значения «**-1**». Отключение фильтрации необходимо при работе с подчиненными модулями энкодеров для того, чтобы не пропускать высокочастотные сигналы, а также в тех случаях, когда ПЛК функционирует без ограничения цикла по частоте, т.е. на максимально возможной частоте.

Примечание. *На вкладке модуля дискретных входов представлены восемь одноименных параметров «Время фильтрации» – для каждого битового канала (входа), соответственно.*

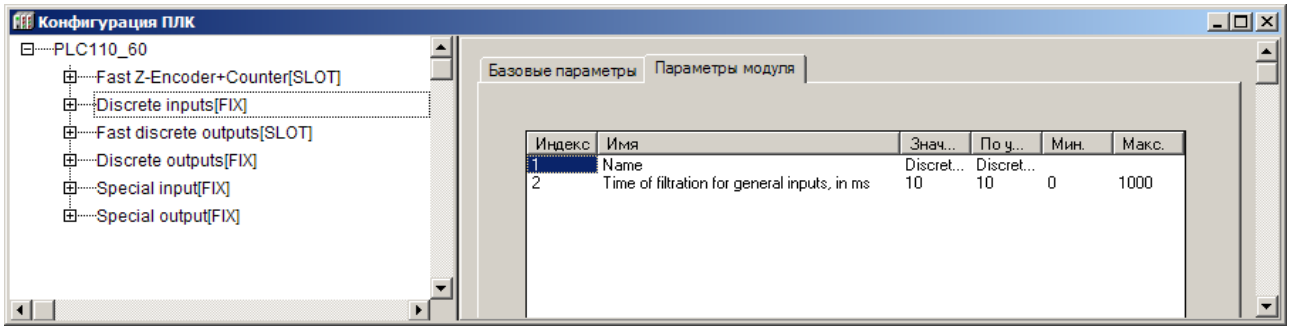


Рисунок 7.14 – Параметры модуля Дискретные входы (Discrete input)

7.3.4 Fast discrete outputs (Быстрые дискретные выходы)

Фиксированный модуль быстрых (высокочастотных) дискретных выходов **(Fast discrete output)** отображает в области памяти ввода/вывода значения быстрых дискретных выходов ПЛК.

Модуль имеет несколько битовых каналов по числу быстрых выходов контроллера.

Параметры модуля (см. рисунок 7.15):

- **Name** – символическое имя элемента, см. п. 7.3.1.
- **Safe Value (Безопасное значение)** – **TRUE** или **FALSE** для быстрого дискретного выхода.
  Назначение параметра следующее: при загрузке или при сбое в работе ПЛК, его выходы могут оказаться выключены или включены. Такая неопределенность может быть недопустима при эксплуатации управляемого оборудования, и для исключения подобной ситуации ПЛК переводит выходы при сбое или во время загрузки в состояние, заданное в параметре **«Безопасное состояние выхода» (Safe Value).** Значения параметра: **FALSE** означает, что выход выключен (=0), **TRUE** – выход включен (=1). Значение параметра устанавливается раздельно для каждого битового канала.
- **Visibility (Видимость)** – задает видимость параметров модуля в протоколе «Gateway» (в частности, в программе «EasyWorkPLC» разработки ПО «ОВЕН»). Значения выбираются из списка **«yes»** и **«no»,** значение по умолчанию – **«no».**

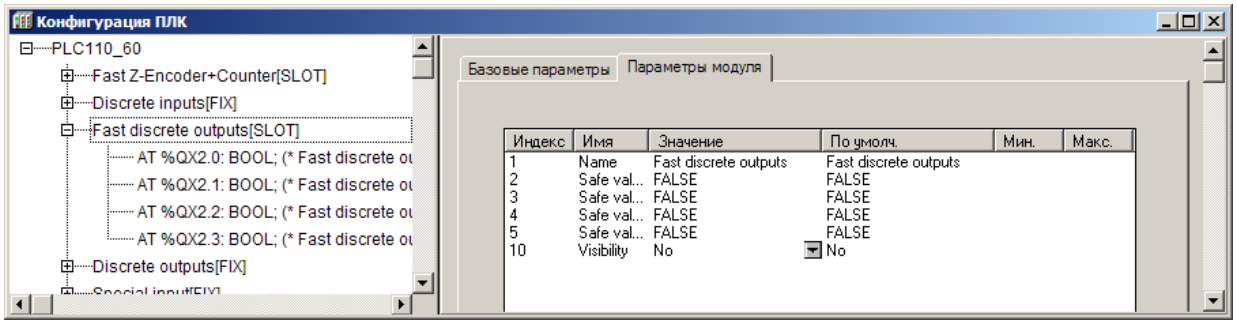


Рисунок 7.15 – Параметры модуля дискретные выходы (Fast discrete output)

7.3.5 Замещающие элементы (модули)

Процедура замещения модуля описана в разделе 7.1.1.3.

Список замещающих модулей модуля «Быстрые дискретные выходы (Fast Discrete output)» (см. рисунок 7.16):

- PWM (Pulse-wide modulator) – ШИМ, см. п. 7.3.5.1,
- Fast discrete outputs – Direct control (Прямое управление быстрыми дискретными выходами), только для ПЛК110-60, см. п.7.3.5.2.

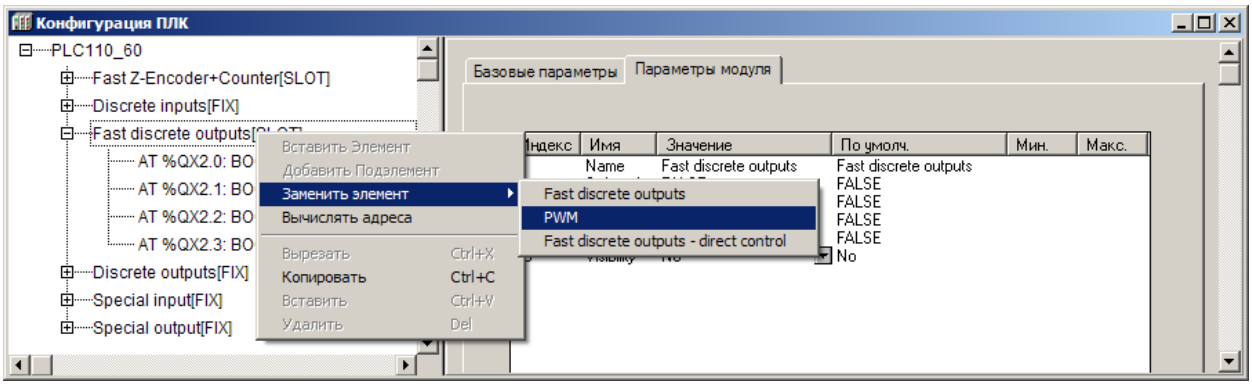


Рисунок 7.16 – Замещающие модули
модуля «Быстрые дискретные выходы (Fast Discrete output)»

7.3.5.1 PWM (Pulse-wide modulator) –ШИМ

Модуль ШИМ **(PWM – Pulse-wide modulator)** – программный модуль, предназначенный для обеспечения функционирования генератора **широтно-импульсной модуляции**, подключенного к дискретному выходу.

Модуль является замещающим для модуля быстрых дискретных выходов.

Модуль имеет каналы:

- **PWM power** – 16-ти битовый канал (формат WORD), задающий значение скважности ШИМа. Изменяется от 0 (0 %) до 1000 (100 %).
- **PWM Period** – 32-х битный канал (формат DWORD), позволяющий задать или прочитать значение периода ШИМ. Диапазон значений от 100 до 360000, задается в 100 мкс.
  В начале работы значение периода ШИМ записывается из одноименного параметра в канал, затем оно может быть изменено в канале. Измененное значение канала PWMPeriod не передается в одноименный параметр модуля, поэтому при выключении контроллера оно не сохраняется.

Параметры модуля (см. рисунок 7.17):

- **Min. duration of PWM in 100 mksec (Минимальная длительность импульса ШИМ в 100 мксек)** – диапазон значений от 1 до 65000, значение по умолчанию – 3000 мкс. Устанавливается ограничение на минимальную длительность импульса ШИМ.
- **PWM default Period in 100 mks (Период ШИМ в 100 мкс)** – задается длительность одного периода ШИМ-регулирования (в сотнях мкс). Принимает значения от 100 до 360000 единиц (1 ед. = 100 мкс), соответственно задавая период ШИМ от 10 миллисекунд до 36 секунд. При наличии одноименного канала в модуле значение параметра переписывается в канал при загрузке пользовательской программы. В дальнейшем значение периода ШИМ может быть изменено в канале модуля.
- **Visibility (Видимость)** – задает видимость параметров модуля в протоколе «Gateway» (в частности, в программе «EasyWorkPLC» разработки ПО «ОВЕН»). Значения выбираются из списка **«yes»** и **«no»,** значение по умолчанию – **«no».**

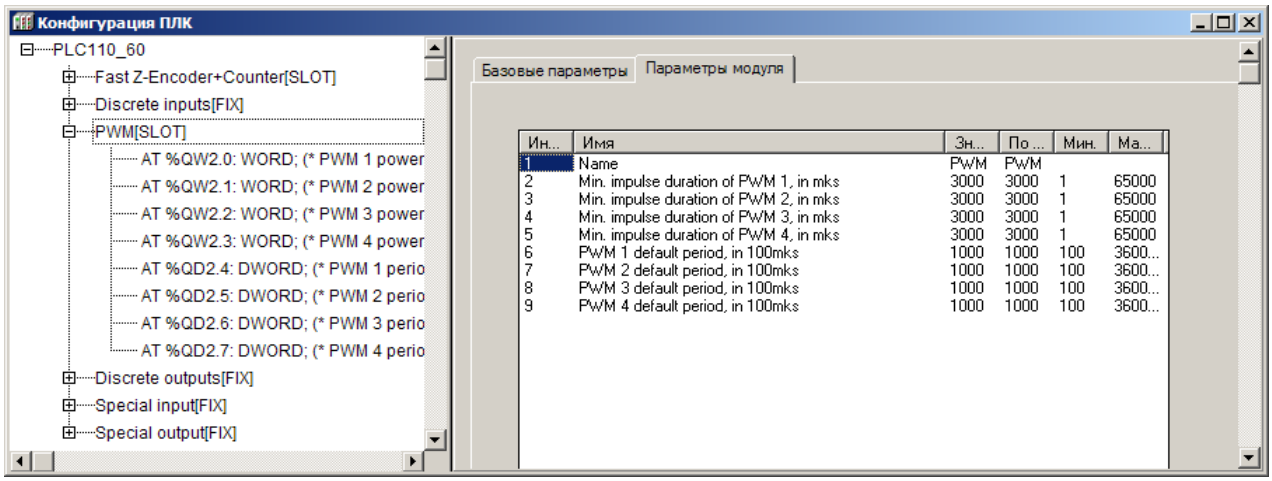


Рисунок 7.17 – Параметры модуля ШИМ (PWM – Pulse-wide modulator)

7.3.5.2 Fast discrete outputs – Direct control (Прямое управление быстрыми дискретными выходами)

Модуль переводит быстрые дискретные выходы в режим прямого управления из функций стандартной библиотеки SysLibPorts.lib. При этом выходы не отображаются в пространстве области вывода (%Q).

Прямое управление быстрыми выходами из библиотеки SysLibPorts.lib необходимо для управления быстрыми выходами из процедуры обработки прерываний высокочастотного таймера, т.к. вызов обработчика по таймеру происходит чаще, чем заканчивается цикл ПЛК и, соответственно, чаще, чем происходит обращение к памяти вывода.

При установке модуля все быстрые выхода переключаются на режим прямого управления из библиотеки SysLibPorts.lib и не реагируют на изменение значений каналов в памяти вывода.

Модуль не имеет параметров и каналов.

7.3.6 Discrete outputs (Дискретные выходы)

Модуль дискретных выходов (Discrete outputs) отображает в области памяти ввода/вывода значения дискретного выхода ПЛК.

Модуль имеет несколько битовых каналов (количество каналов зависит от варианта исполнения контроллера).

Параметры модуля (см. рисунок 7.18):

- **Name** – символическое имя элемента, см. п. 7.3.1.
- **Safe Value (Безопасное значение)** – **TRUE** или **FALSE** для быстрого дискретного выхода.
  Назначение параметра следующее: при загрузке или при сбое в работе ПЛК, его выходы могут оказаться выключены или включены. Такая неопределенность может быть недопустима при эксплуатации управляемого оборудования, и для исключения подобной ситуации ПЛК переводит выходы при сбое или во время загрузки в состояние, заданное в параметре **«Безопасное состояние выхода» (Safe Value).** Значения параметра: **FALSE** означает, что выход выключен (= 0), **TRUE** – выход включен (= 1). Значение параметра устанавливается раздельно для каждого битового канала.

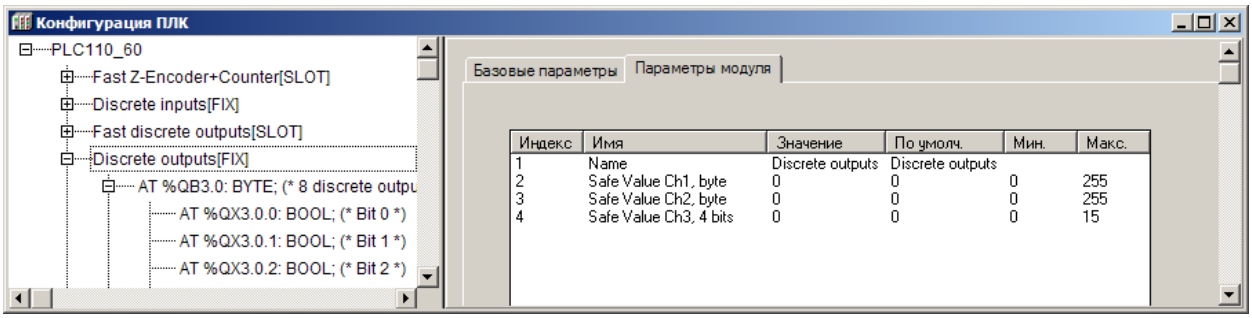


Рисунок 7.18 – Параметры модуля Дискретных выходов (Discrete outputs)

7.3.7 Special input (Специальный дискретный вход)

Модуль специального дискретного входа **(Special input)** – программный модуль, отображающий в область памяти ввода/вывода состояние тумблера «Работа/Стоп/Сброс», расположенной на передней панели контроллера. В рабочем режиме тумблер функционирует как дискретный вход, при этом положению «Работа» соответствует состояние TRUE, а положению «Стоп» - FALSE. Состояние «Сброс никак не отображается.

Модуль имеет один битовый выходной канал и один параметр: **«Видимость» (Visibility)**, задающий видимость параметров модуля в протоколе «Gateway» (в частности, в программе «EasyWorkPLC» разработки ПО «ОВЕН»). Значения выбираются из списка **«yes»** и **«no»,** значение по умолчанию – **«no».**

7.3.8 Special output (Специальный дискретный выход)

Модуль специального дискретного выхода (**Special output)** – модуль, содержащий битовую переменную, управляющую специальным оборудованием – устройством подачи звукового сигнала. При значении переменной – **TRUE** подаётся непрерывный звуковой сигнал.

Модуль имеет битовый канал и один параметр: **«Видимость» (Visibility)**, задающий видимость параметров модуля в протоколе «Gateway» (в частности, в программе «EasyWorkPLC» разработки ПО «ОВЕН»). Значения выбираются из списка **«yes»** и **«no»,** значение по умолчанию – **«no».**

7.4 Добавляемые модули и подмодули (подэлементы) конфигурации

Процедура добавления подмодулей (подэлементов) в текущую конфигурацию описана в п. 7.1.1.2.

В конфигурацию могут быть добавлены подмодули (подэлементы) настройки режимов работы (Master/Slave) и интерфейсов связи: ModBus (master), DCON (Master), OWEN (Slave), OWEN (Spy), OWEN (Master), ModBus (Slave).

Кроме того, в конфигурацию могут быть добавлены вспомогательные модули: Statistic (модуль статистики), Universal network module, Constant (переменная, объявляемая как константа. Нескольких типов: 32-битные константы (32bit signed constant), константы формата Real (Float constant) и 16-битные константы (16bit signed constant) и Archiver (архиватор).

7.4.1 Модуль ModBus (Master)

Модуль ModBus (Master) используется, когда требуется, чтобы контроллер работал в режиме Мастера сети, т.е. опрашивал и контролировал другие ModBus-устройства, работающие в сети в подчиненном режиме (slave) – например, модули ввода-вывода, панели оператора, частотные преобразователи и т.д. При установке

модуля «ModBus (Master)» выбирается коммуникационный интерфейс для обмена данными с другими устройствами, добавляются и настраиваются требуемые переменные.

Модуль имеет один параметр: **«Visibility (Видимость)»**, который задает видимость параметров модуля в протоколе «Gateway» (в частности, в программе «EasyWorkPLC» разработки ПО «ОВЕН»). Значения выбираются из списка **«yes»** и **«no»,** значение по умолчанию – **«no»**.

При опросе модулем «ModBus (Master)» подчиненных устройств информация о ходе обмена записывается в соответствующих каналах его переменных.

Каналы модуля:

- **Last Address** – адрес последнего опрошенного **ModBus (Slave)** устройства. Модуль запрашивает устройство, и, соответственно, тут же меняется значение: показывается значение адреса последнего запроса.
- **Last Error** – код ошибки. В переменной отображается код ошибки, если информационный обмен прошел неудачно. Это необходимо для корректности работы опрашиваемого устройства. Коды ошибок данного модуля представлены в Приложении В.

7.4.1.1 Добавление и настройка коммуникационных интерфейсов «ModBus (Master)»

При добавлении модуля «**ModBus (Master)**» в конфигурацию ПЛК в состав модуля уже подключен порт **Debug RS-232** (см. рисунок 7.19). При необходимости работы через другой коммуникационный интерфейс этот порт можно заменить требуемым последовательным портом или модемом (для этого следует выбрать команду «Заменить элемент | <Элемент>» контекстного меню строки «Debug RS-232»).

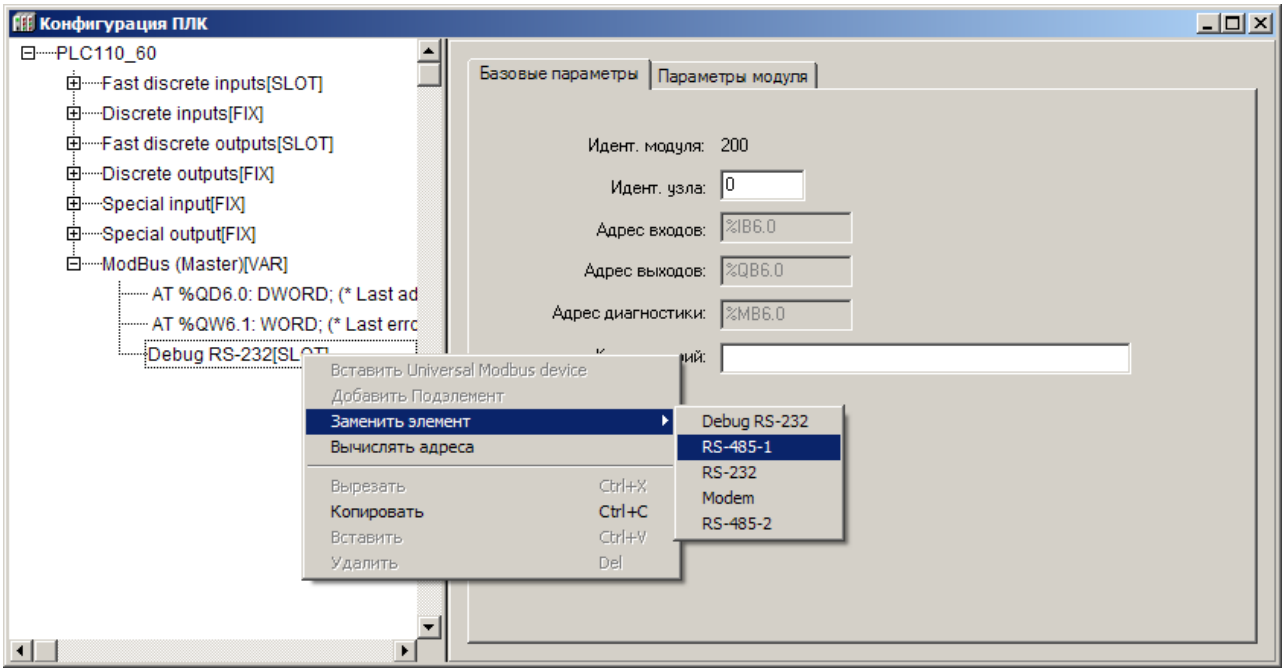


Рисунок 7.19 – Выбор коммуникационных интерфейсов «ModBus (Master)»

Внимание!
Если предполагается, что Мастер сети будет работать с устройствами по протоколу TCP, то необходимая настройка производится в подмодуле устройства (параметр «TCP port» модуля Universal ModBus Device).

7.4.1.2 Подмодуль UniversalModBusDevice

Для добавления в список опроса, проводимого мастером сети, устройств, работающих в режиме ModBus (Slave) следует в модуле ModBusMaster добавить подмодуль «Universal ModBus Device» (универсальное устройство ModBus). Для добавления в список нескольких опрашиваемых устройств эту процедуру следует повторить столько раз, сколько устройств должно быть подключено: для каждого устройства должен быть добавлен соответствующий ему модуль Universal ModBus Device с индивидуальными настройками.

Чтобы добавить подмодуль «Universal ModBus Device» в конфигурацию, следует выбрать команду «**Добавить подэлемент | Universal ModBus Device»** контекстного меню строки «ModBus (Master)».

Опрос добавленных «Universal ModBus device» производится последовательно, в порядке следования в конфигурации (если другой порядок не задан отдельно, в настройках модуля).

Универсальное устройство ModBus имеет один канал:**Start/Stop**:

- если в него записывается значение **0x00FF**, то происходит старт работы данного устройства ModBus;
- если модуль уже запущен, то повторная запись в канал значения **0x00FF** приводит к внеочередному запросу одной очередной переменной устройства ModBus;
- если в канал записано значение **0x00FE,** то происходит его остановка и прекращение всех посылок в сеть;
- если в канал ничего не записано, то опрос устройства производится автоматически, в порядке очереди.

При необходимости устройство можно исключить из списка опроса, подав соответствующую команду в канал Start\stop

При добавлении модуля «Universal ModBus Device» его параметры и идентификаторы не привязаны к конкретному внешнему устройству (модулю ввода-вывода, операторской панели). Конкретный вариант внешнего устройства производится заданием значений параметров модуля (конфигурированием модуля).

Параметры подмодуля «Universal ModBus device» (см. рисунок 7.20):

- **Name** – символическое имя элемента, см. п. 7.3.1.
- **Module IP (IP-адрес)** – задает IP-адрес ModBusSlave устройства, которым управляет **Мастер**, если обмен будет идти по TCP.
- **Max timeout (Максимальный тайм-аут, в мс)** – задает максимальное время, в течение которого устройство должно ответить на запрос. Если по истечении этого времени **Мастер** не получил ответ на запрос, то это значит, что произошел сбой или авария. Информация о сбое фиксируется в переменной модуля **Last error**. **Мастер** продолжает опрос других устройств. Максимальное значение не ограничено, оно может быть любым, в т.ч. дробным, но не меньше 10 мс, значение по умолчанию – 150.

- **TCP port (Порт TCP)** – используется только при обмене по TCP; стандартное значение для протокола **ModBus TCP** – 502 (значение по умолчанию), но при необходимости может быть установлено и другое.
- **Net Mode (Режим работы в сети)** – значения выбираются из списка («TCP» и «Serial»), значение по умолчанию – «TCP». Вариант «**TCP»** – подчиненное устройство работает по протоколу TCP используется интерфейс **Ethernet,** опрашиваемое внешнее устройство идентифицируется по IP-адресу**.** Вариант «**Serial»** – подчиненное устройство осуществляет обмен данными через последовательный интерфейс, опрашиваемое внешнее устройство идентифицируется по адресу в сети.
- **Module Slave Address (Адрес подчиненного устройства**) – диапазон значений от 1 до 247, значение по умолчанию – 1. **Внимание!** Значение «0» – специфично и используется для широковещательных сообщений. Например, при работе через шлюз.
- **Work mode (Режим работы)** – задает режим работы модуля **ModBus (Master)** при опросе внешних устройств может иметь несколько значений (значение по умолчанию – «Polling time»):

  By Poll time – «по времени» – контролируемые устройства опрашиваются с периодичностью, заданной в параметре Polling time (Период опроса устройства);

  By Value change – «по изменению значения переменных» – модуль **ModBus (Master)** генерирует запрос к соответствующему Slave устройству при изменении значений выходных переменных модуля;

  Примечание. Выходные переменные – значения, которые модуль ModBus (Master) передает (записывает) в Slave-устройства; входные переменные – параметры, значение которых Мастер запрашивает у Slave-устройств.

  Both – «оба варианта» – опрос параметра производится с временным интервалом, заданным в параметре «**Polling time»** и тогда, когда изменяются значения выходных переменных (в соответствии с значением «**By Value change**»);

  By Command – «по команде» – производится однократная посылка запроса, когда в командный канал «Command» переменной записывается значение **0x00FF**.

  Внимание! Для переменных с командным каналом при работе в режиме **By Command** (По команде) управление осуществляется следующим образом: первая посылка значения **0x00FF** в командный канал включает функционирование этой переменной, повторная посылка значения **0x00FF** инициирует проведение опроса. Аналогично опрос инициируется для переменных с командным каналом при работе в других режимах. При посылке в командный канал значения **0x00FE** переменная исключается из цикла опроса мастера.

- **Amount Repeat (Число повторов)** – определяет число повторов чтения/записи переменных при неудачном сеансе связи. В режиме**«По времени» (Polling time)** значение этого параметра не используется. Рекомендуемый диапазон значений от 0 до 5, значение по умолчанию – 0.
- **Byte Sequence (Порядок передачи байтов посылки)** – значения выбираются из списка: «Native» (порядок байтов, используемый в ПЛК) и «Trace_mode» (порядок байтов, используемый в программе Tracemode). Значение по умолчанию – «Trace_mode». Параметр определяет, в каком порядке будут передаваться байты посылки протокола **ModBus** для пере-

менных длиной **32 бита**. У устройств разных производителей этот порядок разный, он не стандартизирован в рамках протокола и поэтому должен быть задан для конкретного устройства. Для работы с модулями ввода/вывода ОВЕН (например МВА8) следует задать значение параметра **Trace_mode.**

– **Polling time (Период опроса устройства, в мс)** – определяет период опроса внешнего устройства. Диапазон значений от 10 до 10000, значение по умолчанию – 100.
Примечание. В Мастере, когда он работает в режиме **«По изменению значения переменных»** или **«По команде»**, нельзя ставить значение параметра **Polling time** слишком маленьким. По умолчанию его значение 100 мс, и в этих режимах оно не влияет на периодичность посылки запросов мастера. Но если на реальном проекте будет замечено, что Мастер при загрузке программы или при **Login** формирует лишние пакеты и/или запросы, которых не должно быть, значение параметра следует увеличить (до 200, 300 и т.д.) до предотвращения появления лишних пакетов.
– **Visibility (Видимость)** – задает видимость параметров модуля в протоколе «Gateway» (в частности, в программе «EasyWorkPLC» разработки ПО «ОВЕН»). Значения выбираются из списка **«yes»** и **«no»,** значение по умолчанию – **«no».**

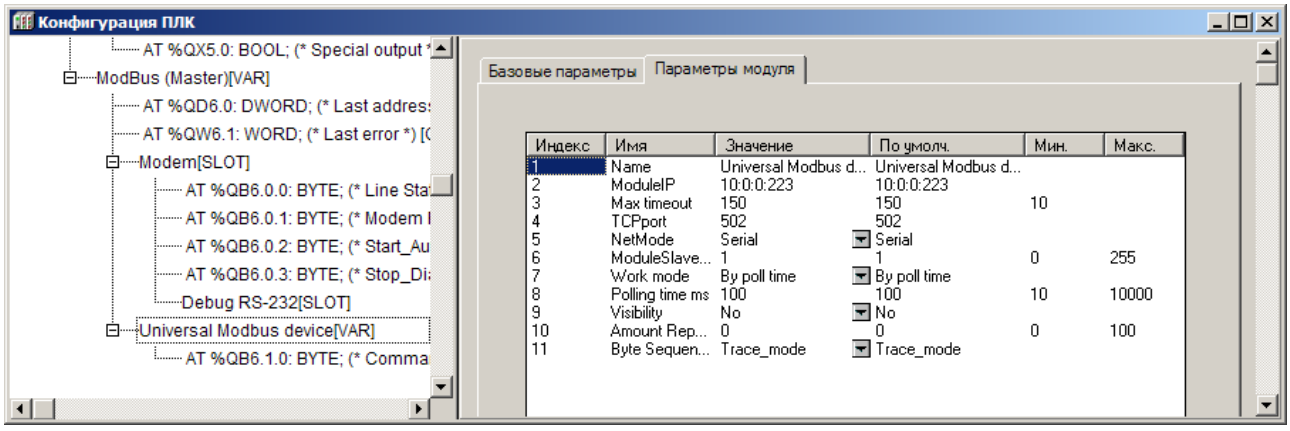


Рисунок 7.20 – Параметры подмодуля Universal ModBus device

7.4.1.2.1 Настройка входов и выходов подмодуля

После задания значений параметров подмодуля «Universal ModBus Device» к нему требуется подключить каналы, задающие входные параметры (параметры, значение которых Мастер запрашивает у Slave-устройств) и выходные параметры (значения, которые Мастер передает – записывает – в Slave устройства) подмодуля.

Добавление каналов производится выбором команды «Добавить подэлемент | <Наименование подэлемента>» контекстного меню строки «Universal ModBus Device» (см. рисунок 7.21).

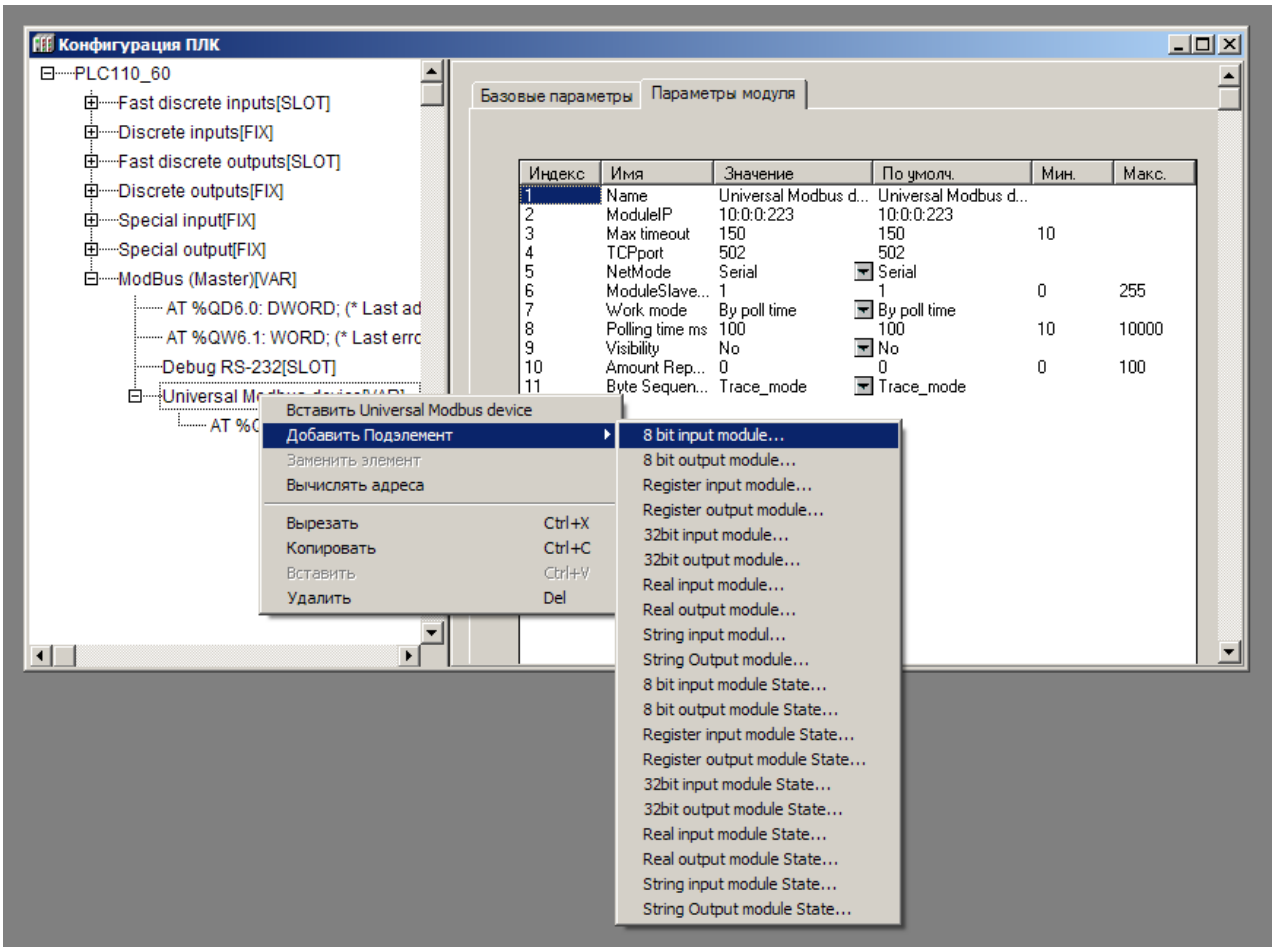


Рисунок 7.21 – Добавление каналов ввода/вывода подмодуля Universal ModBus device

При добавлении канала в «Universl ModBus Device» в дерево конфигурации добавляется подкаталог, содержащий внутри себя канал, в котором и отображаются полученное / передаваемое по сети значение (см. рисунок 7.22).

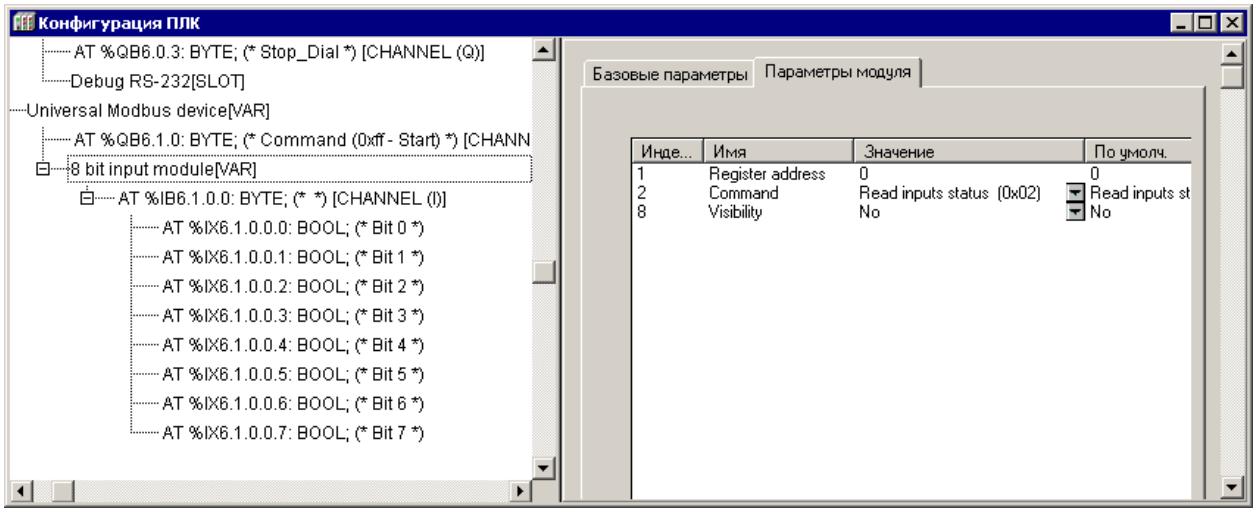


Рисунок 7.22 – Добавление каналов ввода/вывода

В «Universl ModBus Device» могут быть добавлены каналы, перечисленные в таблице 7.1.

Каналы (переменные) могут принадлежать следующим типам: REAL (32 бита), STRING (настраивается параметром, по умолчанию – 80 символов), 4 байта, 2 байта или 8 бит.

Таблица 7.1 – Типы каналов модуля «Universl ModBus Device»

Канал	**Размер в памяти**	**Command (Команда)**	**<C> / <3>****
8-bit input module Register input module 32-bit input module Real input module String input module 8-bit input module State Register input module State 32-bit input module State Real input module State String input module State	8 бит 2 байта 32 бит 32 бита 80 бит* 8 бит 2 байта 32 бита 32 бита 80 бит*	Read coils status (0x01), Read inputs status (0x02), Read holding registers (0x03), Read input registers (0x04), Read bytes (0x70)	C
8-bit output module 8-bit output module State	8 бит 8 бит	Force multiply coils (0x0f) Write bytes (0x71)	3
Register output module Register output module State	2 байта 2 байта	Preset single register (0x06) Write bytes (0x71) Write multiple registers (0x10)	3
String output module String output module State	80 бит* 80 бит*	Force multiple coils (0x0f) Preset multiple Registers (0x10) Preset singl Register (0x06) Write bytes (0x71)	3
32-bit output module Real output module 32-bit output module State Real output module State	32 бит 32 бита 32 бит 32 бита	Force multiply coils (0x0f) Preset multiple Registers (0x10) Write bytes (0x71)	3
*) Для каналов типа String указан размер 80 бит, задаваемый по умолчанию. Может изменяться пользователем. **) <C> – считывает, <3> – записывает			

Переменные с обозначением **«State»** кроме канала получения / передачи данных, содержат дополнительный управляющий канал (Command) для управления передачей, позволяющий организовать обмен значений отдельно взятых переменных (каналов) по команде пользователя (см. рисунок 7.23).

Управление запуском / остановкой обмена по каналам «Command» определяется значением, записываемым в канал:

- если в него записывается значение **0x00FF**, то происходит старт работы данного устройства ModBus;
- если модуль уже запущен, то повторная запись в канал значения **0x00FF** приводит к внеочередному запросу одной очередной переменной устройства ModBus;
- если в канал записано значение **0x00FE,** то происходит его остановка и прекращение всех посылок в сеть;
- если в канал ничего не записано, то опрос устройства производится автоматически, в порядке очереди.

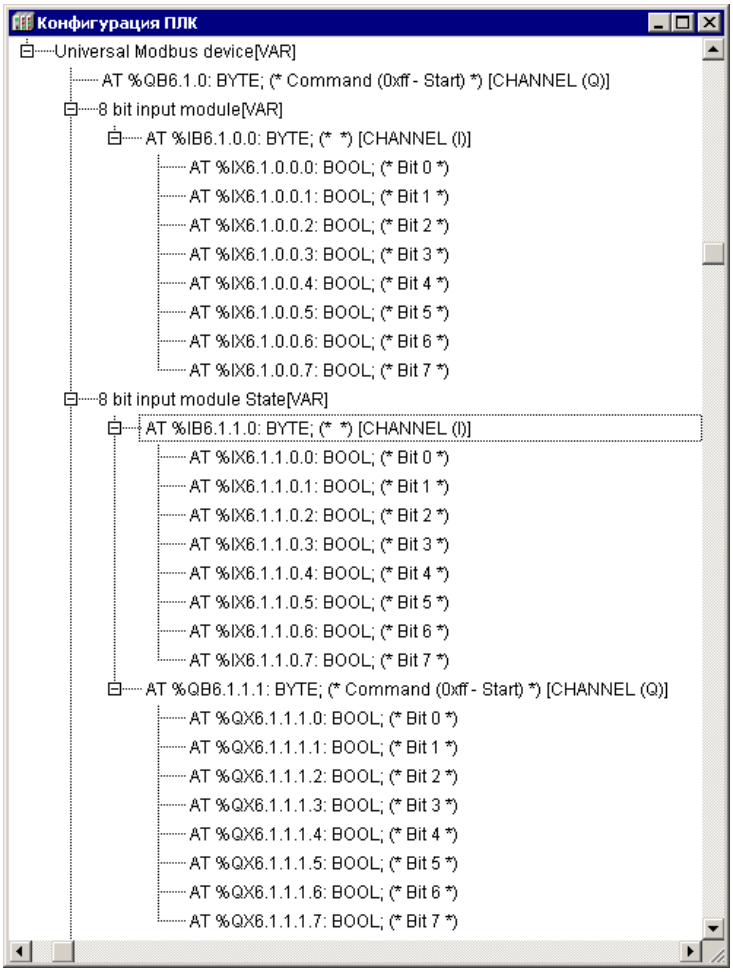


Рисунок 7.23 – Канал ввода и канал ввода типа «State»

Каналы ввода/вывода подмодуля UMD имеют следующие параметры:

- **Command (Номер команды протокола)** – задает номер команды (номер функции) протокола, по которой будет производиться обмен. Команда для обмена определяется Slave устройством и описана в документации на него. При конфигурировании требуемая команда выбирается из списка.
- **Register Address (Адрес регистра Slave-устройства)** – устанавливают адрес регистра опрашиваемого устройства. Адрес определяется Slave-устройством, и описан в документации на него. При конфигурировании ПЛК указывается этот адрес.
- **Visibility (Видимость)** – задает видимость параметров модуля в протоколе «Gateway» (в частности, в программе «EasyWorkPLC» разработки ПО «ОВЕН»). Значения выбираются из списка **«yes»** и **«no»,** значение по умолчанию – **«no».**

Внимание!
Особенность ModBus для CODESYS: для любого номера команды все параметры хранятся в одной и той же области памяти, и можно к одному и тому же участку памяти обращаться как к разным переменным разных типов; задавая новые переменные ModBus нужно помнить расположение данных в памяти.

7.4.2 Модуль ModBus (Slave)

Модуль «ModBus (Slave)» используется, когда требуется, чтобы контроллер работал в сети в подчиненном режиме (slave), т.е. отвечал на запросы устройства, работающего в режиме Master. При установке модуля «ModBus (Slave)» выбирается коммуникационный интерфейс для обмена данными с другими устройствами, добавляются и настраиваются требуемые переменные.

Модуль «**ModBus (slave)»** – составной и имеет в своем составе подмодуль **ModBus (FIX)** – см. ниже.

Параметры модуля (см. рисунок 7.24):

- **Name** – символическое имя элемента, см. п. 7.3.1.
- **Address (Адрес устройства)** – задается адрес ПЛК, по которому прибор будет опрашиваться в сети устройством-Мастером (например, панелью оператора). Параметр имеет значения в диапазоне от 1 до 247, значение по умолчанию – 1.
- **Visibility (Видимость)** – задает видимость параметров модуля в протоколе «Gateway» (в частности, в программе «EasyWorkPLC» разработки ПО «ОВЕН»). Значения выбираются из списка **«yes»** и **«no»,** значение по умолчанию – **«no».**

Переменные, которыми будет обмениваться ПЛК по протоколу **ModBus**, выбираются пользователем командой контекстного меню «Append Subelements (добавить подэлемент)».

Внимание!
При случайном отключении питания во время работы ПЛК последние (текущие) значения переменных сохраняются в энергонезависимой памяти и восстанавливаются при возобновлении работы прибора.

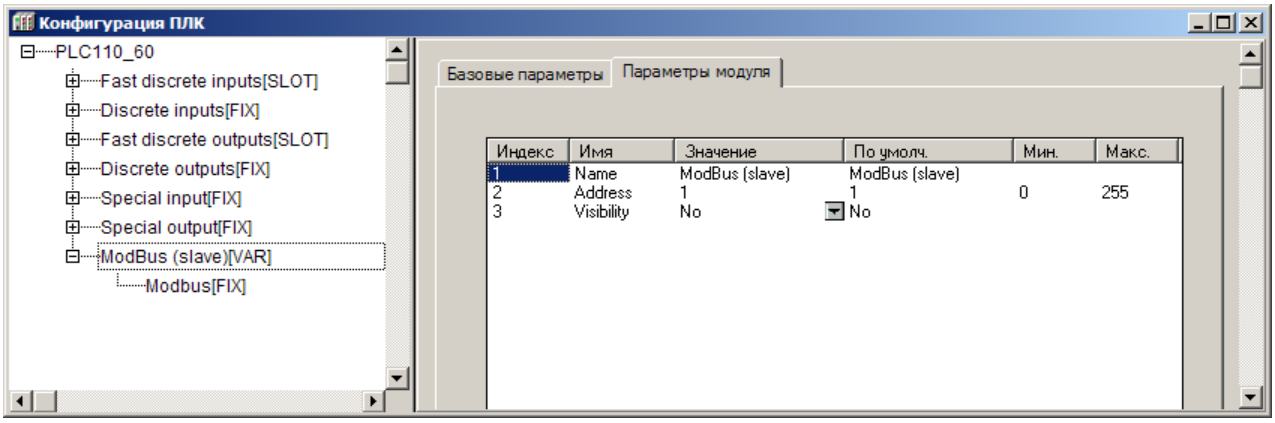


Рисунок 7.24 – Параметры модуля ModBus (Slave)

7.4.2.1 Подмодуль ModBus (FIX). Настройка коммуникационных интерфейсов

При добавлении модуля «**ModBus (slave)**» в конфигурацию ПЛК, в состав модуля уже подключен подмодуль «**ModBus (FIX)»**, к которому, в свою очередь, подключается коммуникационный интерфейс (см. рисунок 7.26).

В ПЛК предусмотрена возможность обмена данными по интерфейсам: **RS-232, RS-485 и TCP (Ethernet).**

Для работы с разными коммуникационными интерфейсами в ПЛК предусмотрены соответствующие подмодули (подэлементы). Подключение подэлементов производится выбором требуемой команды (**Добавить подэлемент | <Имя подэлемента>**) контекстного меню (см. рисунок 7.25).

При работе ПЛК в режиме **«Ведомый (slave)»** возможно использование нескольких разных портов, т.е. опрос может вестись по разным интерфейсам. Таким образом, подключая несколько разных портов, можно один модуль соединить с разными Мастерами по разным физическим линиям (и интерфейсам).

Этот прием может использоваться, например, для создания межсетевого шлюза и/или линии резервного управления (например, подключения SCADA-системы в резервном варианте): например, при выходе из строя порта Ethernet(авария, сбой и пр.), вышестоящая SCADA понимает, что произошел сбой, и начинает информационный обмен с устройством по резервной линии – через COM-порт. Скорость обмена информацией уменьшается, но функционирование устройства продолжается.

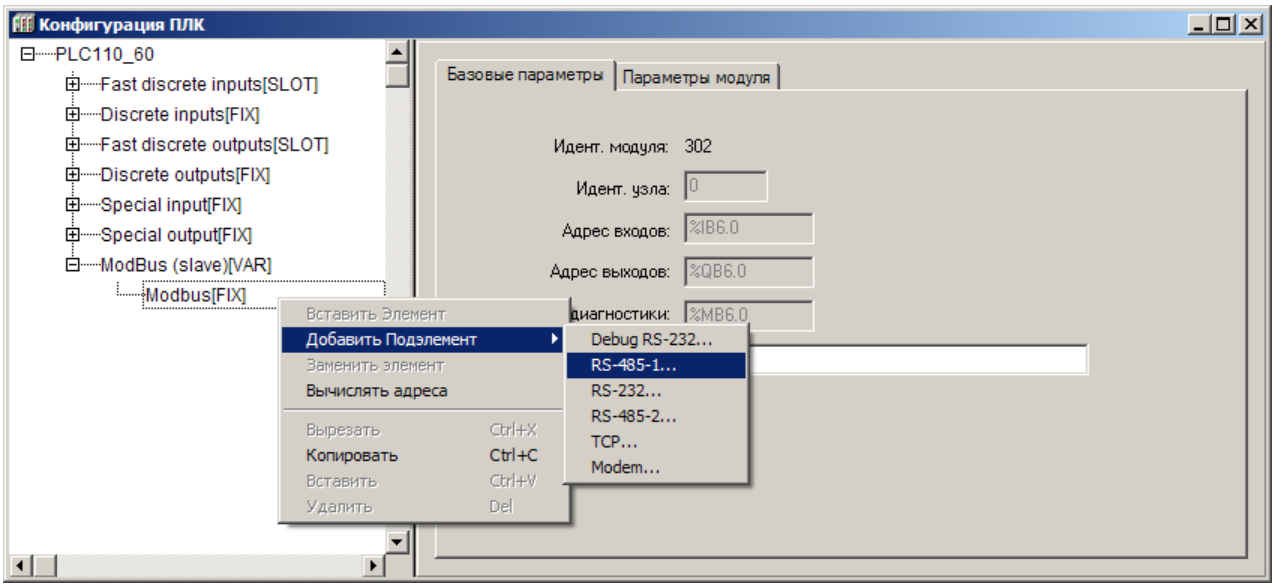


Рисунок 7.25 – Контекстное меню добавления подмодулей настройки коммуникаций

Количество подключаемых портов ограничено лишь конструкцией ПЛК.

Пример подключения нескольких портов представлен на рисунке 7.26.

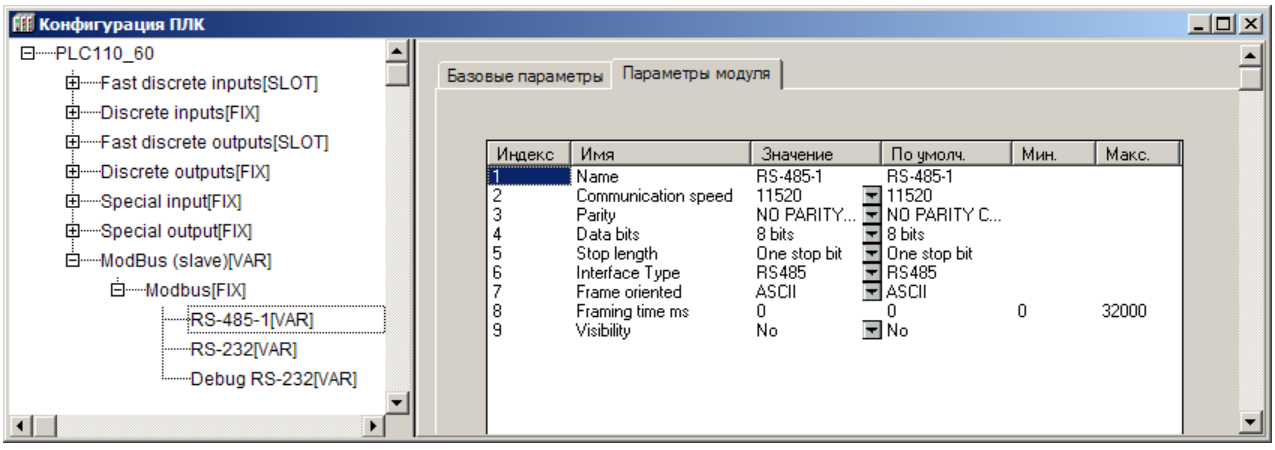


Рисунок 7.26 – Подключение нескольких портов в подмодуле ModBus (FIX)

7.4.2.2 Настройка входов и выходов подмодуля

После задания значений параметров подмодуля «ModBus (Slave)» к нему подключаются каналы (переменные) входа / выхода.

Добавление каналов производится выбором команды «Добавить подэлемент | <Наименование подэлемента>» контекстного меню строки «ModBus (Slave)» (см. рисунок 7.27).

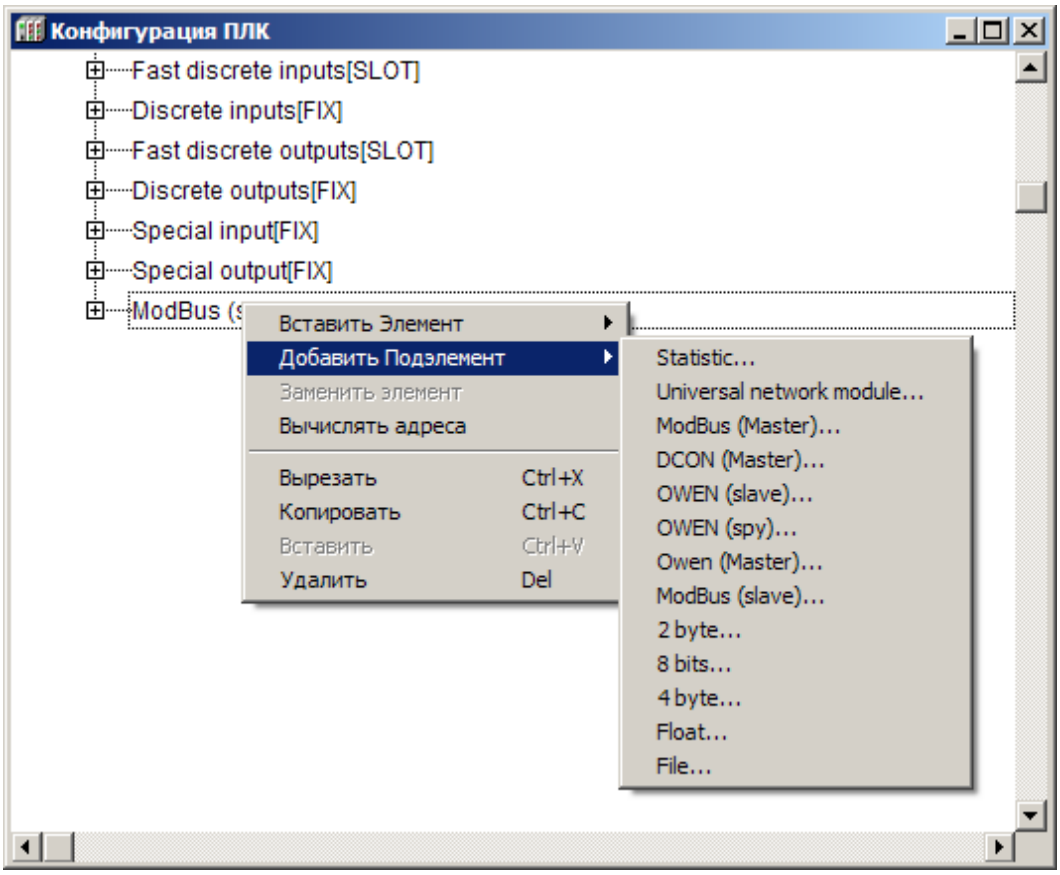


Рисунок 7.27 – Модуль ModBus (Slave). Добавление переменных

Возможные типы каналов (переменных) приведены в таблице 7.2.

Таблица 7.2 – Типы каналов (переменных) модуля «ModBus (Slave)»

Типы передаваемых переменных	Канал	Размер в памяти	Команды считывания	Команды записи
1…8 бит Word, INT, 16 бит Dword, DINT, 32 бита Real Файл	8 bit 2 byte 4 byte Float File	8 бит 2 байта 4 байта 32 бита *)	01, 02 03, 04 03, 04 03,04 20	05 6, 16 16 16 -
*) Об использовании подэлемента «File» – см. раздел 7.4.2.3				

При работе по протоколу ModBus обращение устройства-мастера к используемым переменным производится по адресу переменной в памяти модуля «ModBus (Slave)». Адресация переменных автоматически формируется ПО CODESYS при программировании контроллера, но при этом не отображается в интерфейсе программы. Поэтому для обращения к переменной ее адрес следует

вычислить, учитывая при этом особенности распределения адресов ячеек памяти («выравнивание» переменных).

7.4.2.2.1 Выравнивание адресации

Все используемые переменные хранятся в одном адресном пространстве, и считывать данные из этого пространства можно разными способами. Как известно, переменные бывают следующих типов.

1. Логического (BOOL), они занимают один бит памяти и адресуются как бит с заданным номером в пронумерованном байте.
2. Короткие знаковые и беззнаковые целые, байты (BYTE, SINT, USINT), занимают восемь бит и имеют адрес, увеличивающийся на единицу для каждых следующих восьми бит, т. е., адресуется каждый байт.
3. Знаковые и беззнаковые целые, слова (WORD, INT, UINT), занимают 16 бит и имеют адрес, увеличивающийся на единицу для каждых следующих шестнадцати бит, то есть, адресуется каждые два байта, и началу следующей по адресу ячейке с двухбайтовой переменной соответствует начало байта с адресом, равным адресу слова, умноженному на два.
4. Знаковые и беззнаковые двойные целые, двойные слова (DWORD, DINT, UDINT), занимают 32 бита и имеют адрес, увеличивающийся на единицу для каждых следующих 32 бит, то есть, адресуются каждые четыре байта, и началу следующей по адресу ячейки с четырехбайтовой переменной соответствует байт с адресом, равным адресу двойного слова, умноженному на четыре.

Данный способ адресации наглядно показан в таблице 7.3.

Автоматическая адресация переменных производится последовательно, начиная с нулевого адреса (как для битовых переменных, так и для переменных, передаваемых регистрами).

Таким образом, если в модуле используются переменные одного типа, то при запросе устройством – Мастером регистра с адресом «0», модуль считывает первые два байта, для регистра с адресом «1» – вторые два байта и так далее.

Если переменные имеют длину более двух байт, то при запросе регистра с адресом «0», модуль считывает первые два байта первой переменной, для регистра с адресом «1» – вторые два байта первой переменной и так далее.

Таблица 7.3 – Пример адресации битов и регистров в памяти модуля

Адрес бита	Адрес байта (BYTE, SINT, USINT)	Адрес двухбайтовой переменной (WORD, INT, UINT)	Адрес четырехбайтовой переменной (DWORD, DINT, UDINT)
0… 7	0×0000	0×0000	0x0000
8… 15	0×0001		
16… 23	0×0002	0×0001	
24… 31	0×0003		
32… 39	0×0004	0×0002	0x0001
40… 47	0×0005		
48… 55	0×0006	0×0003	
56… 63	0×0007		

Но если в модуле используются переменные разных типов (например, одновременно восьмибитный, двухбайтный и четырёхбайтный), то при распределении

адресов ПО CODESYS производит «выравнивание» адресов переменных – упорядочение адресов переменных в памяти модуля. Такое упорядочение заключается в организации памяти таким образом, что переменные размером 8 бит, 2 байта и 4 байта располагаются только по определенным адресам: четырёхбайтным переменным присваиваются адреса, кратные 2; двухбайтным – кратные 1. То есть, независимо от порядка задания переменных, выравнивание назначает переменным адреса, кратные их длине.

Таким образом, первая восьмибитная переменная будет расположена в 0… 7 битах памяти модуля, вторая – в 8… 15 и т.д. Если же вторая переменная двухбайтная, она будет располагаться в 16… 31 битах, т.е., по любому (то есть, кратному 1) адресу , и т.д. Четырехбайтная переменная займет следующее свободное место, кратное 2.

Такой порядок размещения переменных в памяти модуля может образовать адресные пространства, не занятые переменными. Эти пространства не отображаются в области ввода/вывода, но они обязательно должны учитываться при организации опроса переменных; учитывать эту особенность следует еще на стадии задания переменных.

Пример «выравнивания» при размещении переменных в памяти модуля приведен в таблице 7.4.

Таблица 7.4 – Пример «выравнивания» адресации переменных в памяти модуля

Адрес регистра	Адрес бита	Адресация переменных
0×00	0… 7	8 бит (1 байт)
	8… 15	↓ Сдвиг выравнивания: пропущенные адреса
0×01	16… 23	2 байта
	24… 31	
0×02	32… 39	8 бит (1 байт)
	40… 47	8 бит (1 байт)
0×03	48… 55	8 бит (1 байт)
	56… 63	↓ Сдвиг выравнивания: пропущенные адреса
0×04	64… 71	4 байта
	72… 79	
0×05	80… 87	
	88… 95	

Внимание!
В некоторых версиях ПО CODESYS 2.3 при подключении модулей «ModBus (Slave)» в конфигурацию ПЛК могут возникнуть ошибки выравнивания адресации.
Например, один и тот же адрес может быть ошибочно присвоен двум каналам разных модулей.
Основной способ исправления ошибки выравнивания – вставить перед неправильно выровненным модулем модуль другого типа. Автоматическое переназначение адресов исправит ошибку.
При возникновении подобной проблемы с выравниванием пространства адресации следует обратиться в группу технической поддержки компании ОВЕН или описать свою проблему на форуме сайта www.owen.ru – технические специалисты компании посоветуют, каким способом исправить ошибку, чтобы она не возникала в дальнейшем.

7.4.2.3 Подэлемент «File». Передача архивных данных

Подэлемент «File» используется для передачи архивных данных (файла) при помощи функции 20 протокола ModBus.

Записанные при помощи модуля **Archiver** (см. раздел 7.4.10) архивные данные (файл) могут быть переданы по последовательному интерфейсу по протоколу ModBus при помощи специализированной функции № 20. Данные могут быть приняты специализированным ПО на ПК или OPC-сервером, поддерживающим работу с этой функцией (например Lectus OPC).

Данные из файла передаются по коммуникационному интерфейсу, настройка которого осуществляется так же, как при передаче обычных данных.

Для реализации передачи архивных файлов следует подключить к модулю «ModBus (Slave)» подмодуль «File». Параметры модуля (см. рисунок 7.28):

- **Name (Имя)** – задает имя файла с архивной информацией, находящегося на Flash-диске ПЛК, внешнем Flash-диске либо на RAM-диске ПЛК, записанного модулем «**Archiver»** (см. п. 7.4.10); расположение файла задается с помощью префикса, так же, как и в модуле работы с файлами (п. 4.2.4.2)**.**
- **Amount Byte (Размер записи)** – определяет размер одной архивной записи в байтах (в запросе записей может быть больше одной).
- **Visibility (Видимость)** – задает видимость параметров модуля в протоколе «Gateway» (в частности, в программе «EasyWorkPLC» разработки ПО «ОВЕН»). Значения выбираются из списка **«yes»** и **«no»,** значение по умолчанию – **«no».**

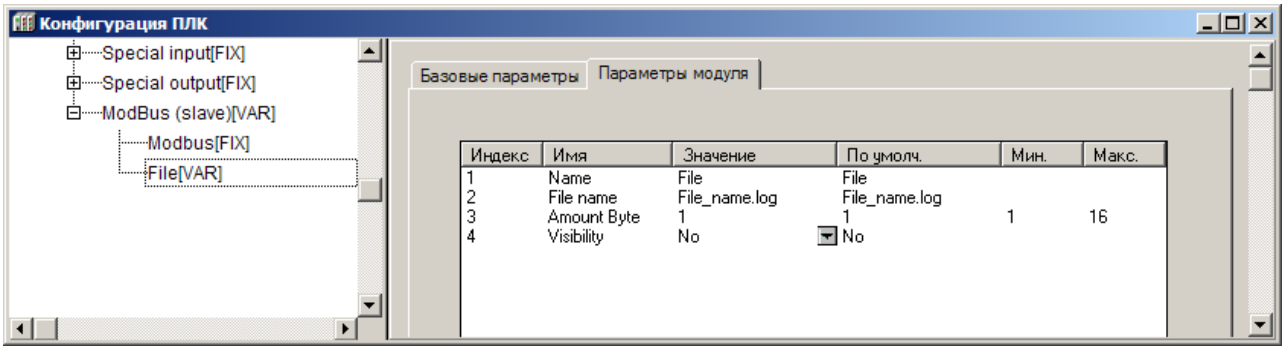


Рисунок 7.28 – Параметры подмодуля «File»

При необходимости работать с несколькими архивными файлами следует последовательно подключить в модуль **ModBus (Slave)** требуемое количество подмодулей **File**, настроив каждый из них. При настройке следует иметь в виду, что в запросе нет возможности передать имя файла; поэтому соответствие имени файла (которых может быть много) и запроса осуществляется следующим образом. Номер файла в запросе соответствует позиции файла в дереве Конфигурации ПЛК (PLC Configurations), начиная с нуля. Таким образом, запрос с нулевым номером файла будет читать данные из первого файла в конфигурации подмодуля, первый – из второго, и так далее. Если запрошен файл, которого нет на диске (или в конфигурации он не указан), то выдается код ошибки с номером 0x04.

Примечание. *Для того чтобы Lectus OPC работал с этой функцией, необходимо дополнительно поместить в рабочую директорию библиотеку ModBus.dll.*

7.4.2.3.1 Форматы запросов и ответов

Информация данного раздела носит справочный характер; она необходима в случае создания собственного программного обеспечения для обмена, запускаемого на ПК. При использовании готового ПО, поддерживающего работу с функцией 20 протокола ModBus особенности обмена скрыты от пользователя.

Формат запросов и ответов приведен в таблице 7.5.

Таблица 7.5 – Формат запросов и ответов

Function Code	0x14	Byte	Код функции
Формат запросов			
Byte count	0x07	Byte	Количество байт, следующих ниже
Referens Type	0x06	Byte	Подфункция (здесь – константа =6)
Hi File number	-	Byte	Старший байт номера требуемого файла
Lo File number	-	Byte	Младший байт номера требуемого файла
Hi Rec addr	-	Byte	Старший байт адреса записи в файле
Lo Rec addr	-	Byte	Младший байт адреса записи в файле
Hi Rec num	-	Byte	Старший байт количества запрашиваемых записей
Lo Rec num	-	Byte	Младший байт количества запрашиваемых записей
Формат ответов			
Byte count	0x07	Byte	Количество байт, следующих ниже
Byte count	0x07	Byte	Количество байт, следующих ниже (необходимо по стандарту)
Referens Type	0x06	Byte	Подфункция (здесь – константа = 6)
Data	-	Byte*Rec_num* Amount_byte	Собственно данные длиной Rec num из (запроса), умноженные на длину одной записи из конфигурации.
Примечания. 1) File number – может принимать значение от 0x0000 до 0xffff 2) Rec number – может принимать значение от 0x0000до 0xffff 3) Rec addr – может принимать значение 0x0000-0xfffe 4) При ошибке возвращается стандартный код ошибки ModBus: 0x02 5) При запросе с адресом Recaddr=0xffff происходит удаление файла.			

7.4.3 Модуль «DCON (Master)»

Модуль «DCON (Master)» используется, когда требуется, чтобы контроллер работал по протоколу DCON в режиме Мастера сети, т.е. опрашивал и контролировал другие устройства, работающие в сети в подчиненном режиме (slave). При работе по протоколу DCON ПЛК может функционировать только в режиме Мастера сети

При установке модуля «DCON (Master)» выбирается коммуникационный интерфейс для обмена данными с другими устройствами, добавляются и настраиваются требуемые переменные.

Модуль имеет один параметр: **«Visibility (Видимость)»**, который задает видимость параметров модуля в протоколе «Gateway» (в частности, в программе «Easy-WorkPLC» разработки ПО «ОВЕН»). Значения выбираются из списка **«yes»** и **«no»,** значение по умолчанию – **«no»**.

При опросе модулем «DCON (Master)» подчиненных устройств информация о ходе обмена записывается в канал модуля:

– **Last Error** – код ошибки. В переменной отображается код ошибки, если информационный обмен прошел неудачно. Это необходимо для корректности работы опрашиваемого устройства. Коды ошибок данного модуля представлены в Приложении В.

7.4.3.1 Настройка коммуникационных интерфейсов «DCON (Master)»

При добавлении модуля «DCON (Master)» в конфигурацию ПЛК в состав модуля уже подключен порт **Debug RS-232** (см. рисунок 7.29). При необходимости работы через другой коммуникационный интерфейс этот порт можно заменить требуемым последовательным портом или модемом (для этого следует выбрать команду «Заменить элемент | <Элемент>» контекстного меню строки «Debug RS-232»).

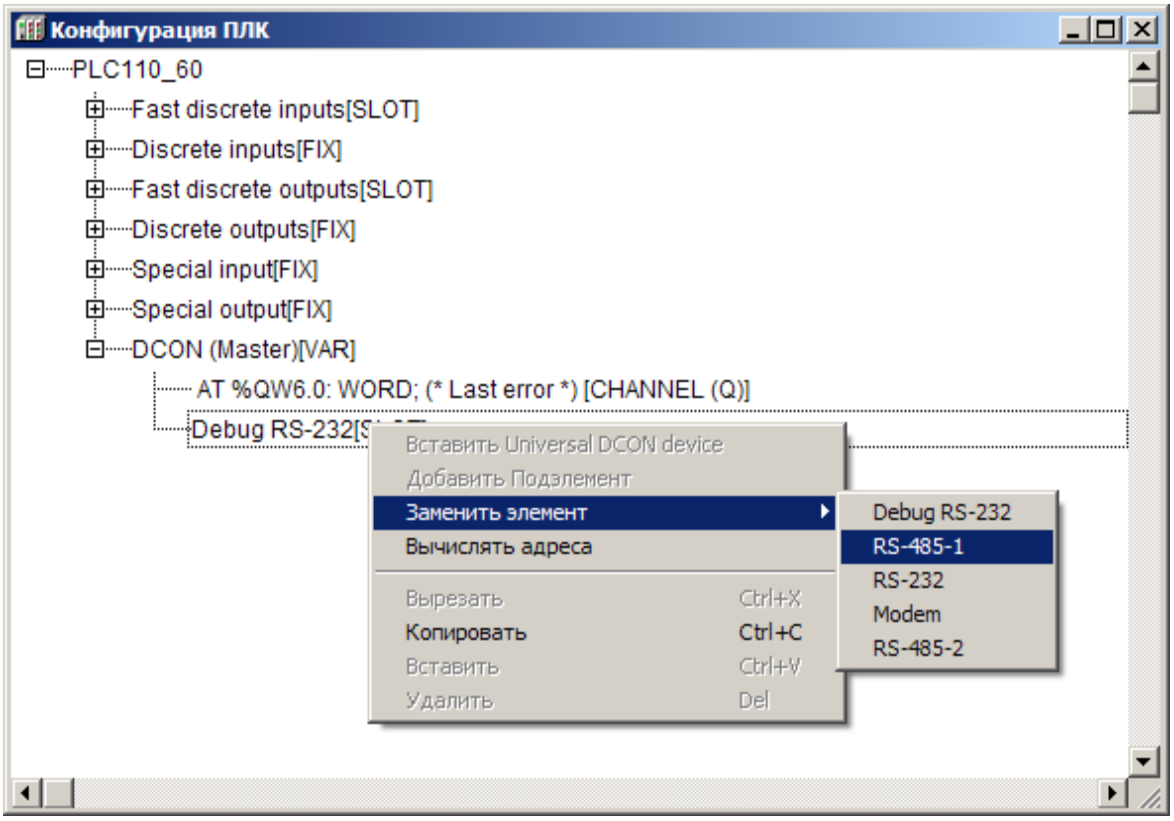


Рисунок 7.29 – Выбор коммуникационных интерфейсов «DCON (Master)»

7.4.3.2 Подмодуль Universal DCON Device

Для добавления в список опроса, проводимого мастером сети, устройств, работающих в режиме DCON (Slave) следует в модуле DCON (Master) добавить подмодуль «Universal DCON Device» (универсальное устройство DCON). Для добавления в список нескольких опрашиваемых устройств эту процедуру следует повторить столько раз, сколько устройств должно быть подключено: для каждого устройства должен быть добавлен соответствующий ему модуль Universal DCON Device с индивидуальными настройками.

Чтобы добавить подмодуль «Universal DCON Device» в конфигурацию, следует выбрать команду «Добавить подэлемент | Universal DCON Device» контекстного меню строки «ModBus (Master)».

Опрос добавленных «Universal DCON device» производится последовательно, в порядке следования в конфигурации (если другой порядок не задан отдельно, в настройках модуля).

«Universal DCON device» имеет один канал: **Status**:

- если в него записывается значение **0x00FF**, то происходит старт работы данного устройства DCON;
- если модуль уже запущен, то повторная запись в канал значения **0x00FF** приводит к внеочередному запросу одной очередной переменной устройства DCON;
- если в канал записано значение **0x00FE,** то происходит его остановка и прекращение всех посылок в сеть;
- если в канал ничего не записано, то опрос устройства производится автоматически, в порядке очереди.

При необходимости устройство можно исключить из списка опроса, подав соответствующую команду в канал Start\Stop.

При добавлении модуля «Universal DCON Device» его параметры и идентификаторы не привязаны к конкретному внешнему устройству (модулю ввода-вывода, операторской панели). Конкретный вариант внешнего устройства производится заданием значений параметров модуля (конфигурированием модуля), см. рисунок 7.30.

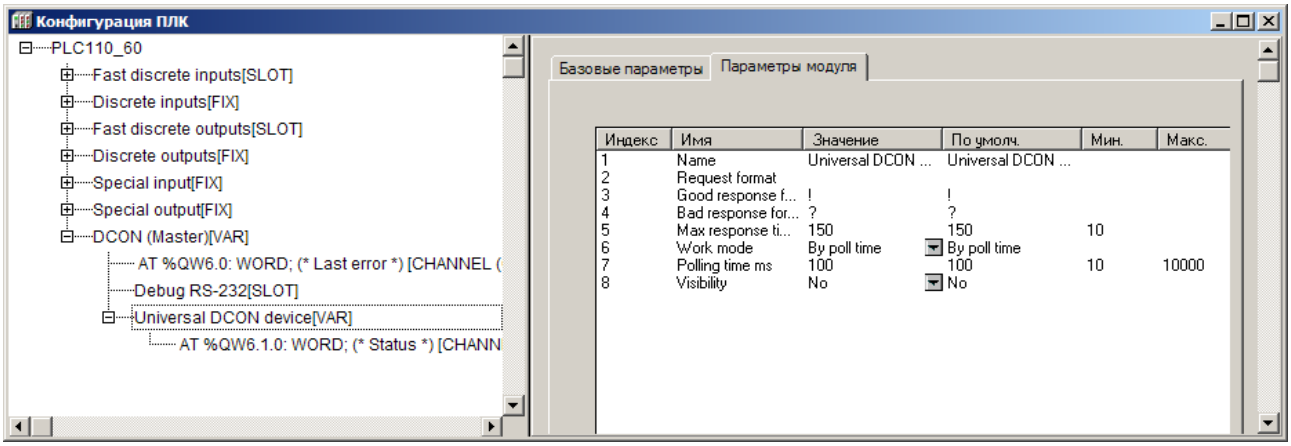


Рисунок 7.30 – Параметры подмодуля Universal DCONdevice

Параметры подмодуля «UniversalDCONdevice»:
Параметры универсального устройства DCON (Master), см. рисунок 7.32:

- **Name** – символическое имя элемента, см. п. 7.3.1.
- **Request format (Формат запроса) –** формат запроса, может быть любым, ограничения не накладываются (см. раздел 7.4.3.2.1).
- **Good response format (Формат правильного ответа)** – формат правильного ответа, значение по умолчанию – **«!»**(см. раздел 7.4.3.2.1)**.**
- **Bad response format (Формат неправильного ответа)** – формат неправильного ответа, значение по умолчанию – **«?»**(см. раздел 7.4.3.2.1)**.**
- **Max response timeout (Максимальное время ответа)** – задает время, за которое опрашиваемый прибор должен ответить на запрос **DCON (Master)**. Если в течение этого времени прибор не отвечает, то считается, что он отключен или произошел обрыв линии связи. Информация об этом заносится в переменную **«Код последней ошибки» (Last error).** Значение сверху не ограничено, может быть любым, в т.ч. дробным, но не меньше 10 мс, значение по умолчанию – 150 мс.

– **Work mode (Режим работы)** – задает режим работы модуля **«DCON (Master)»** при опросе внешних устройств. Значения выбираются из списка (значение по умолчанию – «By poll time»):

  By poll time – «по времени»– контролируемые устройства опрашиваются с периодичностью, заданной в параметре **«Период опроса устройства» (Polling time)**;

  By value change – «по изменению значения переменных»– модуль **DCON (Master)** генерирует запрос устройству при изменении значений выходных переменных модуля;

  Both – «оба варианта» – опрос производится с временным интервалом, заданным в параметре **Polling time** и тогда, когда изменяются значения выходных переменных;

  By command – «по команде»– производится однократная посылка запроса, когда в канал **Статус (Status)** модуля **Универсальное устройство DOCN** записывается значение **0x00FF**.

– **Polling time (Период опроса устройства, в мс)** – диапазон значений от 10 до 10000, значение по умолчанию – 100 (см. п. 3.2.1.6).
– **Visibility (Видимость)** – видимость параметров модуля в протоколе «Gateway» (в частности, в программе «EasyWorkPLC» разработки ПО «ОВЕН»). Значения выбираются из списка **«yes»** и **«no»,** значение по умолчанию – **«no»**.

Примечание. *В Мастере, когда он работает в режиме* ***«По изменению значения переменных»*** *или* ***«По команде»****, нельзя ставить значение параметра* ***Polling time*** *слишком маленьким. По умолчанию его значение 100 мс. Однако, если на реальном проекте будет замечено, что Мастер при загрузке программы или при* ***Login*** *формирует лишние пакеты и/или запросы, которых не должно быть, значение параметра увеличивают (до 200, 300 и т.д.) до предотвращения появления ложных пакетов.*

7.4.3.2.1 Параметры «Формат…»

В протоколе **DCON** при организации опроса устройств создается строка запроса. При ее посылке опрашиваемое устройство может вернуть два варианта ответа: ответ правильный (команда распознана, данные есть) – один формат, и ответ неправильный (не распознана команда, нет данных и/или пр.) – другой формат.

Строки «Формат запроса» (Request format), «Формат правильного ответа» (Good response format), «Формат неправильного ответа» (Bad response format) используются для задания формата запроса DCON (Master) и разбора правильного/неправильного ответа.

Строки формата ответа могут не задаваться, если устройство не отвечает на запрос.

Строка формата представляет собой строку, содержащую символы и спецкоманды.

– **Символы** – любой символ, кроме служебных, к которым относятся символы **«$»** (знак доллара), **«[»** и **«]»** (открывающая и закрывающая квадратные скобки).
  Примечание. При необходимости вывести служебный символ в качестве обычного, он вводится в строку два раза подряд.
– **Спецкоманда** имеет формат: **[{модификатор} действие]**.

  Модификатор – количество символов, обрабатываемых действием. Представляет собой десятичное целое число. Может быть у всех дей-

ствий, кроме вычисления контрольных сумм. Наличие модификатора необязательно, значение по умолчанию = 1.

Действие – отображается в строке спецкоманды одним из символов – **D, H, F, S, *, +, %**. Регистр символов значения не имеет.

Символы соответствуют следующим видам действий:

D – представляет передаваемую переменную в ASCII-символах в десятичном формате (без знака) или преобразует ASCII-строку из десятичного формата (без знака) в принимаемую переменную. Количество символов задается модификатором.

H – представляет передаваемую переменную в ASCII-символах в шестнадцатиричном формате или преобразует ASCII-строку из шестнадцатиричного формата в принимаемую переменную. Количество символов задается модификатором.

F – представляет передаваемую переменную в ASCII-символах в десятичном формате со знаком, разделителем целой и дробной части числа (точкой). Строка имеет фиксированное число символов, заданное модификатором. Для принимаемых переменных производит обратное преобразование из ASCII-строки в число.

S – осуществляет прямое копирование из передаваемой строковой переменной в строку запроса числа символов, заданного **модификатором** или обратное копирование из строки ответа в принимаемую переменную строкового типа.

***** – задает в строке ответа набор символов, которые надо пропустить. Количество символов может быть задано модификатором.

+ – вставляет в строку запроса контрольную сумму или получает ее в строке ответа. Контрольная сумма вычисляется путем сложения с переполнением по модулю 256. Данное действие не может иметь модификатора.

% – вставляет в строку запроса контрольную сумму или получает ее в строке ответа. Контрольная сумма вычисляется по 8-ми битному полиному (DOW-CRC). Данное действие не может иметь модификатора**.**

Примечание. *При работе ПЛК по протоколу DCON есть три варианта работы: без расчета контрольных сумм, с расчетом контрольных сумм путем сложения значений всех символов и с расчетом контрольных сумм 8-ми битовых. Вариант работы пользователь выбирает в соответствии с тем, какой вариант расчета контрольной суммы используется в опрашиваемом приборе.*

Используются следующие алгоритмы преобразования:

- **при формировании запроса –** все символы вне спецкоманд копируются в строку запроса без изменения, спецкоманды заменяются на значения передаваемых (выходных) переменных. Значения переменных кодируются в формате, заданном **действием**, число символов соответствует **модификатору**;
- **при разборе ответа –** все символы вне спецкоманд сравниваются с соответствующими позициями ответа и, при нахождении различия, вырабатывается сообщение об ошибке. Данные в позициях ответа, соответствующих спецкомандам, преобразуются и сохраняются в соответствующих принимаемых (входных) переменных.

Если запрос жестко фиксированный, т.е. в строке не содержатся изменяемые данные, то строка набивается без каких-либо команд, в таком виде отсылается; при этом может быть добавлена контрольная сумма.

Аналогично с ответом: если приходит строка, не содержащая каких-либо данных (в конце может быть контрольная сумма), это означает, что прибор работает, реагирует и факт получения ответа от прибора уже является информацией.

7.4.3.2.2 Настройка входов и выходов подмодуля

После задания значений параметров подмодуля «Universal DCON Device» к нему требуется подключить каналы, задающие входные параметры (параметры, значение которых Мастер запрашивает у Slave-устройств) и выходные параметры (значения, которые Мастер передает – записывает – в Slave устройства) подмодуля.

Добавление каналов производится выбором команды «Добавить подэлемент | <Наименование подэлемента>» контекстного меню строки «Universal DCON Device».

В «Universl DCON Device» могут быть добавлены каналы, перечисленные в таблице 7.6.

Каналы (переменные) могут принадлежать следующим типам: REAL (32 бит, целое беззнаковое, с плавающей точкой или строка), STRING (16 байтовая строка), 8, 16 или 32 бит.

Таблица 7.6 – Типы каналов модуля «Universl DCON Device»

Канал	Размер в памяти	Направление пересылки данных
8-bit input 16-bit input 32-bit input Float input String input	8 бит 16 бит 32 бита 32 бита 16 байт (строка)	Считывает
8-bit output 16-bit output 32-bit output Float output String output	8 бит 16 бит 32 бита 32 бита 16 байт (строка)	Записывает

Тип и порядок расположения входных и выходных переменных в модуле должны соответствовать строкам команд в полях **«Request format»** и **«Good response format»** (см. раздел 7.4.3.2).

Примеры настройки модуля **DCON (Master)** для опроса устройств ввода/вывода представлены в Приложении Г.

7.4.4 Модуль «Owen (Master)»

Модуль Owen (Master) используется, когда требуется, чтобы контроллер работал в режиме Мастера сети, т.е. опрашивал и контролировал другие устройства, работающие в сети по протоколу Owen в подчиненном режиме (slave) – например, модули ввода-вывода, панели оператора, частотные преобразователи и т.д. (протокол предназначен для описания процесса обмена информацией между приборами фирмы «Овен» и между приборами и ПК на базе сети RS-485).

Примечание. *При случайном отключении питания в процессе работы ПЛК последние (текущие) значения переменных сохраняются в энергонезависимой памяти и восстанавливаются при возобновлении работы прибора.*

При установке модуля «Owen (Master)» выбирается коммуникационный интерфейс для обмена данными с другими устройствами, добавляются и настраиваются требуемые переменные.

Параметры модуля (см. рисунок 7.31):

- **Name** – символическое имя элемента, см. п. 7.3.1.
- **Max response delay, ms (Максимальное время задержки ответа, мс) –** задает время, за которое опрашиваемый прибор должен ответить на запрос Owen (Master). Если он не отвечает, то считается, что прибор отключен или произошел обрыв линии связи. Диапазон значений от 0 до 32000, значение по умолчанию – 50.
- **Visibility (Видимость)** – задает видимость параметров модуля в протоколе «Gateway» (в частности, в программе «EasyWorkPLC» разработки ПО «ОВЕН»). Значения выбираются из списка **«yes»** и **«no»,** значение по умолчанию – **«no».**

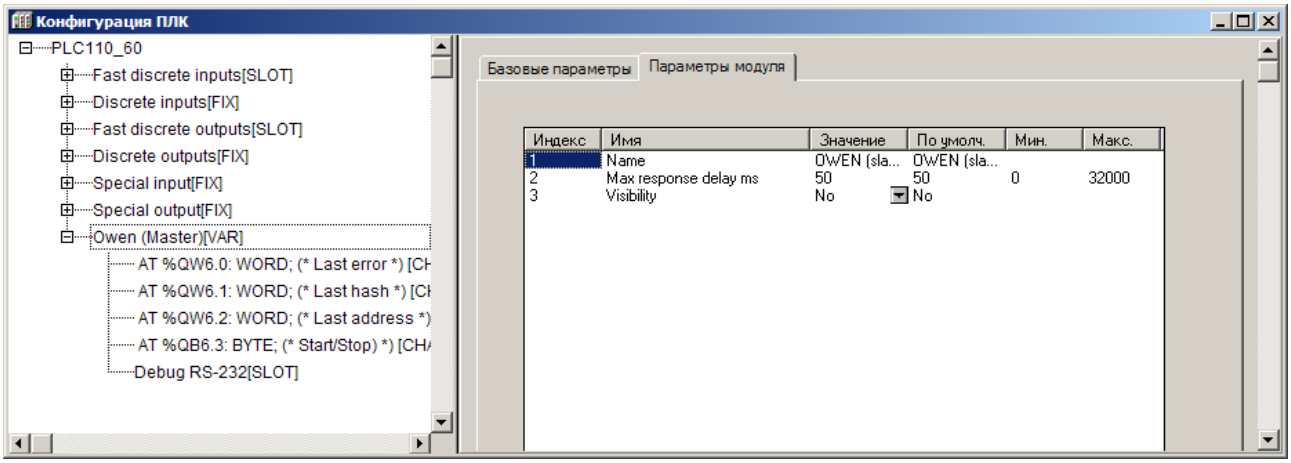


Рисунок 7.31 – Каналы и параметры модуля Owen (Master)

При опросе модулем «Owen (Master)» подчиненных устройств информация о ходе обмена записывается в соответствующих каналах его переменных.

Каналы модуля:

- **Last address (Последний адрес)** – адрес последнего прибора, по которому обращался Owen (Master). При использовании 8 бит адреса – 8-битный адрес при отображении умножается на 2^3 (т.е. на 8), при 11 - не умножается.
- **Last error (Код последней ошибки) –** код ошибки, которая произошла при последнем опросе.
- **Last Hash (Последний Hash-код)–Hash-**код параметра, который фигурировал в последнем опросе.
- **Start/Stop (Старт/Стоп)–** используется для управления включением/выключением работы модуля мастера: если в канал записывается значение **0x00FF**, то происходит старт работы модуля, если же в канал записано значение **0x00FE,** то происходит его остановка и прекращение всех посылок в сеть. Если модуль уже запущен, то повторная запись в канал значения **0x00FF** приводит к внеочередному запросу очередной переменной протокола ОВЕН.

Коды ошибок работы ПЛК и пояснения к ним представлены в Приложении В.

7.4.4.1 Настройка коммуникационных интерфейсов «OWEN(Master)»

При добавлении модуля «**OWEN (Master)**» в конфигурацию ПЛК в состав модуля уже подключен порт **Debug RS-232** (см. рисунок 7.32). При необходимости работы через другой коммуникационный интерфейс этот порт можно заменить требуемым последовательным портом или модемом (для этого следует выбрать команду «Заменить элемент | <Элемент>» контекстного меню строки «Debug RS-232»).

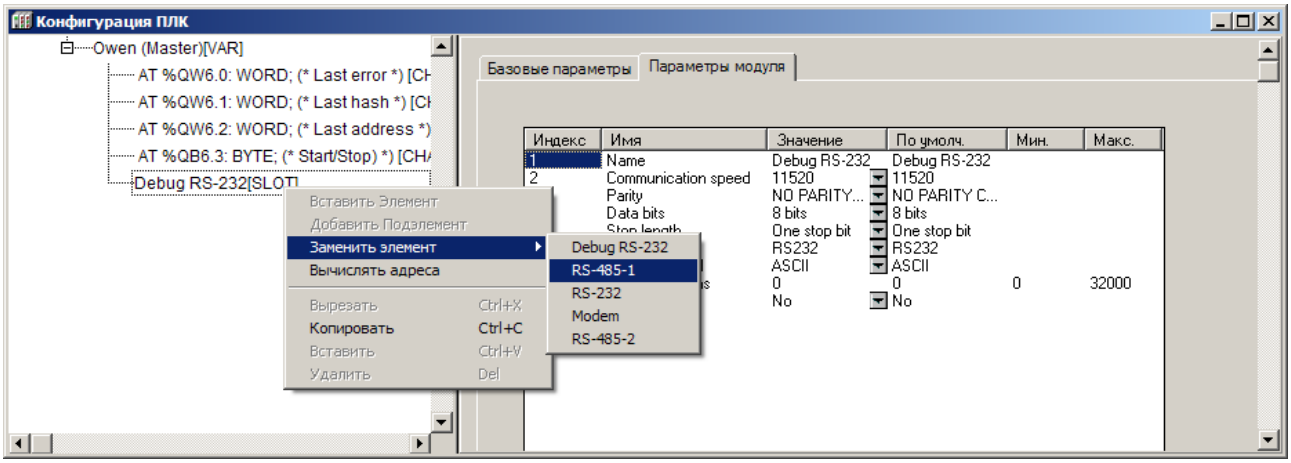


Рисунок 7.32 – Выбор коммуникационных интерфейсов «OWEN (Master)»

7.4.4.2 Настройка входов и выходов подмодуля

После задания значений параметров подмодуля «OWEN (Master)» к нему требуется подключить каналы, задающие входные параметры (параметры, значение которых Мастер запрашивает у Slave-устройств) и выходные параметры (значения, которые Мастер передает – записывает – в Slave устройства) подмодуля.

Добавление каналов производится выбором команды «Добавить подэлемент | <Наименование подэлемента>» контекстного меню строки «OWEN (Master)» (см. рисунок 7.33).

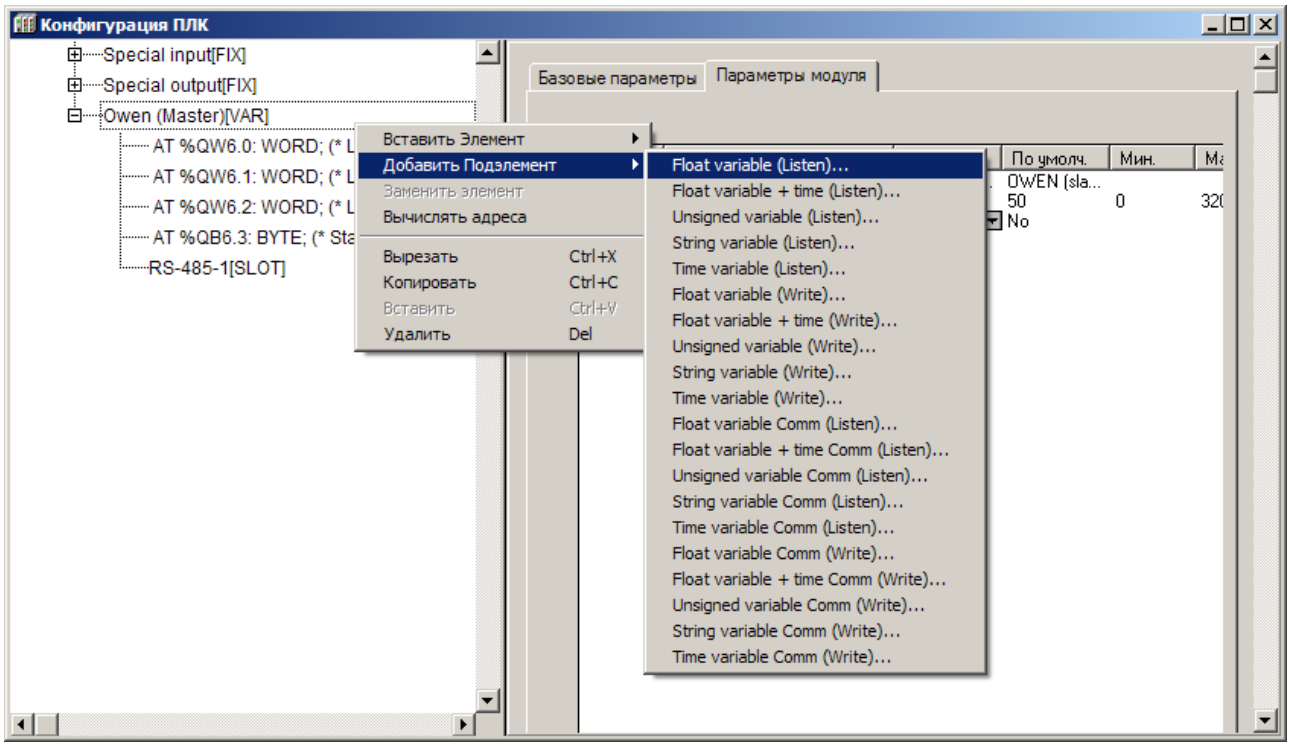


Рисунок 7.33 – Добавление каналов ввода/вывода модуля «OWEN (Master)»

При добавлении канала в «OWEN (Master)» в дерево конфигурации добавляется подкаталог, содержащий внутри себя канал, в котором и отображаются полученное / передаваемое по сети значение (см. рисунок 7.34).

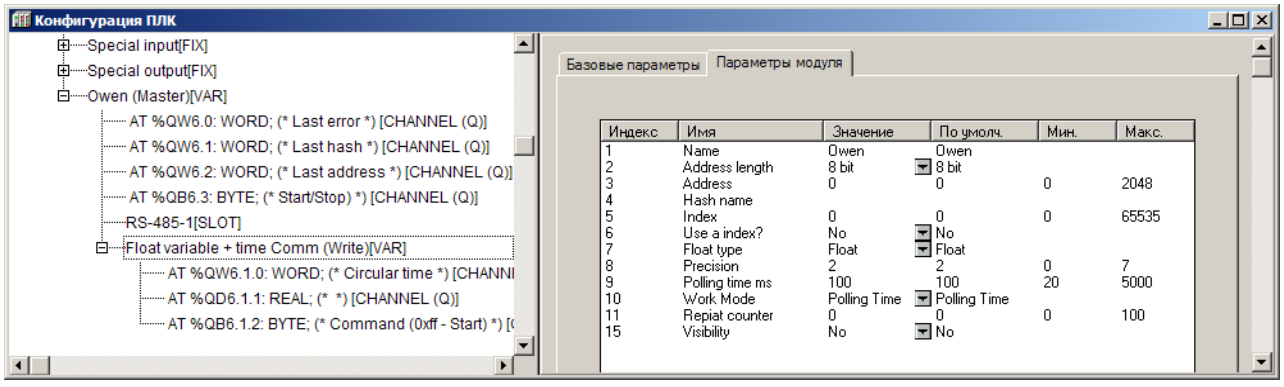


Рисунок 7.34 – Каналы и параметры переменной

В модуле **Owen (Master)** могут быть использованы переменные четырех типов:

1) Переменные, предназначенные для чтения (обозначаемые «**Listen»**).
2) Переменные, предназначенные для записи (обозначаемые «**Write»**).
3) Переменные, предназначенные для чтения (обозначаемые «**Listen»**), имеющие дополнительный командный управляющий канал (обозначаемые «**Comm**»).
4) Переменные, предназначенные для записи (обозначаемые «**Write»**), имеющие дополнительный командный управляющий канал (обозначаемые «**Comm**»).

Переменные каждого типа, различаются по типу данных (всего – шесть типов):

- **Float variable** – число с плавающей точкой,
- **Float variable + time** – число с плавающей точкой с модификатором времени;
- **Unsigned variable** – целочисленная переменная;
- **Unsigned variable + time** – целочисленная переменная с модификатором времени;
- **String variable** – строковая переменная; максимальная длина – 15 символов, в соответствии со стандартом протокола ОВЕН;
- **String variable + time** – строковая переменная с модификатором времени;
- **Time variable** – позволяет передать время; в протоколе ОВЕН при передаче времени данные имеют следующий формат: «**год:месяц:день:час:минута:секунда:миллисекунда»**.

7.4.4.2.1 Особенности переменных типа Unsigned

Переменные типа «Unsigned» позволяют пользователю передавать любые данные в любом произвольном (в т.ч., собственном специализированном) формате. В переменную типа «Unsigned» командой «Добавить подэлемент (Insert element)» контекстного меню вставляются переменные длиной 1, 2, либо 4 байта. Допускается вставить не более 4-х переменных суммарной длинной не более 16 байт.

7.4.4.2.2 Параметры переменных протокола ОВЕН

Параметры переменных протокола ОВЕН, общие для всех типов переменных, (см. рисунок 7.39) таковы:

- **Name** – символическое имя элемента, см. п. 7.3.1.
- **Address Length (Длина адреса устройства) –** значения выбираются из списка **«8 bit»** и **«11 bit»**, значение по умолчанию – **«8 bit».**
- **Address (Адрес устройства)** – диапазон значений от 0 до 255 или от 0 до 2048, в зависимости от размера адреса, значение по умолчанию – 0.
- **Hash name (Сетевое имя переменной)** – указывается сетевое имя переменной. Имена переменных ведомых приборов указываются в руководствах по эксплуатации этих приборов. Вводимое имя преобразуется в ПЛК в **Hash-**код, который используется при обмене по сети RS-485.
- **Index (Индекс прибора)** – задает индекс прибора; в параметре **«Use a index? (Использовать индекс?)»** задают использование индекса. В совокупности параметры применяются для управления конфигурационными параметрами ПЛК, определяют наличие линейного индекса у параметра и задают значение индекса. Диапазон значений от 0 до 65535, значение по умолчанию – 0.
- **Use a index? (Использовать индекс?)** – значения выбираются из списка **«yes»** и **«no»**, значение по умолчанию – **«no».**
- **Polling time, ms (Период опроса устройства, мс)** – диапазон значений – от 20 до 5000, значение по умолчанию – 100.
  Примечание. Следует учитывать физические ограничения сети: скорость информационного обмена в сети ограничена, и, если задается большое количество переменных, значения которых часто запрашиваются, то информация будет поступать к прибору с запаздыванием. Поэтому требуется заранее просчитать пропускную способность сети, и, соответственно, либо уменьшить частоту опроса, либо производить опрос по разным линиям. Поэтому при работе в режимах «Value change (По изменению значения переменных)» и «By Command (По команде)» (режим задается значением параметра «Work mode (Режим работы)», см. ниже), не следует задавать

значение параметра «Polling time» слишком маленьким. По умолчанию его значение 100 мс. Если на реальном проекте будет замечено, что модуль «Owen (Master)» при загрузке программы или при выборе команды «Login» формирует лишние пакеты и/или запросы, то значение параметра следует увеличить (до 200, 300 и более мс), вплоть до прекращения появления лишних пакетов.

- **Work mode (Режим работы)** – задается режим работы модуля «Owen (Master)» при опросе внешних устройств (для переменных различных типов список допустимых значений различен, см. примечание ниже):

  Polling time (По времени)– контролируемые устройства опрашиваются с периодичностью, заданной в параметре **«Polling time (Период опроса устройства)»**;

  Value change (По изменению значения переменных) – модуль Owen (Master) генерирует запрос устройству при изменении значений выходных переменных модуля;

  Both (Оба варианта» – опрос производится с временным интервалом, заданным в параметре «Polling time» и тогда, когда изменяются значения выходных переменных;

  By Command (По команде) – производится однократная посылка запроса, когда в командный канал («Command») переменной, имеющей такой канал, записывается значение **0x00FF**.

  Примечания

  1) Для считываемых переменных (тип «Listen») доступен только режим «Polling time (По времени)».

  2) Для записываемых переменных (тип «Write») доступны режимы «Polling time (По времени)», «Value change (По изменению значения переменных)» и «Both (По времени и по изменению значения переменных)».

  *3) Для переменных с командным каналом (тип «Command») доступен режим «Command (По команде)». В этом режиме управление осуществляется следующим образом: первая посылка значения **0x00FF** в командный канал включает функционирование этой переменной, повторная посылка значения **0x00FF** инициирует проведение опроса. Аналогично опрос инициируется для переменных с командным каналом при работе в других режимах. При посылке в командный канал значения **0x00FE** переменная выключается из цикла опроса мастера.*

- **Repiat Counter** – задает число повторов запроса при ошибке связи.
- **Visibility (Видимость)** – задает видимость параметров модуля в протоколе «Gateway» (в частности, в программе «EasyWorkPLC» разработки ПО «ОВЕН»). Значения выбираются из списка **«yes»** и **«no»,** значение по умолчанию – **«no».**

Для переменных типов «Float variable» (число с плавающей точкой), «Float variable + time» (число с плавающей точкой с модификатором времени) всех модификаций («Listen» – предназначенные для чтения, «Write» – предназначенные для записи, «Command» – имеющие дополнительный командный управляющий канал) – задаются параметры «Float Type (Тип числа с плавающей точкой)», который уточняет вид переменной типа Float и «Precision», который задает точность (определяет положение десятичной точки):

- **Float Type** (тип числа с плавающей точкой) – значение выбирается пользователем из списка (значение по умолчанию – «Float»):

Float – число с плавающей точкой в формате IEEE, обычно используемое в программировании, в **CODESYS** называется Real, имеет длину 4 байта;

Float-Pic – переменная размером в 3 байта, и один байт из мантиссы удаляется, т.е. число с меньшей точностью, но и размер переменной (количество байт) меньше;

Fix point binary – число с фиксированной точкой в двоичном виде; например, число 3 будет записано в виде «0011». Положение десятичной точки для параметров с фиксированной точкой задается параметром «**Precision (Точность)».**

Fix point BCD –число с фиксированной точкой в двоично-десятичном виде (двоично-десятичный код – форма записи целых чисел, когда каждый десятичный разряд числа записывается в виде его четырёхбитного двоичного кода. Например, число 311 будет записано в виде «0011 0001 0001». Положение десятичной точки для параметров с фиксированной точкой задается параметром «**Precision (Точность)».**

– **Precision (Точность)** – задает точность (определяет положение десятичной точки) для параметров с фиксированной точкой: если выбирается значение точности «2», то число 10,12 так и будет передано. При значении «1» – 10,1, при значении «3» – 10,120. Диапазон значений – от 0 до 7, значение по умолчанию – 2.

Отличие переменной **Float** с модификатором времени состоит в том, что, при тех же параметрах, в поле данных, кроме собственно значения, присутствует еще и время (в сотых долях секунды).

Примеры задания параметров для различных случаев применения ПЛК представлены в Приложении Б.

7.4.5 Модуль «Owen (Slave)»

Модуль «OWEN (Slave)» используется, когда требуется, чтобы контроллер работал в сети в подчиненном режиме (slave), т.е. отвечал на запросы устройства, работающего в режиме Master. При установке модуля «OWEN (Slave)» выбирается коммуникационный интерфейс для обмена данными с другими устройствами, добавляются и настраиваются требуемые переменные. Модуль «OWEN (Slave)» обеспечивает обмен информацией по протоколу ОВЕН (протокол предназначен для описания процесса обмена информацией между приборами фирмы «Овен» и между приборами и ПК на базе сети RS-485).

Примечание. *При случайном отключении питания в процессе работы ПЛК последние (текущие) значения переменных сохраняются в энергонезависимой памяти и восстанавливаются при возобновлении работы прибора.*

Модуль «OWEN (Slave)» – составной и имеет в своем составе подмодуль «**OWEN (FIX)»** – см. ниже.

Параметры модуля (см. рисунок 7.35):

– **Name** – символическое имя элемента, см. п. 7.3.1.
– **Slave Name (Имя прибора)** – задает имя ПЛК в сети ОВЕН. Значение по умолчанию – **«max 8 sym»** (т.е. максимально 8 символов).
– **Address Length (Длина адреса устройства)** – задает длину адреса в битах. Приборы разработки ПО «ОВЕН» поддерживают два варианта адресов – 8-ми и 11-ти битовые (задается тот же вариант адреса, что выставлен на управляющей стороне: значение длины адреса Master и Slave-

устройств должны совпадать). Значения выбираются из списка **«8 bit»** и **«11 bit»**, значение по умолчанию – **«8 bit».**

- **Address (Адрес устройства)** – адрес прибора, по которому посылаются запросы. Параметр имеет значения в диапазоне от 0 до 2048 значение по умолчанию – 1.
- **Visibility (Видимость)** – задает видимость параметров модуля в протоколе «Gateway» (в частности, в программе «EasyWorkPLC» разработки ПО «ОВЕН»). Значения выбираются из списка **«yes»** и **«no»,** значение по умолчанию – **«no».**

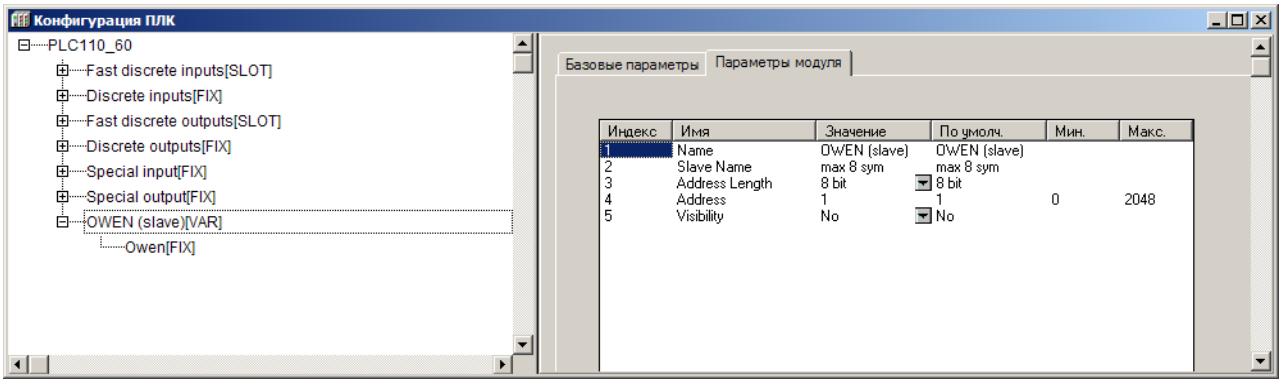


Рисунок 7.35 – Параметры модуля «Owen (Slave)»

7.4.5.1 Подмодуль OWEN (FIX). Настройка коммуникационных интерфейсов

При добавлении модуля «OWEN (Slave)» в конфигурацию ПЛК, в состав модуля уже подключен подмодуль «OWEN (FIX)», к которому, в свою очередь, подключается коммуникационный интерфейс (см. рисунок 7.36).

В ПЛК предусмотрена возможность обмена данными по интерфейсам: **RS-232, RS-485** и **TCP (Ethernet)**.

Для работы с разными коммуникационными интерфейсами в ПЛК предусмотрены соответствующие подмодули (подэлементы). Подключение подэлементов производится выбором требуемой команды (**Добавить подэлемент | <Имя подэлемента>**) контекстного меню (см. рисунок 7.37).

При работе ПЛК в режиме **«Ведомый (slave)»** возможно использование нескольких разных портов, т.е. опрос может вестись по разным интерфейсам. Таким образом, подключая несколько разных портов, можно один модуль соединить с разными Мастерами по разным физическим линиям (и интерфейсам).

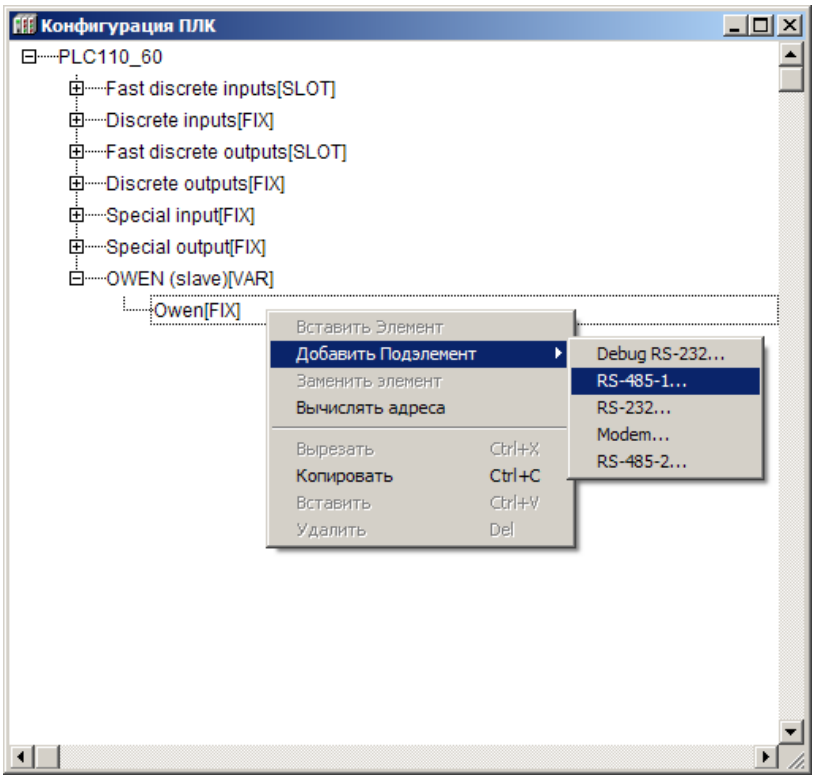


Рисунок 7.36 – Модуль «Owen [FIX]». Выбор интерфейса связи

Количество подключаемых портов ограничено лишь конструкцией ПЛК. Пример подключения нескольких портов представлен на рисунке 7.37.

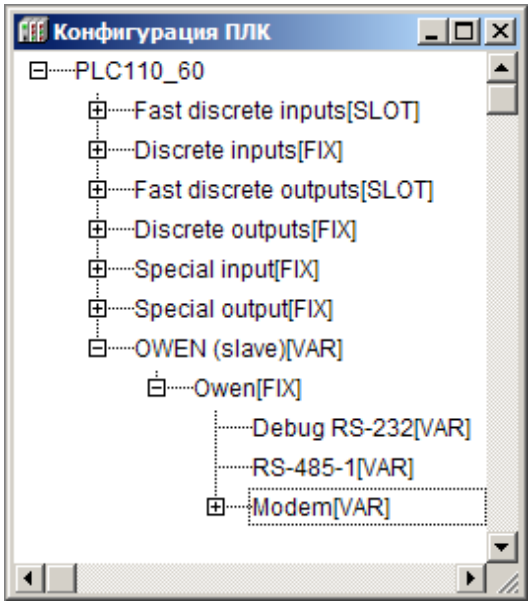


Рисунок 7.37 – Подключение нескольких портов в подмодуле Owen [FIX]

7.4.5.2 Настройка входов / выходов подмодуля

После задания значений параметров подмодуля «OWEN (Slave)» к нему требуется подключить каналы входа / выхода.

Добавление каналов производится выбором команды «Добавить подэлемент | <Наименование подэлемента>» контекстного меню строки «OWEN (Slave)» (см. рисунок 7.44).

В модуле «Owen (Slave)» могут быть использованы только переменные, предназначенные для чтения (обозначены «**Listen»**), использующие различные типы данных (всего – шесть типов):

- **Float variable** – число с плавающей точкой,
- **Float variable + time** – число с плавающей точкой с модификатором времени;
- **Unsigned variable** – целочисленная переменная;
- **Unsigned variable + time** – целочисленная переменная с модификатором времени;
- **String variable** – строковая переменная; максимальная длина – 15 символов, в соответствии со стандартом протокола ОВЕН.
- **Time variable** – позволяет передать время; в протоколе ОВЕН при передаче времени данные имеют следующий формат: «**год:месяц:день:час:минута:секунда:миллисекунда»**.

Переменные, которыми будет обмениваться ПЛК по протоколу ОВЕН, выбираются командой «**Добавить Подэлемент (Append Subelements)»** контекстного меню строки «OWEN(slave)» (см. рисунок 7.38).

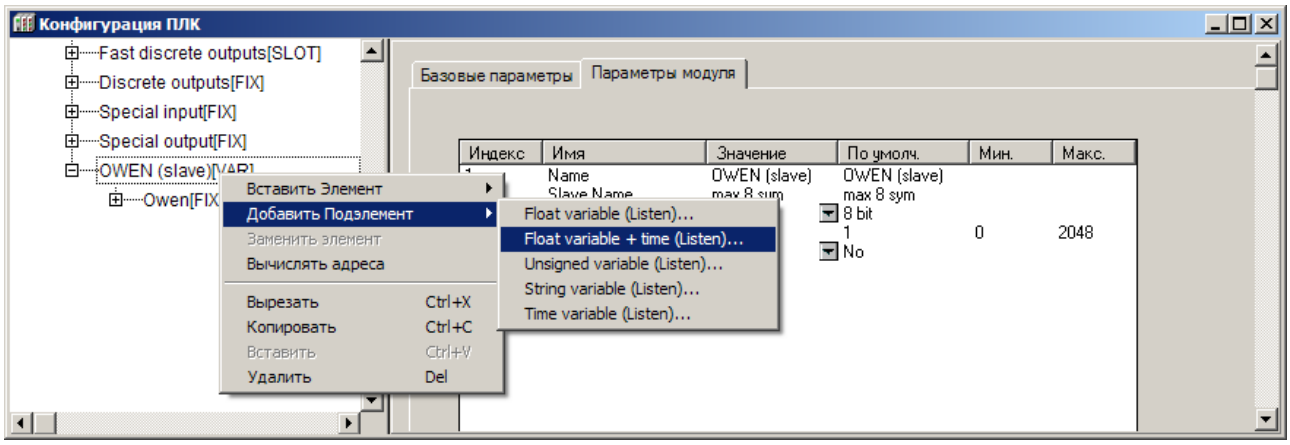


Рисунок 7.38 – Выбор переменных для обмена ПЛК по протоколу ОВЕН

7.4.5.2.1 Параметры переменных протокола ОВЕН

Параметры переменных протокола ОВЕН, общие для всех типов переменных, (см. рисунок 7.39) таковы:

- **Name** – символическое имя элемента, см. п. 7.3.1.
- **Address Length (Длина адреса устройства) –** значения выбираются из списка **«8 bit»** и **«11 bit»**, значение по умолчанию – **«8 bit».**
- **Address (Адрес устройства)** – диапазон значений от 0 до 255 или от 0 до 2048, в зависимости от размера адреса, значение по умолчанию – 0.
- **Hash name (Сетевое имя переменной)** – указывается сетевое имя переменной. Имена переменных ведомых приборов указываются в руководствах по эксплуатации этих приборов. Вводимое имя преобразуется в ПЛК в **Hash-**код, который используется при обмене по сети RS-485.
- **Index (Индекс прибора)** – задает индекс прибора; в параметре **«Use a index? (Использовать индекс?)»** задают использование индекса. В совокупности параметры применяются для управления конфигурационными параметрами ПЛК, определяют наличие линейного индекса у параметра и

задают значение индекса. Диапазон значений от 0 до 65535, значение по умолчанию – 0.

- **Use a index? (Использовать индекс?)** – значения выбираются из списка **«yes»** и **«no»**, значение по умолчанию – **«no».**
- **Polling time, ms (Период опроса устройства, мс)** – диапазон значений – от 20 до 5000, значение по умолчанию – 100.
  Примечание. Следует учитывать физические ограничения, накладываемые характеристиками сети: скорость информационного обмена в сети ограничена, и, если задается большое количество переменных, значения которых часто запрашиваются, то информация будет поступать к прибору с запаздыванием. Поэтому требуется заранее просчитать пропускную способность сети, и, соответственно, либо уменьшить частоту опроса, либо производить опрос по разным линиям и/или др. Поэтому при работе в режимах «Value change (По изменению значения переменных)» и «By Command (По команде)» (режим задается значением параметра «Work mode (Режим работы)», см. ниже), нельзя задавать значение параметра «Polling time» слишком маленьким. По умолчанию его значение 100 мс. Если на реальном проекте будет замечено, что модуль «Owen (Master)» при загрузке программы или при при выборе команды «Login» формирует лишние пакеты и/или запросы, то значение параметра следует увеличить (до 200, 300 и более мс), вплоть до прекращения появления лишних пакетов.
- **Work mode (Режим работы)** – задается режим работы модуля «Owen (Master)» при опросе внешних устройств (для переменных различных типов список допустимых значений различен, см. примечание ниже):

  Polling time (По времени)– контролируемые устройства опрашиваются с периодичностью, заданной в параметре **«Polling time (Период опроса устройства)»**;

  Value change (По изменению значения переменных) – модуль Owen (Master) генерирует запрос устройству при изменении значений выходных переменных модуля;

  Both (Оба варианта» – опрос производится с временным интервалом, заданным в параметре «Polling time» и тогда, когда изменяются значения выходных переменных;

By Command (По команде)– производится однократная посылка запроса, когда в командный канал («Command») переменной, имеющей такой канал, записывается значение **0x00FF**.

Примечание.

1) Для считываемых переменных (тип «Listen») доступен только режим «Polling time (По времени)».

2) Для записываемых переменных (тип «Write») доступны режимы «Polling time (По времени)», «Value change (По изменению значения переменных)» и «Both (По времени и по изменению значения переменных)».

*3) Для переменных с командным каналом (тип «Command») доступен режим «Command (По команде)». В этом режиме управление осуществляется следующим образом: первая посылка значения **0x00FF** в командный канал включает функционирование этой переменной, повторная посылка значения **0x00FF** инициирует проведение опроса. Аналогично опрос инициируется для переменных с командным каналом при работе в других режимах. При посылке в командный канал значения **0x00FE** переменная выключается из цикла опроса мастера.*

- **Repiat Counter** – задает число повторов запроса при ошибке связи.
- **Visibility (Видимость)** – задает видимость параметров модуля в протоколе «Gateway» (в частности, в программе «EasyWorkPLC» разработки ПО «ОВЕН»). Значения выбираются из списка **«yes»** и **«no»,** значение по умолчанию – **«no».**

Для переменных типов «Float variable» (число с плавающей точкой), «Float variable + time» (число с плавающей точкой с модификатором времени) всех модификаций («Listen» – предназначенные для чтения, «Write» – предназначенные для записи, «Command» – имеющие дополнительный командный управляющий канал) – задаются параметры «Float Type (Тип числа с плавающей точкой)», который уточняет вид переменной типа Float и «Precision», который задает точность (определяет положение десятичной точки):

- **Float Type** (тип числа с плавающей точкой) – значение выбирается пользователем из списка (значение по умолчанию – «Float»):

  Float – число с плавающей точкой в формате IEEE, обычно используемое в программировании, в **CODESYS** называется Real, имеет длину 4 байта;

  Float-Pic – переменная размером в 3 байта, и один байт из мантиссы удаляется, т.е. число с меньшей точностью, но и размер переменной (количество байт) меньше;

  Fix point binary – число с фиксированной точкой в двоичном виде; например, число 3 будет записано в виде «0011». Положение десятичной точки для параметров с фиксированной точкой задается параметром «**Precision (Точность)».**

  Fix point BCD –число с фиксированной точкой в двоично-десятичном виде (двоично-десятичный код – форма записи целых чисел, когда каждый десятичный разряд числа записывается в виде его четырёхбитного двоичного кода. Например, число 311 будет записано в виде «0011 0001 0001». Положение десятичной точки для параметров с фиксированной точкой задается параметром «**Precision (Точность)».**

- **Precision (Точность)** – задает точность (определяет положение десятичной точки) для параметров с фиксированной точкой: если выбирается зна-

чение точности «2», то число 10,12 так и будет передано. При значении «1» – 10,1, при значении «3» – 10,120. Диапазон значений – от 0 до 7, значение по умолчанию – 2.

Отличие переменной **Float** с модификатором времени состоит в том, что, при тех же параметрах, в поле данных, кроме собственно значения, присутствует еще и время (в сотых долях секунды).

Примеры задания параметров для различных случаев применения ПЛК представлены в Приложении Б.

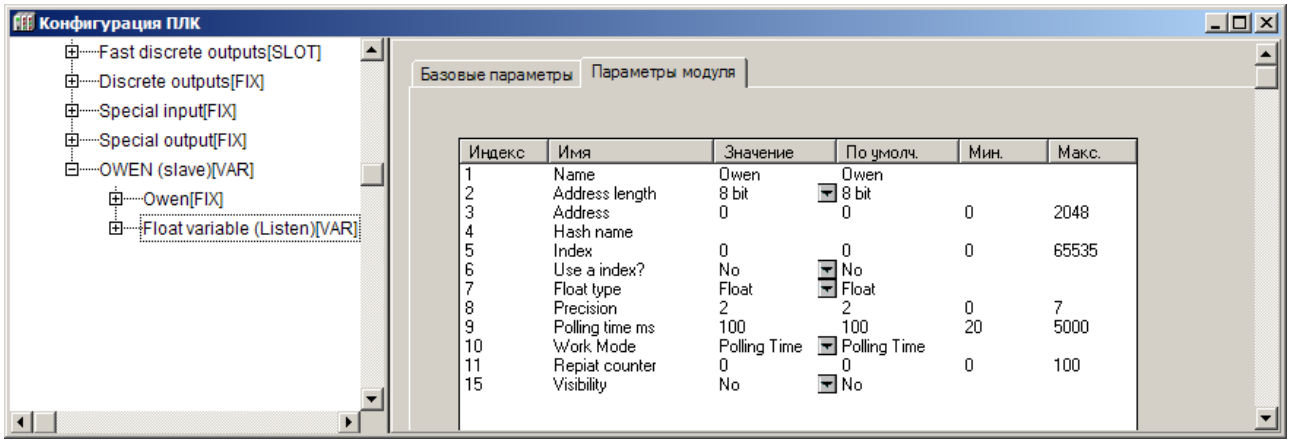


Рисунок 7.39 – Параметры переменных протокола ОВЕН

7.4.6 Модуль «Owen (Spy)»

Модуль «OWEN (Spy)» используется для проведения мониторинга информационных обменов в сети, в которую включен ПЛК. Модуль не отвечает на запросы Мастера сети, а только прослушивает обмен данными в сети RS-485 и может быть настроен таким образом, что в ходе опроса Мастером сети какого-либо устройства, ответ этого устройства прослушивается модулем «OWEN (Spy)» и записывается им во встроенную переменную. Этот механизм может быть использован, если ПЛК требуется интегрировать в существующую сеть: получать из нее данные для последующей обработки и выполнения заданных действий. Использование модуля «OWEN (Spy)» позволяет не останавливать работу системы в ходе интеграции: ПЛК прослушивает требуемые данные и выполняет запрограммированные действия.

При установке модуля «OWEN (Spy)» выбирается коммуникационный интерфейс для обмена данными с другими устройствами, добавляются и настраиваются требуемые переменные. Модуль «OWEN (Spy)» обеспечивает обмен информацией по протоколу ОВЕН (протокол предназначен для описания процесса обмена информацией между приборами фирмы «Овен» и между приборами и ПЭВМ на базе сети RS-485).

Примечание. *При случайном отключении питания в процессе работы ПЛК последние (текущие) значения переменных сохраняются в энергонезависимой памяти и восстанавливаются при возобновлении работы прибора.*

Модуль «OWEN (Spy)» – составной и имеет в своем составе подмодуль «**OWEN [FIX]»** – см. ниже.

Модуль имеет единственный параметр: «Visibility (Видимость)», который задает видимость параметров модуля в протоколе «Gateway» (в частности, в программе «EasyWorkPLC» разработки ПО «ОВЕН»). Значения выбираются из списка **«yes»** и **«no»,** значение по умолчанию – **«no».**

7.4.6.1 Подмодуль OWEN (FIX). Настройка коммуникационных интерфейсов

При добавлении модуля «OWEN (Spy)» в конфигурацию ПЛК, в состав модуля уже подключен подмодуль «OWEN (FIX)», к которому, в свою очередь, подключается коммуникационный интерфейс.

В ПЛК предусмотрена возможность обмена данными по интерфейсам: **RS-232** и **RS-485**.

Для работы с разными коммуникационными интерфейсами в ПЛК предусмотрены соответствующие подмодули (подэлементы). Подключение подэлементов производится выбором требуемой команды (**Добавить подэлемент | <Имя подэлемента>**) контекстного меню (см. рисунок 7.40). Допустимо использование нескольких разных портов, т.е. опрос может вестись по разным интерфейсам.

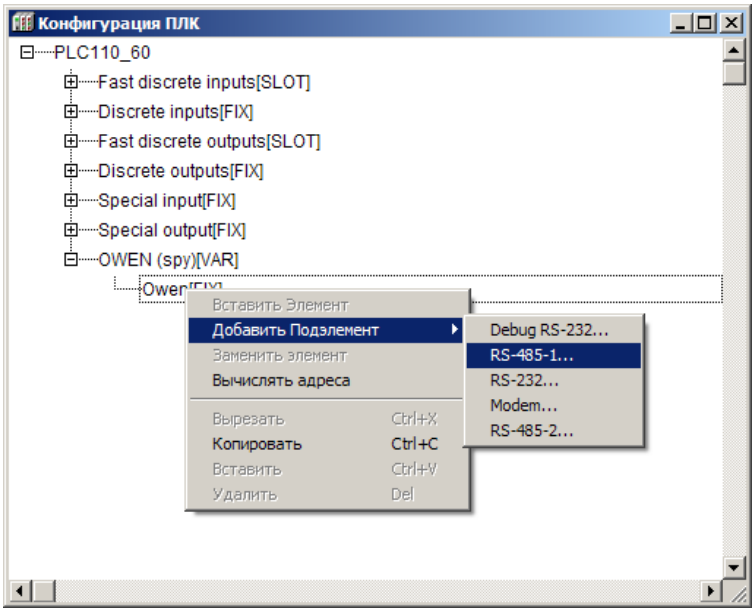


Рисунок 7.40 – Модуль «Owen [FIX]». Выбор интерфейса связи

Количество подключаемых портов ограничено лишь конструкцией ПЛК. Пример подключения нескольких портов представлен на рисунке 7.41.

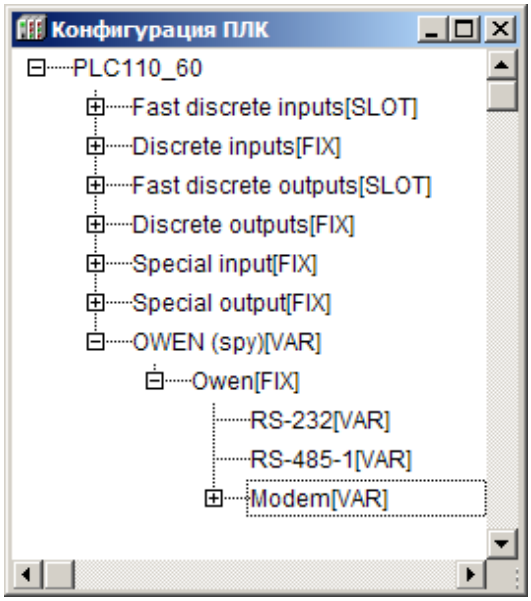


Рисунок 7.41 – Подключение нескольких портов в подмодуле Owen [FIX]

7.4.6.2 Настройка входов и выходов модуля

После задания значений параметров подмодуля «OWEN (Spy)» к нему требуется подключить каналы, задающие входные параметры (параметры, значение которых модуль запрашивает у сети) модуля.

В модуле «Owen (Spy)» могут быть использованы только переменные, предназначенные для чтения (обозначены «**Listen»**), использующие различные типы данных (всего – шесть типов):

- **Float variable** – число с плавающей точкой,
- **Float variable + time** – число с плавающей точкой с модификатором времени;
- **Unsigned variable** – целочисленная переменная;
- **Unsigned variable + time** – целочисленная переменная с модификатором времени;
- **String variable** – строковая переменная; максимальная длина – 15 символов, в соответствии со стандартом протокола ОВЕН.
- **Time variable** – позволяет передать время; в протоколе ОВЕН при передаче времени данные имеют следующий формат: «**год:месяц:день:час:минута:секунда:миллисекунда»**.

Переменные, которыми будет обмениваться ПЛК по протоколу ОВЕН, выбираются командой «**Добавить Подэлемент (Append Subelements)»** контекстного меню строки «OWEN (Spy)» (см. рисунок 7.42).

При добавлении канала в «OWEN (Spy)» в дерево конфигурации добавляется подкаталог, содержащий внутри себя канал, в котором и отображаются полученное / передаваемое по сети значение (см. рисунок 7.42).

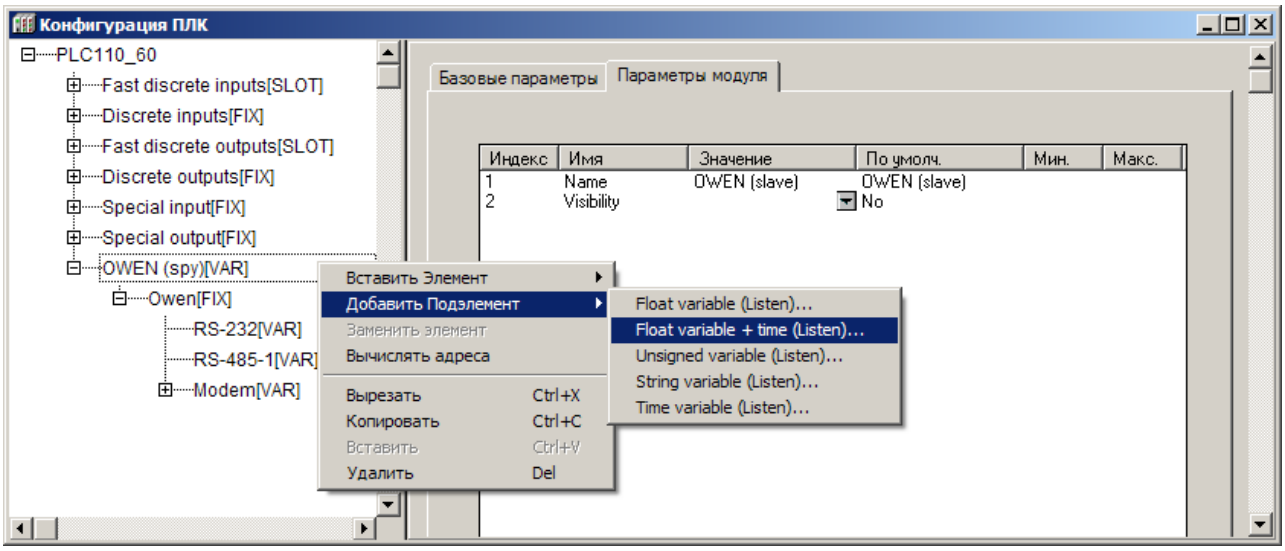


Рисунок 7.42 – Добавление каналов ввода/вывода модуля «OWEN (Spy)»

7.4.6.3 Параметры переменных протокола ОВЕН

Параметры переменных протокола ОВЕН, общие для всех типов переменных, (см. рисунок 7.43) таковы:

- **Name** – символическое имя элемента, см. п. 7.3.1.
- **Address Length (Длина адреса устройства) –** задает в битах размер адреса ведомого устройства, ответ которого необходимо прослушать; значения выбираются из списка **«8 bit»** и **«11 bit»**, значение по умолчанию – **«8 bit».**
- **Address (Адрес устройства)** – задает адрес ведомого устройства, ответ которого необходимо прослушать; диапазон значений от 0 до 255 или от 0 до 2048, в зависимости от размера адреса, значение по умолчанию – 0.
- **Hash name (Сетевое имя переменной)** – указывается сетевое имя переменной ведомого устройства, опрашиваемого Мастером; имена переменных ведомых приборов указываются в руководствах по эксплуатации этих приборов. Вводимое имя преобразуется в ПЛК в **Hash**-код, который используется при обмене по сети RS-485.
- **Index (Индекс прибора)** – задает индекс прибора; в параметре **«Use a index? (Использовать индекс?)»** задают использование индекса. В совокупности параметры применяются для управления конфигурационными параметрами ПЛК, определяют наличие линейного индекса у параметра и задают значение индекса. Диапазон значений от 0 до 65535, значение по умолчанию – 0.
- **Use a index? (Использовать индекс?)** – значения выбираются из списка **«yes»** и **«no»**, значение по умолчанию – **«no».**
- **Polling time, ms (Период опроса устройства, мс)** – модулем «OWEN (Spy)» не используется.
- **Work mode (Режим работы)** – задается режим работы модуля при опросе внешних устройств: «**Polling time (По времени)**».
- **Repiat Counter** – задает число повторов запроса при ошибке связи.
- **Visibility (Видимость)** – задает видимость параметров модуля в протоколе «Gateway» (в частности, в программе «EasyWorkPLC» разработки ПО «ОВЕН»). Значения выбираются из списка **«yes»** и **«no»,** значение по умолчанию – **«no».**

Для переменных типов «Float variable» (число с плавающей точкой), «Float variable + time» (число с плавающей точкой с модификатором времени) всех модификаций («Listen» – предназначенные для чтения, «Write» – предназначенные для записи, «Command» – имеющие дополнительный командный управляющий канал) – задаются параметры «Float Type (Тип числа с плавающей точкой)», который уточняет вид переменной типа Float и «Precision», который задает точность (определяет положение десятичной точки):

– **Float Type** (тип числа с плавающей точкой) – значение выбирается пользователем из списка (значение по умолчанию – «Float»):

  Float – число с плавающей точкой в формате IEEE, обычно используемое в программировании, в **CODESYS** называется Real, имеет длину 4 байта;

  Float-Pic – переменная размером в 3 байта, и один байт из мантиссы удаляется, т.е. число с меньшей точностью, но и размер переменной (количество байт) меньше;

  Fix point binary – число с фиксированной точкой в двоичном виде; например, число 3 будет записано в виде «0011». Положение десятичной точки для параметров с фиксированной точкой задается параметром «**Precision (Точность)».**

  Fix point BCD – число с фиксированной точкой в двоично-десятичном виде (двоично-десятичный код – форма записи целых чисел, когда каждый десятичный разряд числа записывается в виде его четырёхбитного двоичного кода. Например, число 311 будет записано в виде «0011 0001 0001». Положение десятичной точки для параметров с фиксированной точкой задается параметром «**Precision (Точность)».**

– **Precision (Точность)** – задает точность (определяет положение десятичной точки) для параметров с фиксированной точкой: если выбирается значение точности «2», то число 10,12 так и будет передано. При значении «1» – 10,1, при значении «3» – 10,120. Диапазон значений – от 0 до 7, значение по умолчанию – 2.

Отличие переменной **Float** с модификатором времени состоит в том, что, при тех же параметрах, в поле данных, кроме собственно значения, присутствует еще и время (в сотых долях секунды).

Примеры задания параметров для различных случаев применения ПЛК представлены в Приложении Б.

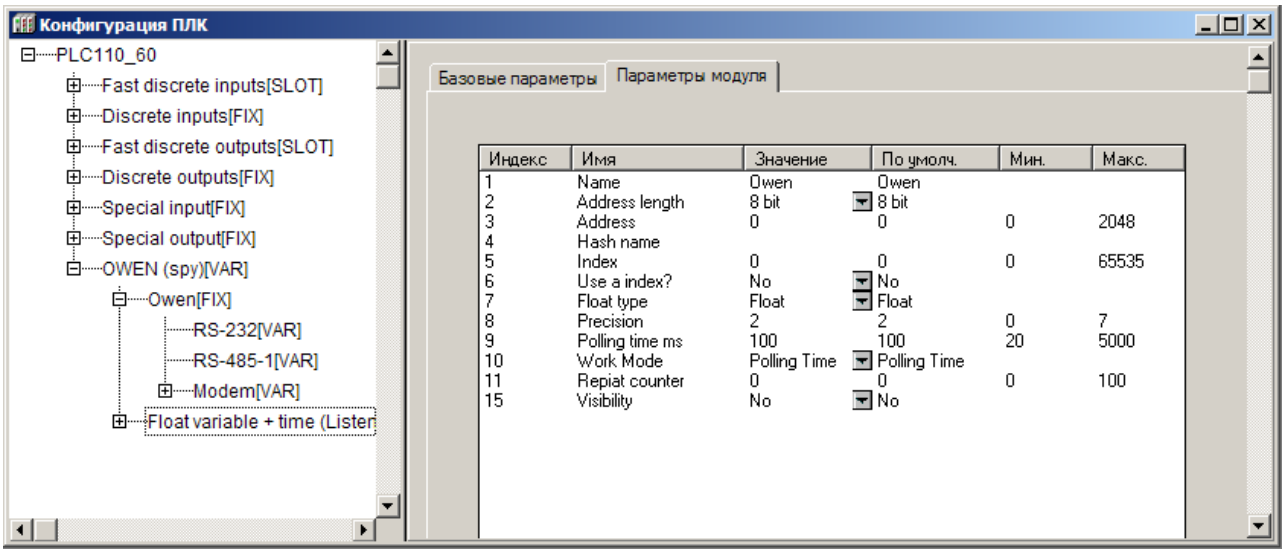


Рисунок 7.43 – Параметры переменных протокола ОВЕН

7.4.7 Statistic (Модуль статистики)

Модуль статистики (Statistic) предназначен для выдачи в программу пользователя информационных данных о функционировании ПЛК.

Модуль не имеет параметров и содержит следующие каналы (см. рисунок 7.44):

- **Last cycle time in 100 mks (Значение последнего цикла работы ПЛК в мкс)**, позволяет пользователю оценить объем вычислительных ресурсов, который требуется для работы написанной им программы. Если цикл оказывается больше, заданного в параметрах работы ПЛК параметра **MinCycleLength**, то это означает, что программа пользователя слишком требовательна к ресурсам, и параметр **MinCycleLength** желательно увеличить, чтобы циклы не перекрывались;
- **Time to backup power down, s (Время, оставшееся до выключения ПЛК, сек)**, отражает ресурс времени работы ПЛК от аккумуляторных батарей (с отключенным внешним питанием). Характеристика оценочная, точный учет влияния всех факторов (температура ПЛК и внешней среды, точность измерения и пр.) затруднителен. В случае работы ПЛК от сети аккумуляторная батарея заряжается, и этот параметр косвенно указывает на процесс зарядки (600 с соответствует полностью заряженной батарее);
- **Temp inside PLC (Температура внутри ПЛК),** отражает температуру, замеренную датчиком внутри корпуса ПЛК (у разных моделей ПЛК температура может измеряться на разных платах, определяется интенсивностью нагрева конкретных плат). Характеристика косвенно свидетельствует о рабочем состоянии ПЛК.
- **Наличие/отсутствие питания** – отображается в битовом канале **Power status** (питание есть – «1»);
- **Error (Ошибка)** – описание ошибок ПЛК см. в приложении В.
- **Power status (Состояние питания)**– логическая переменная, устанавливающаяся в TRUE при наличии питания от сети и имеющая значение FALSE в случае отсутствия питания или аварии п по питанию;
- **CPU is overloaded, optimize your program or increase PLC cycle (центральный процессор перегружен, оптимизируйте вашу программу или увеличьте время цикла ПЛК)** – логическая переменная, устанавли-

вающаяся в TRUE, если цикл оказывается больше, заданного в параметрах работы ПЛК параметра **MinCycleLength,** в случае выполнения программы в рамках заданного времени цикла переменная имеет значение FALSE;

– **Free processor resource mks in 1 cycle (свободное процессорное время за один цикл в мкс)**– значение этой переменной соответствует времени в цикле исполнения программы, не занятому исполнением программы.

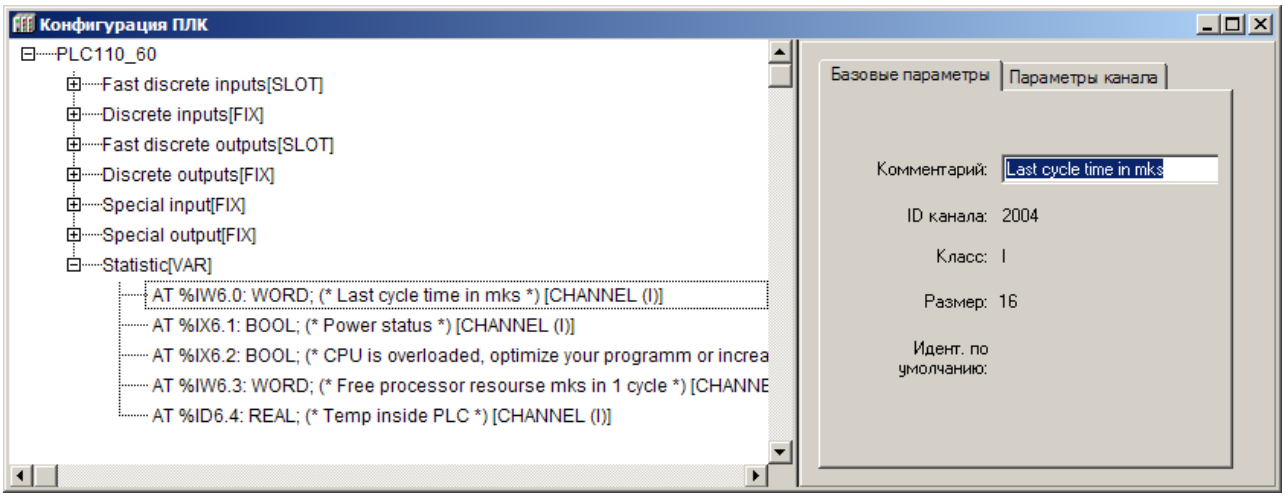


Рисунок 7.44 – Каналы модуля статистики (Statistic)

7.4.8 Universal network module (Универсальный сетевой модуль)

Модуль «Universal network module (Универсальный сетевой модуль)» предназначен для организации универсального коммуникационного интерфейса ПЛК, выполняющего прием / передачу последовательности байт через встроенные порты контроллера (RS-232/RS-485/Ethernet). Подключение модуля резервирует область памяти ввода / вывода ПЛК, с которой будут выполняться функции, включенные в специализированную библиотеку дополнительных программных модулей UNM.lib.

Особенностью данной библиотеки (и, соответственно, модуля конфигурации ПЛК) является возможность работать одновременно с протоколами ModBus, DCON и ОВЕН на одном физическом интерфейсе.

Это позволяет создать модуль опроса устройства стандартными командами и в то же время выдавать в интерфейс и получать из интерфейса в нужное время произвольную последовательность байт. Например, если к порту RS-232 подключен модем, то до начала работы стандартного модуля опроса программа пользователя позволит установить связь с удаленным устройством используя «AT» последовательности. После этого начнет работать модуль опроса устройства через стандартный протокол.

Описание функций библиотеки UNM.lib (Universal Network Module) см. в документе «Библиотека UNM_01».

Модуль имеет один канал (DebugRS-232 [SLOT]) и один параметр: **«Видимость» (Visibility)**, задающий видимость параметров модуля в протоколе «Gateway» (в частности, в программе «EasyWorkPLC» разработки ПО «ОВЕН»). Значения выбираются из списка **«yes»** и **«no»,** значение по умолчанию – **«no».**

При добавлении модуля «Universal network module (Универсальный сетевой модуль)» в конфигурацию ПЛК, в состав модуля уже подключен подмодуль «DebugRS-232[SLOT]», к которому, в свою очередь, подключается коммуникационный интерфейс (см. рисунок 7.45).

В ПЛК предусмотрена возможность обмена данными по интерфейсам: **RS-232, RS-485** и **TCP (Ethernet)**. Для работы с разными коммуникационными интерфейсами в ПЛК предусмотрены соответствующие подмодули (подэлементы). Подключение подэлементов производится выбором требуемой команды (**Заменить подэлемент | <Имя подэлемента>**) контекстного меню (см. рисунок 7.45).

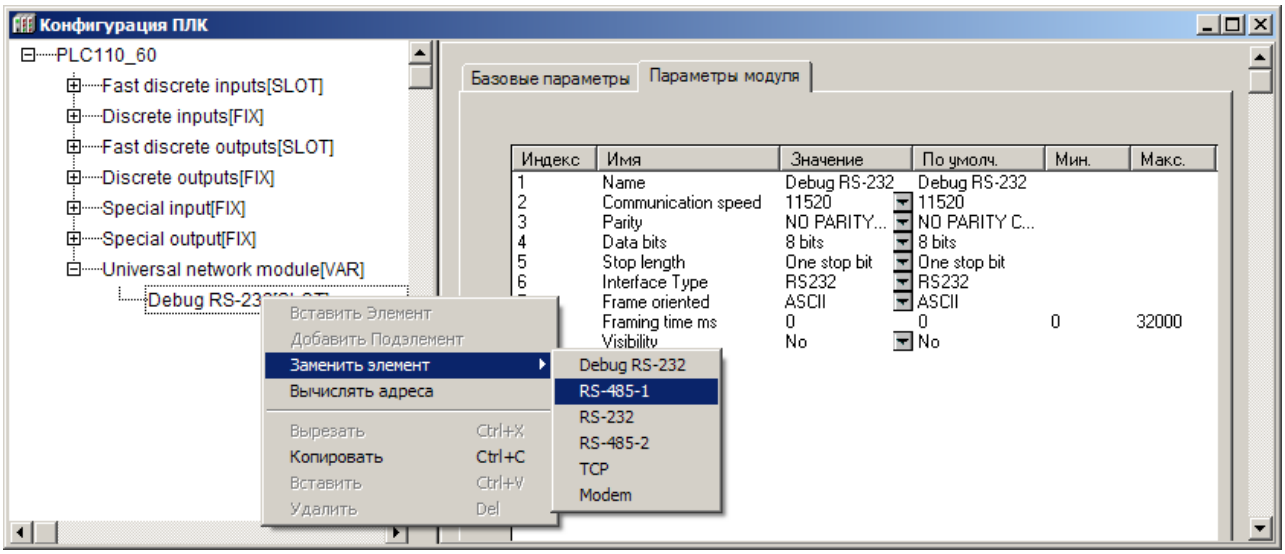


Рисунок 7.45 – Параметры и контекстное меню замены подэлементов Универсального сетевого модуля (Universal network module)

7.4.9 Модуль «Архиватор» (Archiver)

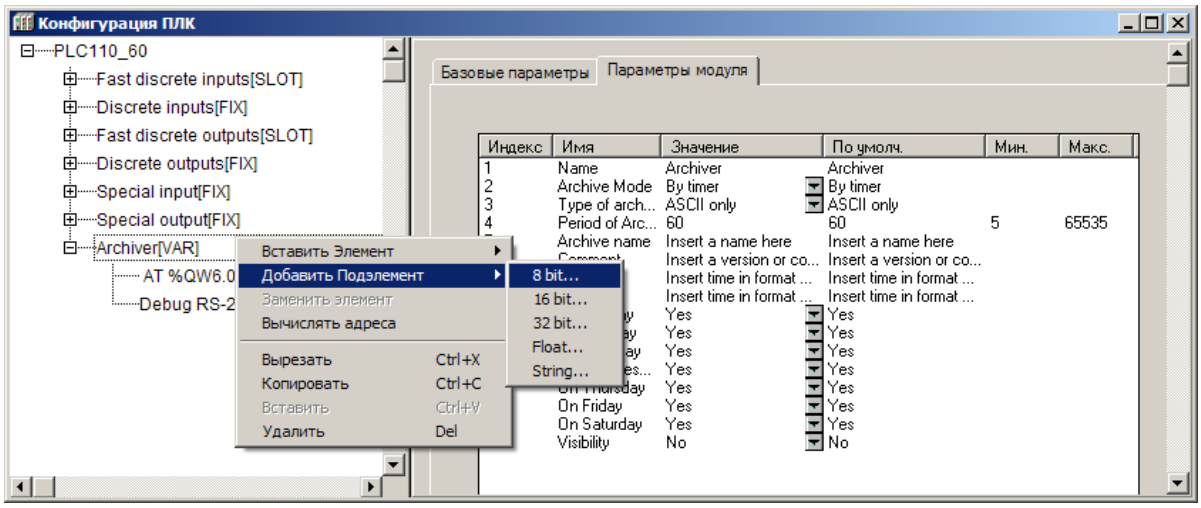
Модуль «Archiver (Архиватор)» используется для архивирования требуемых данных. Архивируемые данные могут храниться на **Flash** диске ПЛК и извлекаться оттуда при необходимости, или выводиться через коммуникационный интерфейс (например, они могут быть распечатаны на принтере, подключенном к ПЛК через последовательный интерфейс). Кроме того, к модулю «Archiver (Архиватор)» может быть добавлен подмодуль архивации информации в файл («File Output», см. ниже в пункте 7.4.9.1.3).

Создание перечня архивируемых переменных (внесение переменных в список для последующего архивирования) производится вызовом команды «Добавить подэлемент (Append Subelement)» контекстного меню строки «Archiver».

В список могут быть добавлены переменные следующих типов: 8-, 16- и 32-битная, число с плавающей точкой («Float») и текстовая строка («String») (максимум 15 символов + завершающий нуль).

После добавления переменной любого типа в ее параметрах необходимо задать имя архивируемой переменной – **Variable Name** – этим же именем переменная будет именоваться в архивном файле (см. рисунок 7.46, б).

а)



б)

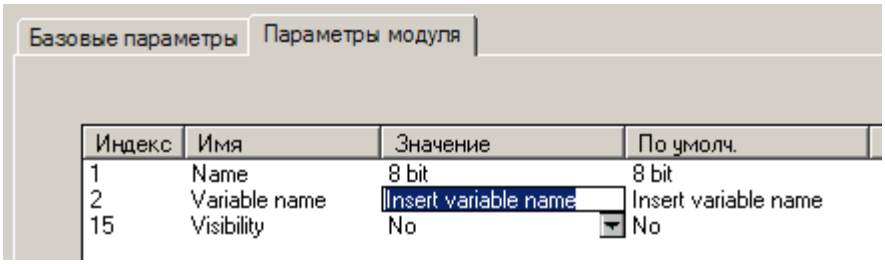


Рисунок 7.46 – Внесение (а) и именование (б) переменных в список архивируемых

Параметры передачи архивных данных (режим передачи, формат данных) задаются в параметрах модуля (см. рисунок 7.47):

- **Name** – символическое имя элемента, см. п. 7.3.1.
- **Archive Mode (Режим проведения архивации)** – задает режим архивации. Значения выбираются из списка («By timer», «By change value» и «By command»), значение по умолчанию – «By timer».

  By timer (по таймеру) – данные записываются в архив с заданным периодом архивации;

  By change value (по изменению значений): если какая-то переменная, включенная в список архивации, меняет свое значение, то происходит ее архивация, причем только этой переменной. Изменения могут записываться не чаще, чем раз в секунду;

  By command (по команде): если в переменной **Status** модуля архивации записана специальная команда, то происходит либо старт архивации, либо ее остановка (**0x00FE – «стоп»**, а **0x00FF – «старт»**); [7]

- **Type of archive (Тип данных архивации)** – задает тип записи архивируемых данных. Значения выбираются из списка («ASCII only» и «Mixed»), значение по умолчанию – «ASCII only»:

  ASCII only – данные выдаются в текстовом виде, удобном для чтения пользователя, для печати и т.п.;

[7] Действия архиватора в ответ на команды сводятся к следующему:
- при загрузке проекта модуль архивации находится в режиме "Работает".
- повторная команда “старт” приводит к немедленному акту архивации.
- если архиватор остановлен, то команда “старт”запустит его.

Mixed – данные выдаются в смешанном виде: запись имеет заголовок архива с именами переменных, архива, временные данные в удобном для чтения виде, а все архивируемые переменные записываются в бинарном виде.

- **Period of Archiving (Период архивации, сек)** – задает периодичность обновления данных архива при работе модуля в режиме **«по таймеру»**. Диапазон значений от 5 до 65535 сек, значение по умолчанию – 60.
- **Archive Name (Имя архива)** – задает имя архива, которое записывается в начале файла.
- **Comment (Комментарий архива)** – задает текст комментария к архиву. Здесь может быть введена информация, позволяющая в будущем идентифицировать конкретный архив по дополнительным признакам.
- **Start time(Время начала архивации)** – устанавливается время старта архивации.
- **Stop time (Время остановки архивации)** – устанавливается время остановки архивации.

  Внимание! *Параметры «Start time (Время начала архивации)» и «Stop time (Время остановки архивации)», задающие временные рамки процесса архивирования, независимы друг от друга, т.е. один или оба параметра могут быть не заданы. Для параметров определен формат, в котором они должны задаваться –* ***чч:мм:сс****, – с обязательным использованием полноформатного задания величин и разделителя «двоеточие». При неполном формате и/или использовании иного разделителя программа проигнорирует информацию, как ошибочную.*

- **On Sunday (Воскресенье)… On Saturday (Суббота) –** всего семь параметров – назначается день (дни) недели, когда будет производиться архивация. Значения выбираются из списка **«yes»** и **«no»**, значение по умолчанию – **«yes».**
- **Visibility (Видимость)** – задает видимость параметров модуля в протоколе «Gateway» (в частности, в программе «EasyWorkPLC» разработки ПО «ОВЕН»). Значения выбираются из списка **«yes»** и **«no»,** значение по умолчанию – **«no».**

  Внимание! *Установлен приоритет между всеми условиями старта и остановки архивирования. Главный приоритет имеет переменная* ***File Status****: если в ней записана команда* ***«стоп»****, то операция в любом случае прекратится, если* ***«старт»*** *– она будет выполнена минимум один раз. Следующим по приоритету идет день недели и, далее, время старта и время останова процедуры архивации.*

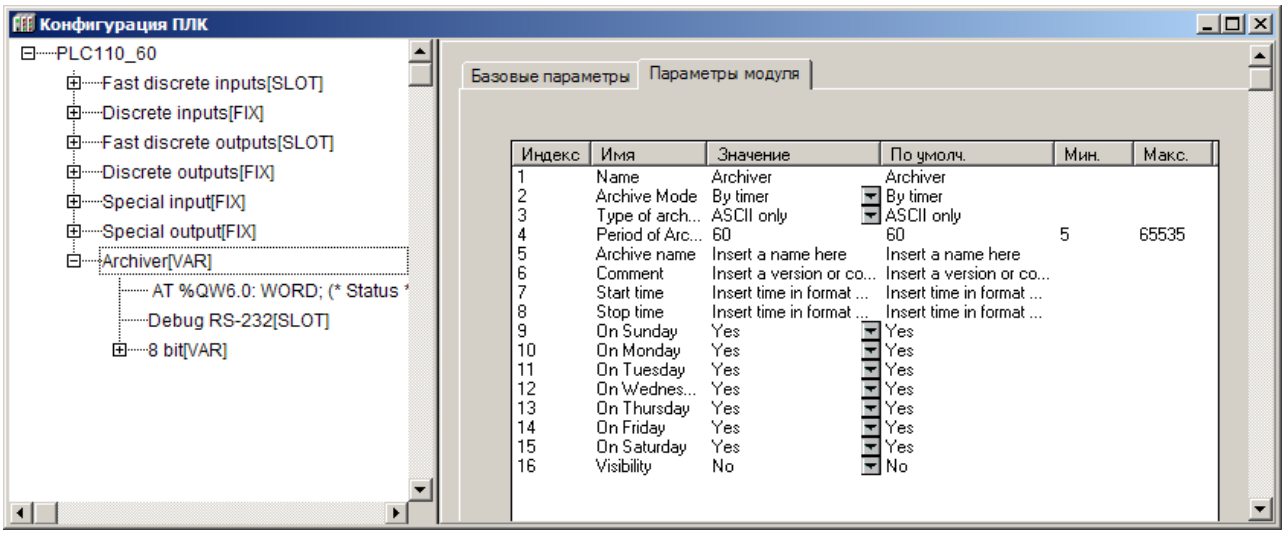


Рисунок 7.47 – Параметры модуля «Archiver (Архиватор)»

7.4.9.1 Настройка коммуникационных интерфейсов модуля

По умолчанию к модулю «Archiver (Архиватор)» подключен подмодуль интерфейсного порта, через который будут передаваться архивные данные. Интерфейсный порт может быть заменен на требуемый выбором команды «Replace element (Заменить элемент) контекстного меню (см. рисунок 7.48).

Настройка последовательных интерфейсов, интерфейса TCP, модема, и выгрузка архива в файл (порт «File Output») описана ниже.

В модуле «Archiver (Архиватор)» имеется канал «Status», отображающий статус архива и коды возникающих ошибок; коды ошибок ПЛК представлены в Приложении В.

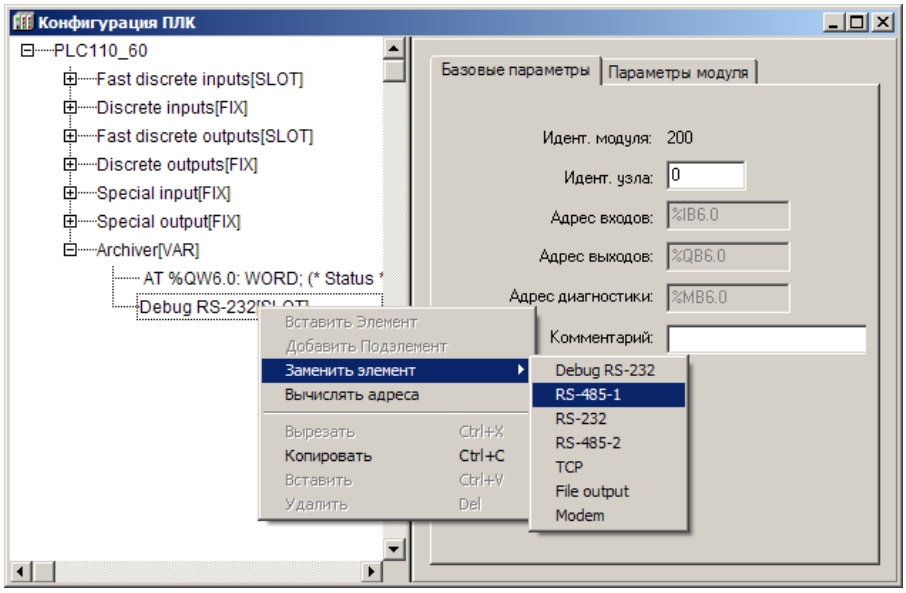


Рисунок 7.48 – Смена интерфейсного порта модуля «Архиватор»

7.4.9.1.1 Подмодуль архивации информации в файл (File Output)

Подмодуль интерфейсного порта «File Output» – программный модуль, задающий параметры архивации информации в файл.

Модуль «File Output» является подчиненным подмодулем модуля «Архиватор».

Подмодуль «File Output» имеет собственную переменную «File Status», в которой сохраняется информация о работе подмодуля. Коды ошибок ПЛК представлены в Приложении В.

Параметры подмодуля (см. рисунок 7.49):

- **Name** – символическое имя элемента, см. п. 7.3.1.
- **File name (Имя файла)** – задает имя файла, в который будет записываться архивная информация. Значение по умолчанию – **«File_name.log»**.
- **Mode (Режим работы модуля)** – задает вариант архивации информации из списка «Append to end», «Rewrite on start», «Rewrite on oversize» и «Shift Mode» (значение по умолчанию – «Appendtoend»):

  Append to end – «добавить в конец» – информация добавляется в конец файла, и, как только файл переполняется, запись прекращается. Файл имеет ограничение либо по размеру (в байтах), либо по количеству записей (задаваемому в параметре **Max file size**);

  Rewrite on start – «перезапись при старте»– старый файл стирается при старте ПЛК или загрузке новой конфигурации и начинается запись файла с самого начала;

  Rewrite on oversize – «перезапись старого файла при превышении заданного размера» – файл стирается при достижением им заданного размера, и запись начинается сначала;

  Shift Mode– «режим сдвига» – вариант работы, при котором, при достижении файлом заданного размера, вторая (более поздняя по времени записи) половина файла переносится в начало, запись продолжается, дописывается, т.е. остаются самые последние записи

- **Type (Тип)** – определяет, каким образом подсчитывается размер файла. Значение выбирается из списка **«Text»** и **«Binary»** (значение по умолчанию – **«Text»)**:

  при текстовом режиме **(Text)** подсчет осуществляется по количеству записей, при этом каждая запись заканчивается символом перевода каретки;

  при цифровом (двоичном) режиме **(Binary)** подсчет осуществляется по размеру файла в байтах.

- **Max file size (Размер записи)** – задает ограничение размера записываемого файла, при этом размер определяется в зависимости от типа: при текстовом – количество записей, при бинарном – количество байт. Диапазон значений от 100 до 320000, значение по умолчанию – 500.
- **Visibility (Видимость)** – задает видимость параметров модуля в протоколе «Gateway» (в частности, в программе «EasyWorkPLC» разработки ПО «ОВЕН»). Значения выбираются из списка **«yes»** и **«no»,** значение по умолчанию – **«no».**

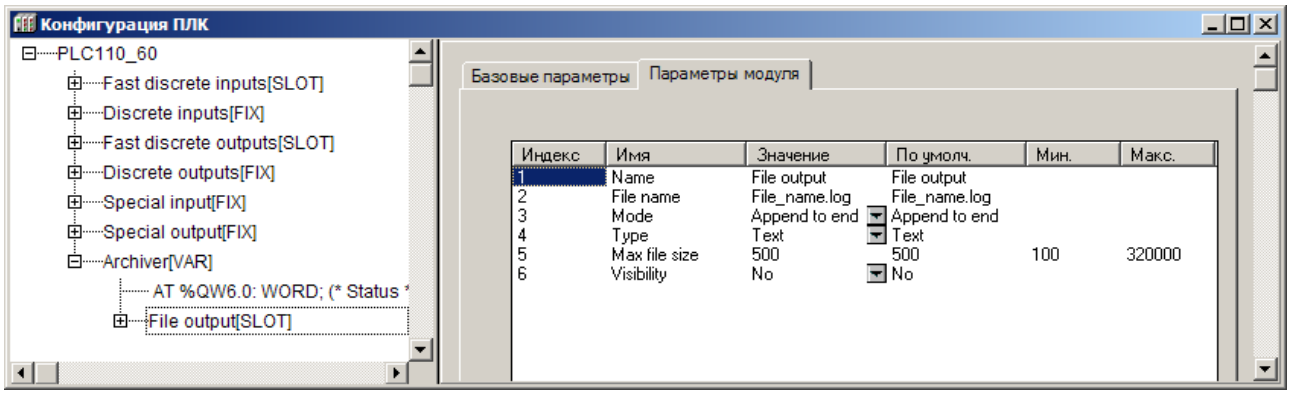


Рисунок 7.49 – Параметры подмодуля FileOutput

7.4.10 Модуль Constant value (Константа)

Модуль «Константа» (Constant value) – содержит значение константы (задаваемой в параметре модуля). При загрузке конфигурации данное значение записывается в канал модуля и может быть считано в программу ПЛК.

Основным назначением модулей **Константа (Constant value)** является их использование при работе программы EasyWorkPLC: в программе EasyWorkPLC комментарий высвечивается как имя константы, при этом ее значение может редактироваться пользователем. Таким образом, значения переменных, объявленных как константы, можно изменять, используя программу EasyWorkPLC, не подключая ПО CODESYS. Эта возможность позволяет обеспечить изменения значений переменных в программе ПЛК, не меняя при этом самой программы, обслуживающим персоналом, не имеющим доступа к ПО CODESYS.

В текущую конфигурацию могут добавляться константы трех видов: 32-битные константы, константы формата FLOAT (Real) и 16-битные константы.

Параметры модуля (см. рисунок 7.50):

– **«Значение константы» (Constant Value)** – задает значение вводимой константы. В зависимости от вида константы пользователь вводит значения, учитывая возможные диапазоны вводимых констант (значение по умолчанию – 0):

  для 32-битной константы диапазон составляет [-2*109…2*109];
  для константы формата FLOAT (Real) – числа с плавающей точкой;
  для 16-битной константы – [-32768…32768].

– **«Комментарий для программы EasyWorkPLC»** (CommentforEasyWorkPLC) – вводится строка комментария для программы EasyWorkPLC. Комментарий может быть написан на русском языке.
– **«Видимость» (Visibility)** – видимость параметров модуля в протоколе «Gateway» (в частности, в программе «EasyWorkPLC» разработки ПО «ОВЕН»). Значения выбираются из списка **«yes»** и **«no»,** значение по умолчанию – **«no»**.

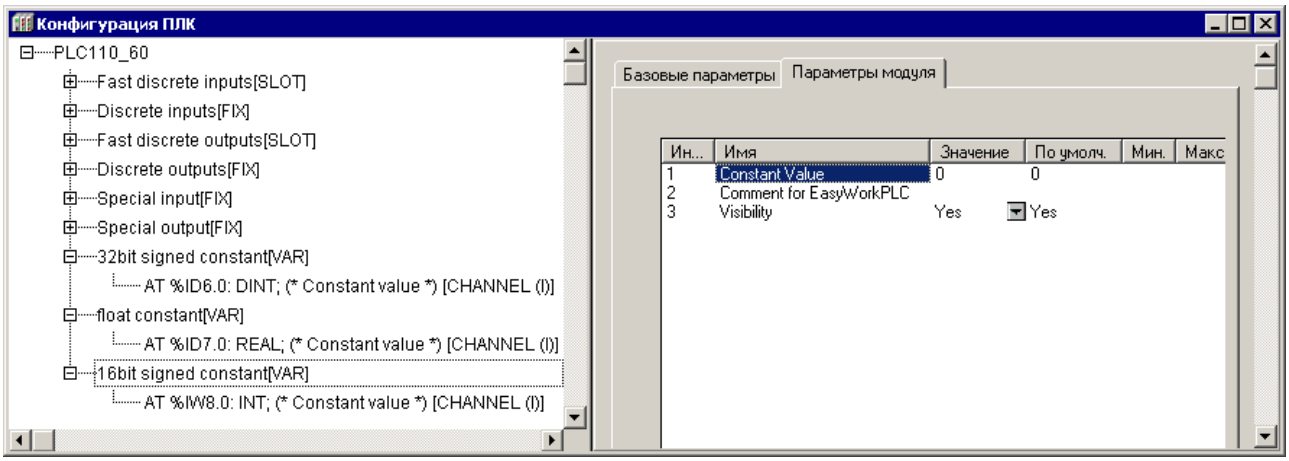


Рисунок 7.50 – Параметры модуля «Константа» (Constantvalue)

7.4.11 Модуль Extended settings (Расширенные настройки)

Модуль «Расширенные настройки» (Extended settings) – служит для управления режимом работы подтяжки линий RS-485 интерфейсов (Физический мастер — физическое устройство расширения). При загрузке конфигурации значение записывается в канал модуля и может быть считано в программу ПЛК. Управление возможно как при старте, так и в ходе работы программа ПЛК По умолчанию оба интерфейса 485 работают в режиме «Физический мастер».

Также в модуле «Extended settings» индицируется состояние батареи питания часов ПЛК.

Текущую конфигурация содержит следующие параметры:

– **«Состояние батареи» (Batery discharged)** – показывает состояние заряда батареи. В зависимости от состояния батареи данный параметр будет принимать одно из двух состояний: False (0)- батарея заряжена,True (1) – указывает, что до окончания срока работы батарее не более 5 месяцев

Поле в модуле дублирует индикатор на лицевой панели и включается при снижении напряжения на батарее до 2,7 В. При этом часы ПЛК сохраняют работоспособность при снижении напряжения питания на батарее до 1,4В. После включения индикатора «батарея разряжена» работа часов при нормальных условиях обеспечивается ещё на протяжении 6 месяцев..

– **«Подтягивающий резистор интерфейса RS 485-1» (RS 485-1 master mode)** –. Значения выбираются из списка **«master device»** и **«terminal device»,** значение по умолчанию – **«master device»**. Значение выбирается в зависимости от режима работы по данному порту: **«master device» -** режим-master и **«terminal device» -** режим slave на данном порте.

– **«Подтягивающий резистор интерфейса RS 485-2» (RS 485-2 master mode) -** Значения выбираются из списка **«master device»** и **«terminal device»,** значение по умолчанию – **«master device»**. Значение выбирается в зависимости от режима работы по данному порту: **«master device» -** режим-master и **«terminal device» -** режим slave на данном порте.

Параметры модуля (см. рисунок 7.51).

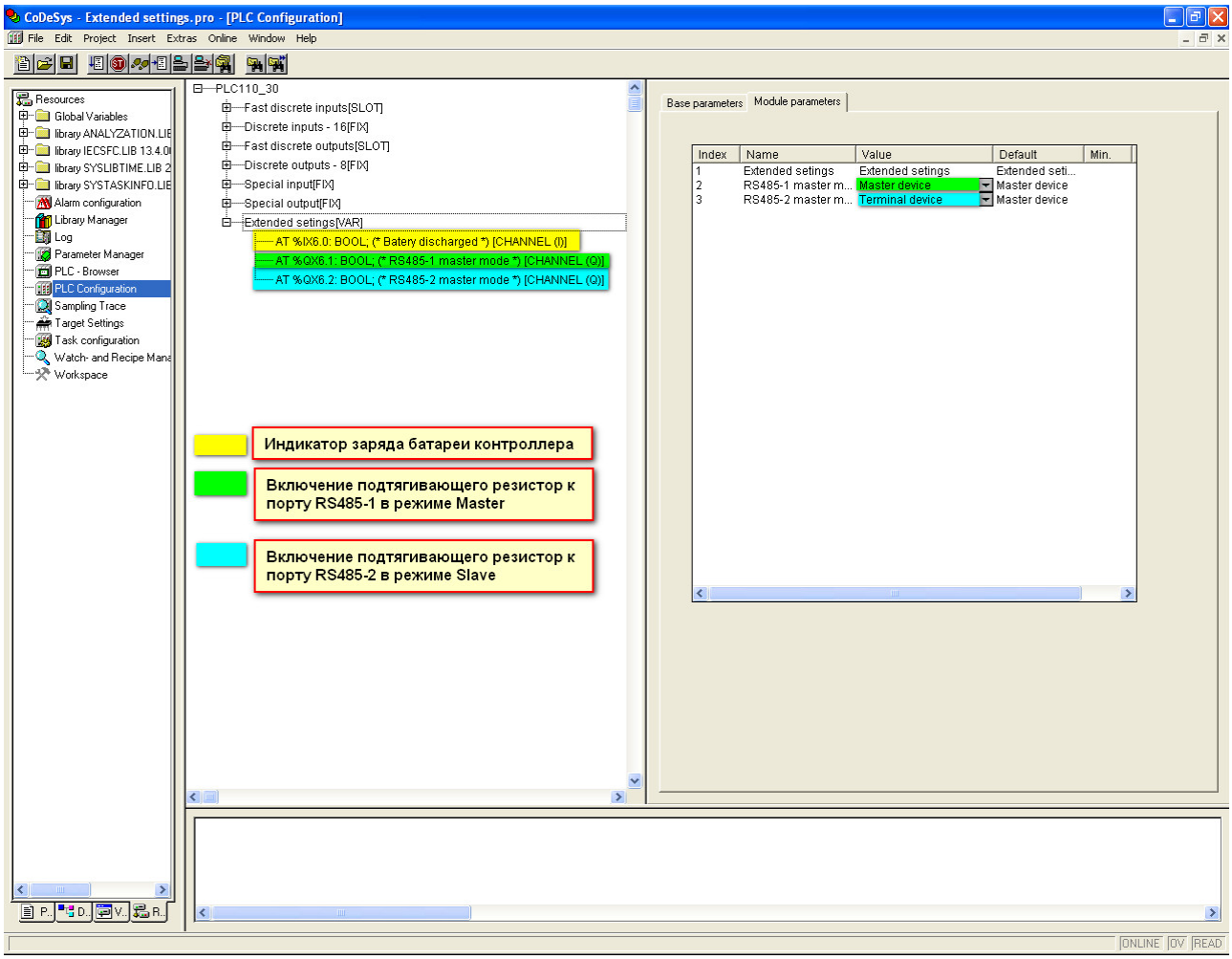


Рисунок 7.51 – Параметры модуля Расширенных настроек (Extended settings)

7.4.12 Настройка коммуникационных интерфейсов модуля

У большинства модулей есть возможность подключения подмодуля интерфейсного порта, через который будут передаваться и/или считываться данные. Интерфейсный порт может быть заменен на требуемый выбором команды «Replace element (Заменить элемент) контекстного меню (см. рисунок 7.52).
Настройка последовательных интерфейсов, интерфейса TCP, модема и интерфейса Empty описана ниже.

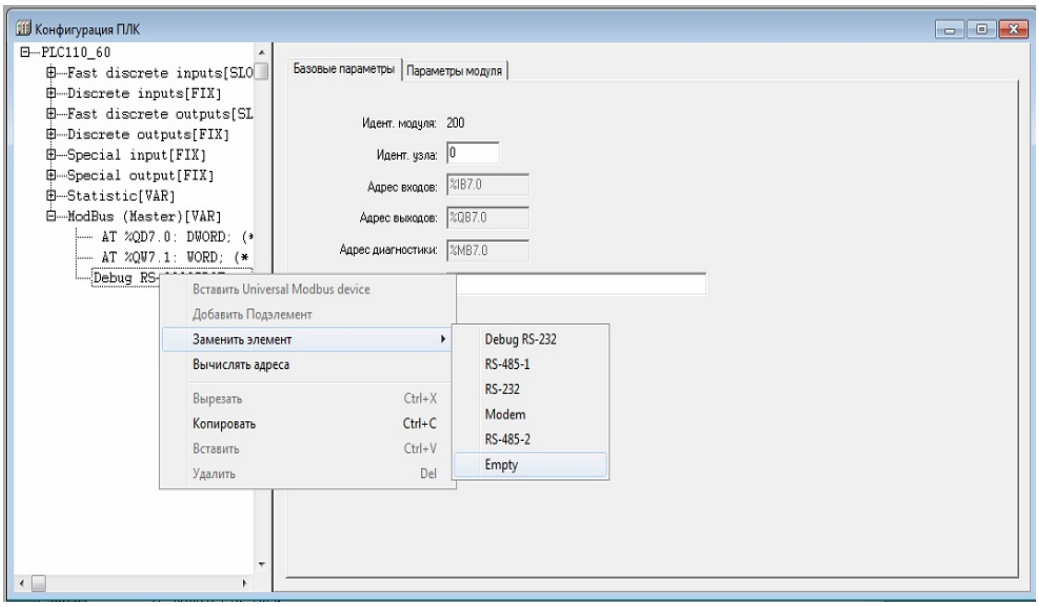


Рисунок 7.52 – Смена интерфейсного порта модуля «ModBus (master)»

7.4.12.1 Подмодули последовательных портов

Параметры последовательных портов DebugRS-232, RS-232, RS-485-1 и RS-485-2, используемых в ПЛК, идентичны (см. рисунок 7.53). При их конфигурировании меняется только название и собственно физический порт, в котором происходит работа. При программировании последовательные порты имеют следующую нумерацию:

Последовательный порт – программный номер порта:

RS485 – 1	-	COM0
RS232	-	COM1
RS485 – 2	-	COM2
RS232 – Debug	-	COM4

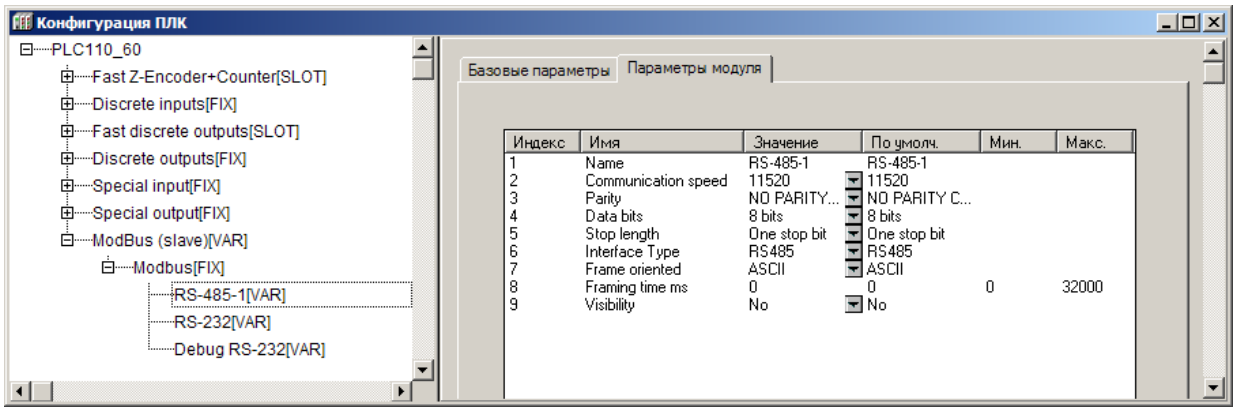


Рисунок 7.53 – Параметры последовательных портов

Параметры последовательного порта:

- **Name** – символическое имя элемента, см. п. 7.3.1.
- **Communication speed (Скорость передачи информации)** – задает скорость передачи информации через последовательный порт. Значения выбираются из списка (2400, 4800, 9600, 14400, 19200, 28800, 38400, 57600, 115200), значение по умолчанию – 115200.
- **Parity (Проверка четности)** – определяет наличие бита четности и его значение (четность, нечетность). Значения выбираются из списка («EVEN, ODD, SPACE, MARK») значение по умолчанию – «NOPARITYCHECK» (отсутствие проверки четности).
- **Data Bits (Количество бит данных)** – задает количество значащих бит в одном байте посылке. Значения в диапазоне от 5 до 8 б, значение по умолчанию – 8.
- **Stop Length (Количество стоп-битов)** – задает количество стоп–битов. Значения выбираются из списка (один, полтора или два стоп–бита), значение по умолчанию – один стоп–бит (Onestopbit).
- **Interface Type (Тип интерфейса)** – задает тип последовательного интерфейса, по которому осуществляется информационный обмен. Задается при выборе подэлемента (**RS-232** или**RS-485)**.
- **Frame Oriented (Тип протокола обмена)** – значения выбираются из списка **«**ASCII**»** и **«**RTU**»**, значение по умолчанию – ASCII [8] .
- **Framing time (Время, на которое необходимо задерживать ответ на запрос в мс)** – задает временную задержку между последним байтом принятого пакета и первым байтом, передаваемым в ответ. Задержка бывает необходима для работы с устройствами с низкими скоростями информационного обмена. Рекомендуемый диапазон значений от 0 до 50 мс (для работы в режиме «slave»), значение по умолчанию – 0 (для работы в режиме «master»).
- **Visibility (Видимость)** – задает видимость параметров модуля в протоколе «Gateway» (в частности, в программе «EasyWorkPLC» разработки ПО «ОВЕН»). Значения выбираются из списка **«yes»** и **«no»,** значение по умолчанию – **«no».**

  Примечание. *Порт DebugRS-232 работает только со следующими настройками:*

  Communication speed *=115200 (значение по умолчанию).*
  Parity *= NOPARITYCHECK*
  Data Bits *= 8*
  Stop Length *= 1*
  Frame Oriented **=** *ASCII[9] (значение по-умолчанию).*

[8] В ПЛК используются следующие типы протоколов обмена: ориентированный на передачу текстовых символовASCII и ориентированный на передачу потока байтов RTU.
В ASCII-режиме информация передается последовательностью символов, и начало и окончание посылки имеют четко обозначенные специальные символы, обычно это – символы решетки, перевода строки и др.
В RTU-режиме иная структура передачи информации: передаются байты, несущие полезную информацию, без какого-либо указания начальных и/или конечных границ (заголовочных и конечных байтов). Сама посылка и ее границы определяются по наличию разрыва. Если время разрыва превышает определенное время (например, для Modbus – время передачи 3,5 символов), – устройствоопределяет, что посылка закончилась, началась другая посылка. Таким образом, посылки отделяются другот друга и их можно идентифицировать.

[9] В ПЛК используются следующие типы протоколов обмена: ориентированный на передачу текстовых символов ASCII и ориентированный напередачу потока байтов RTU.

Внимание! *Порт* ***Debug RS-232*** *в модуле* ***ModBus*** *(****Master)*** *может работать только в режиме ASCII, работа в режиме RTU невозможна.*

7.4.12.2 Порт TCP

Помимо последовательного порта в конкретной системе устройств может использоваться порт TCP.

Параметры порта TCP (см. рисунок 7.54):

- **Name** – символическое имя элемента, см. п. 7.3.1.
- **Remote Port (Удаленный порт)** – задает адрес удаленного порта. Значения устанавливаются в диапазоне от 0 до 65535, значение по умолчанию – 502.
- **Visibility (Видимость)** – задает видимость параметров модуля в протоколе «Gateway» (в частности, в программе «EasyWorkPLC» разработки ПО «ОВЕН»). Значения выбираются из списка **«yes»** и **«no»,** значение по умолчанию – **«no».**

Примечание. *Обращение мастера сети к ПЛК по Ethernet происходит по базовым сетевым настройкам ПЛК, заводским или пользовательским. Изменить сетевые настройки ПЛК можно в режиме «ПЛК-Браузер (PLC-Brauser)» (см. приложение Ж). В данном случае модуль использует MAC-адрес и IP-адрес контроллера.*

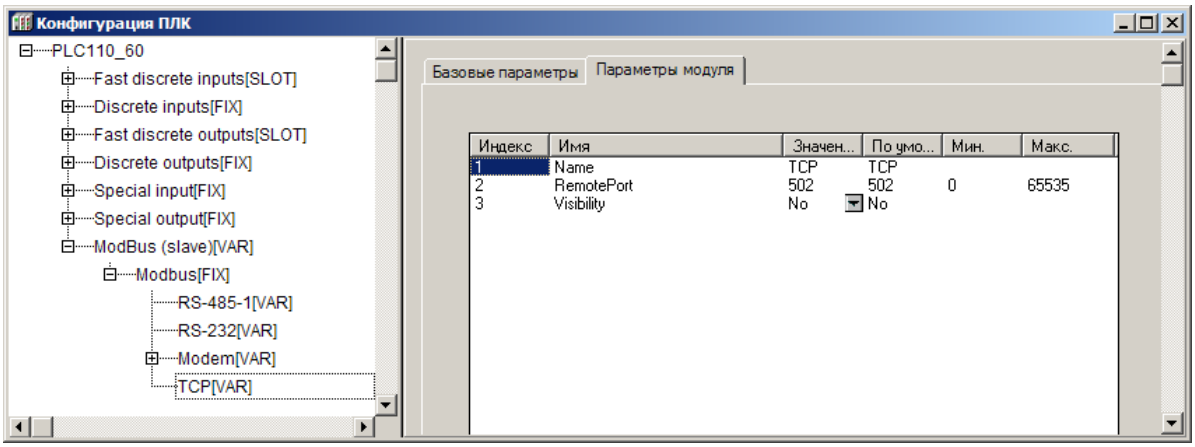


Рисунок 7.54 – Параметры порта TCP

7.4.12.3 Порт Empty

В ряде модулей, работающих в режиме master, также может использоваться порт Empty. Данный порт используется при передачи данных по Ethernet и освобождения последовательных портов при работе по TCP.

В ASCII-режиме информация передается последовательностью символов, и начало и окончание посылки имеют четко обозначенные специальные символы, обычно это – символы решетки, перевода строки и др.

В RTU-режиме иная структура передачи информации: передаются байты, несущие полезную информацию, без какого-либо указания начальных и/или конечных границ (заголовочных и конечных байтов). Сама посылка и ее границы определяются по наличию разрыва. Если время разрыва превышает определенное время (например, для Modbus – время передачи 3,5 символов), – устройство определяет, что посылка закончилась, началась другая посылка. Таким образом, посылки отделяются друг отдруга и их можно идентифицировать.

7.4.12.4 Настройка модемного подключения

В качестве интерфейса в контроллере может также выступать модем. В зависимости от используемого режима модема указывается различные настройки. Ниже рассмотрены основные режимы работы модема и работы с ним.
Особенностью обновленного ПЛК является реализация поддержки протоколов стека PPP через модем (GPRS). Поддерживается 1 сетевой интерфейс PPP (№1) и 1 модем на любом из портов ПЛК.
Маршрутизация пакетов из Ethernet(сетевой интерфейс №0) в PPP и обратно в автоматическом режиме не поддерживается.
Для настройки PPP соединения необходимо отредактировать файл local_addres.dat в соответствии с настройками порта, типом модема, адресом точки доступа и настройками протокола

7.4.12.4.1 Режим GPRS

При работе с модемом в режиме GPRS не требуется установка PPP-драйвера, как было в прежних модификациях, а также рекомендуется устанавливать порт Empty (см. рисунок 7.55).

Для корректной работы модема в режиме GPRS требуется произвести корректировку файла ПЛК, а именно local_addres.dat. Более подробную настройку смотрите дополнительный документ «Использование протокола PPP в ПЛК110М».

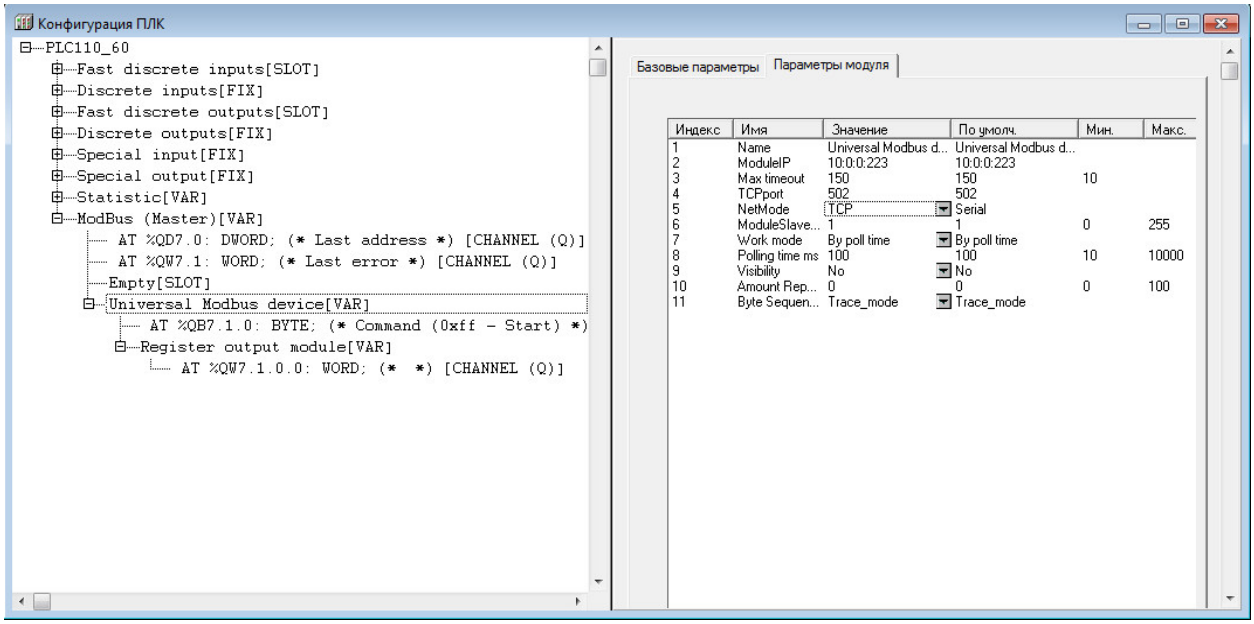


Рисунок 7.55 – Конфигурация модуля при использовании модема

8 Настройка дополнительных устройств

ПЛК110 содержат ряд дополнительных устройств (таких как часы реального времени и др.), настройка которых осуществляется в режимах «Конфигурация ПЛК (PLC Configuration)» и «ПЛК-браузер (PLC Browser)» ПО CoDeSyS.

В этих же режимах могут настраиваться поддерживаемые в ПЛК110 интерфейсы или протоколы обмена и конфигурирования.

Подробнее работа режиме «ПЛК-браузер (PLC Browser)» ПО CoDeSyS описана в приложении Ж.

8.1.1 Задание значения часов реального времени

Часы реального времени в ПЛК110 имеют независимый источник питания[10] и продолжают отсчет времени при выключении основного питания контроллера. Время, в течение которого встроенные часы будут идти при отсутствии внешнего питания ПЛК, зависит от состояния встроенного источника питания и условий хранения. Так, при хранении в помещении с комнатной температурой и новом источнике питания время работы часов составит 5 лет. Поэтому задание их значения следует производить однократно, либо после длительного (более 6 мес.) неиспользования ПЛК.

Задание часов производится в режиме «ПЛК-браузер (PLC Browser)» ПО CODESYS командами «**SetTime»** (Задание времени), «**SetDate»** (задание календарной даты).

Команды могут быть введены в командной строке режима вручную или выбором стандартных команд из перечня, ызываемого нажатием кнопки с тремя точками, расположенной у правого края командной строки (см. рисунок 4.49).

Просмотр значения времени и даты осуществляется командой **GetTime**.

Примечание: Также установить или считать значение времени с часов можно с помощью соответствующей библиотеки. Описание см. файл… SysLibTime

8.1.2 Задание настроек порта Ethernet

Для корректной работы порта Ethernet в сети необходимо задать ему необходимый IP-адрес, а также маску подсети IP.

Задание этих параметров производится в режиме «ПЛК-браузер (PLC Browser)» ПО CoDeSyS командами **SetIP** и **SetMask**.

Измененные значения вступают в силу только после перезагрузки ПЛК.

Значения этих параметров сохраняются в файле local_addres.dat на внутреннем Flash-диске ПЛК110. Этот файл при желании может быть считан с контроллера на ПК. Для этого надо вызвать команду **Online | Readfile from PLC** главного меню ПО CoDeSyS**.**

Текущие настройки интерфейса можно посмотреть в режиме «ПЛК-браузер (PLC Browser)» ПО CoDeSyS по команде «**PLCInfo»** (см. рисунок 8.1).

[10] Тип источника питание указан в «Руководстве по эксплуатации».

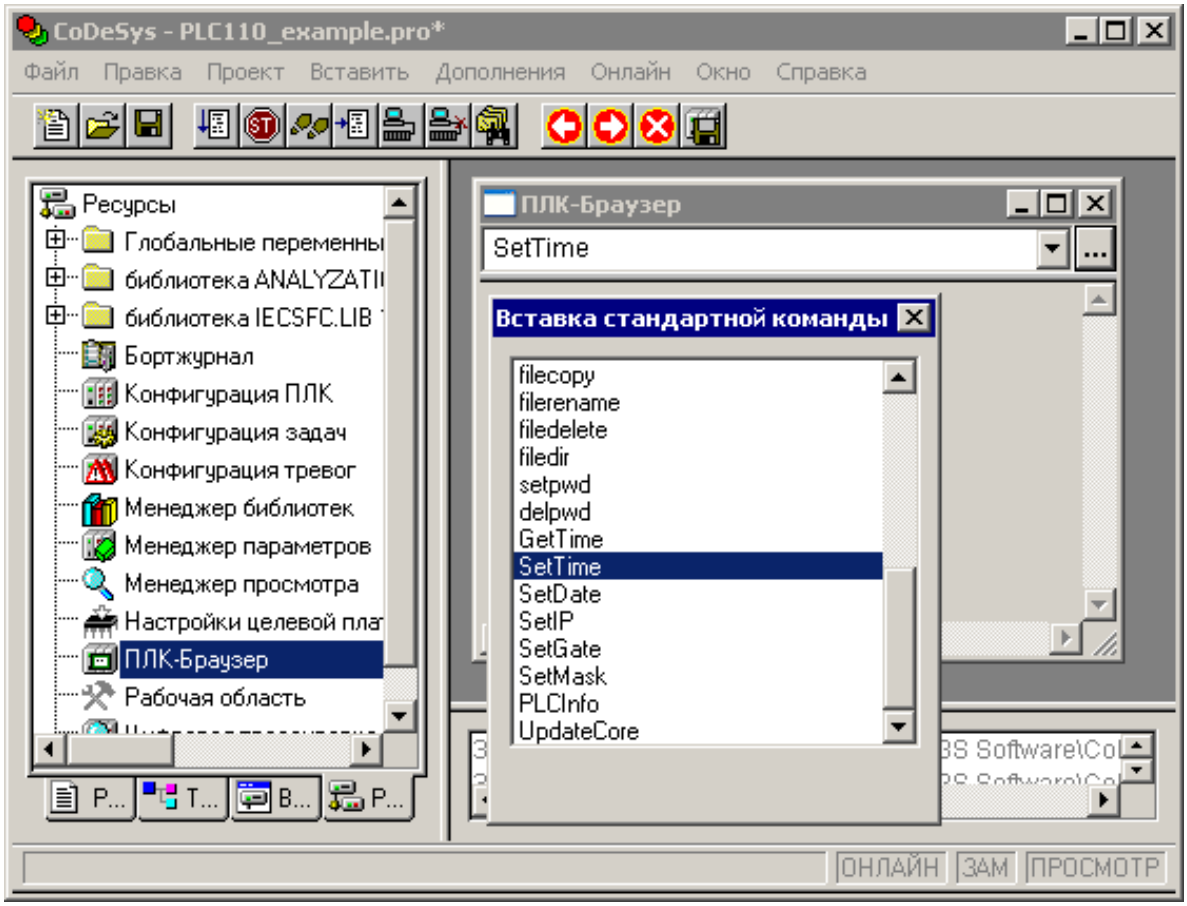


Рисунок 8.1 – Режим «ПЛК-Браузер» ПО CODESYS. Задание значения реального времени

9 Работа с высокочастотным таймером

ПЛК110 имеет встроенный таймер, по прерыванию которого может быть вызвана отдельная программа, не связанная с выполнением основной программы ПЛК. Пример такой программы приведен в файле «hi_timer.pro», включаемом в состав дистрибутивного диска; описание этого примера и некоторых приемов работы с ПО CODESYS приведено в данном разделе.

Перед открытием проекта его следует скопировать на жесткий диск компьютера.

В программе, вызываемой по прерыванию встроенного таймера, могут обрабатываться состояния «быстрых» входов и выходов ПЛК (Fast inputs & Fast outputs). Подробно о количестве «быстрых» входов и выходов см. Руководство по эксплуатации ПЛК.

Такой режим обработки может потребоваться для задач, время обработки которых должно быть существенно меньше времени цикла ПЛК или для автоматизации объектов, критичных ко времени реакции на определенные события. Минимальный период вызовов прерываний таймера составляет 20 мкс и может быть увеличен при вызове функции инициализации (при этом период должен быть кратен 20 мкс).

Рассмотрим порядок создания и подключения программы обработки прерывания таймера на примере, содержащемся в файле «hi_timer.pro».

Пусть имеется объект, требующий выключение исполнительного механизма при замыкании дискретного датчика. Датчик подключен к «быстрому» входу 1 ПЛК110, исполнительный механизм управляется «быстрым» выходом 1 того же контроллера. Пусть также требуется вычислять, сколько таких переключений произошло между началами основных циклов ПЛК.

Основные операции процедуры создания программы обработки прерывания таймера таковы:

1) Создать проект. Процедура описана в разделе 3 . В примере используется контроллер ПЛК110-60.К-М; основная программа контроллера – PLC_PRG будет написана на языке ST.
2) Перейти в режим POU ПО CODESYS; для входа в режим следует перейти на вкладку «POU» Организатора объектов ПО.
3) Выбрать команду «**Add Object...**» контекстного меню вкладки «POU» Организатора объектов или команду «**Проект | Объект | Добавить**» главного меню.
4) Воткрывшемся окне (см. рисунок 9.1) задать тип, имя и язык написания POU: добавляемый POU должен быть типа «Программа (Program)», имя и язык написания могут быть любыми, в примере выбран язык ST и дано имя Timer_POU.

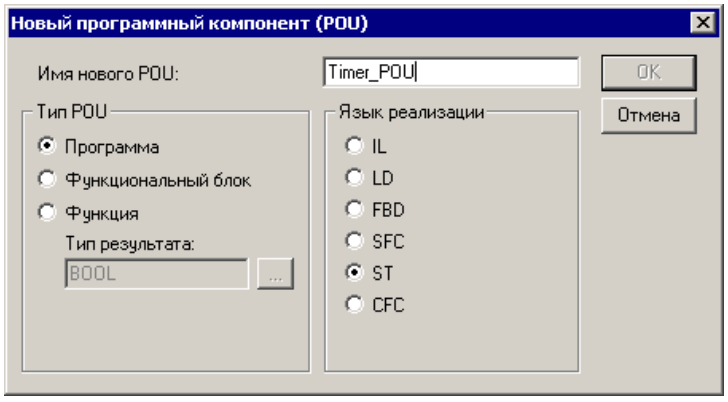


Рисунок 9.1 – Добавление POU

5) Для подключения POU к прерыванию таймера следует перейти в режим «Конфигурация задач (Task Configuration)» ПО CODESYS; для входа в режим следует перейти на вкладку «Ресурсы» Организатора объектов ПО и выбрать строку «Конфигурация задач (Task Configuration)». Откроется окно режима (см. рисунок 9.2).
В левой части окна режима – выделить строку «Системные события (System events)». В правой части окна отобразится доступный для выбраного контроллера список прерываний «Системные события (System events)». Следует выбрать прерывание «Timer» (установить флажок в поле соответствующего переключателя) и в колонке «Вызываемый POU (called POU)» указать имя созданного POU (в примере – «Timer_POU»).
Использовать кнопку Create POU не следует, т.к. она создает POU типа Function, а для работы требуется POU типа Program.

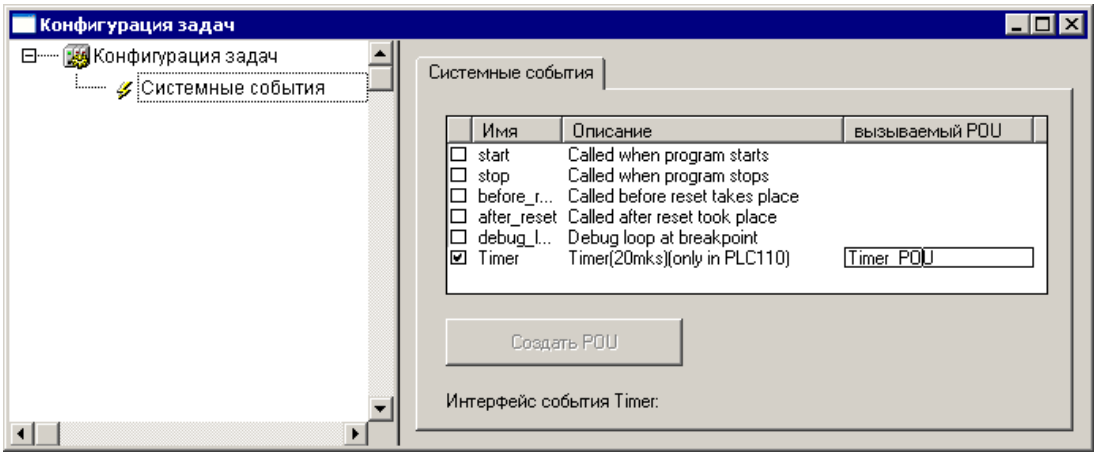
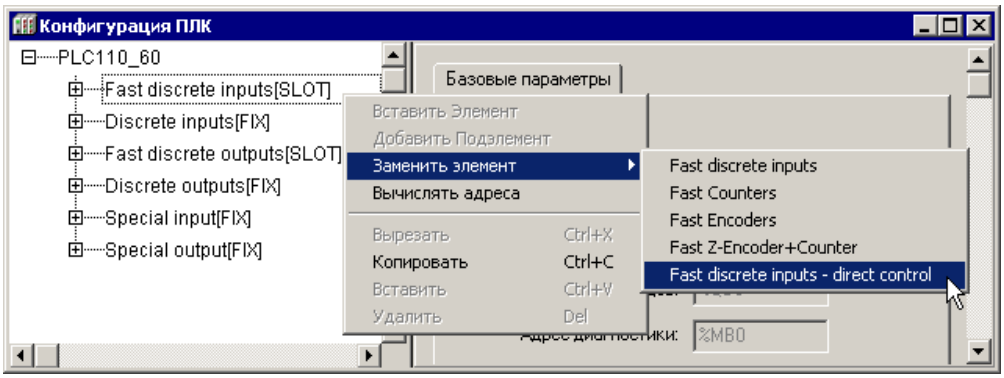


Рисунок 9.2 – Окно режима «Конфигурация задач (Task Configuration)»

6) Перевести «быстрых» дискретные входы и выходы в режим прямого управления (управления из POU).
Это следует сделать в связи с тем, что т.к. прерывание таймера вызывается в несколько десятков раз чаще, чем происходит цикл ПЛК, то использовать в POU обработки прерываний таймера значения из области памяти ввода/вывода не имеет смысла.
Для исключения ситуации, когда одним и тем же входом или выходом управляет POU обработки прерывания таймера и основная программа ПЛК, следует перевести «быстрые» входы и выходы в специальный режим: режим прямого управления (Direct Control mode). В этом режиме для «быстрых» входов и выходов не выделяется каналов в памяти ввода/вывода.
Для перевода выходов в режим прямого управления необходимо перейти в режим «Конфигурация ПЛК (PLC Configuration)» и заменить модуль «Fast Discrete Outputs» на модуль «Fast Discrete Outputs – Direct Control» и модуль «Fast Discrete Inputs» – на модуль «Fast Discrete Inputs – Direct Control» (см. рисунок 9.3).

а)



б)

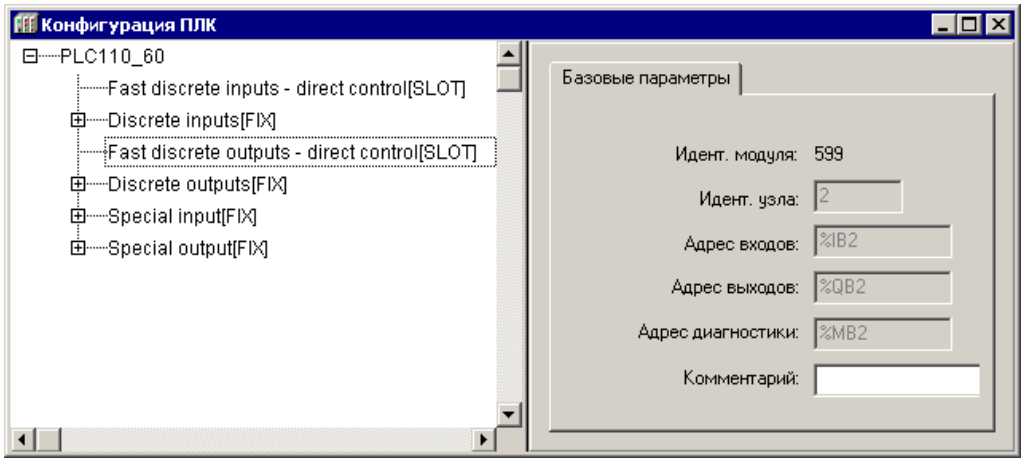


Рисунок 9.3 – Окно режима «Конфигурация ПЛК (PLC Configuration)». Операция (а) и результат (б) замены модулей конфигурации ПЛК

7) Подключить требуемые библиотеки.
Запуск работы таймера производится из основной программы посредством вызова функции, входящей в состав библиотеки Timer.lib (одна из библиотек ОВЕН, поставляемых в комплекте с ПЛК).
Работа с «быстрыми» входами и выходами из POU обработки прерываний таймера выполняется не через пространство памяти ввода/вывода, а посредством вызова функций библиотеки SysLibPorts.lib (библиотека CODESYS, поставляется в комплекте с ПЛК).
Подключение библиотек производится в режиме «Менеджер библиотек (Library Manager)». Процедура подключения описана в разделе 4.2.4 .

8) Написать программу POU обработки высокочастотного таймера. Для этого – перейти в режим «POU», выбрав в списке POU проекта требуемое окно и написать программу обработки прерывания таймера.

Внимание! *В POU обработки прерывания таймера не допускается выполнения сложных математических действий или вызов ресурсоемких функций, т.к. они могут не успеть завершиться до следующего вызова прерывания таймера.*
В описываемом примере программа состоит из чтения состояния «быстрых» входов функцией SysPortIn (из библиотеки SysLibPorts.lib), инвертировании прочитанного значения и передаче его в «быстрые» выходы функцией SysPortOut (см. рисунок 9.4).
Для обеих функций работа ведется с портом 0, с младшими битами. Число ликвидных бит равно числу «быстрых» входов и выходов используемого контроллера.

Внимание! *При отладке прерывания таймера запрещается использовать точку остановки (breakpoint), т.к. это приведет к зависанию контроллера.*

```
Timer_POU (PRG-ST)
0001 PROGRAM Timer_POU
0002 VAR
0003     in: BYTE;
0004     last_in:BYTE;
0005     old_res:DWORD;
0006 END_VAR
0007
0001 (*!! Запрещается использовать breakpoints при отладке в программах, обслуживающих прерывания !!*)
0002
0003 (*Сброс подсчитанного количества срабатываний при начале нового цикла ПЛК  -- пример чтения из  глобальной переменной *)
0004 (* Глобальную переменную t_res в этом обработчике можно только читать, т.к. значения в нее пишутся в основной программе *)
0005 IF (t_res<>old_res) THEN (*Если в основном цикле ПЛК переменная t_res была изменена, то...*)
0006     old_res:=t_res; (*.... запоминаем новоезначение t_res в переменную old_res*)
0007     t_count:=0; (* ... и обнуляем счетчик срабатываний*)
0008 END_IF
0009
0010 (*Чтение "быстрых" входов и запись "быстрых" выходов*)
0011 (*Не забудьте изменить в окне PLC Configuration модули "быстрых" входов и выходов на модули Direc tControl *)
0012 in:=SysPortIn(0); (*Чтение "быстрых" входов из порта 0*)
0013 SysPortOut(0, NOT(in)); (*Запись "быстрых" выходов в порт 0 инвертированным значением входов*)
0014
0015 (*Подсчёт числа изменений состояний выходов за цикл ПЛК -- пример записи глобальных переменных*)
0016 IF (in<>last_in) THEN   (*Если состояние входов изменилось, то...*)
0017     t_count:=t_count+1; (* ... увеличиваем счетчик *)
0018     last_in:=in; (*... и запоминаем текущее состояние*)
0019 END_IF
0020
0021
```

Рисунок 9.4 – Программа обработки высокочастотного таймера («Timer_POU»)

Для передачи информации о количестве срабатываний «быстрых» входов между вызовами циклов ПЛК из обработчика прерывания в основную программу используются глобальные переменные (см. рисунок 9.5). Работа с глобальными переменными производится в режиме «Глобальные переменные (Global Variables)», для входа в который следует выбрать пункт «Глобальные переменные (Global Variables)» Организатора объектов ПО CODESYS.

При работе с глобальными переменными следует придерживаться следующего правила: запись значения в глобальную переменную должна происходить только в одном месте: либо в POU обработки прерывания, либо в основной программе.

```
Global_Variables
0001 VAR_GLOBAL
0002     t_count:DWORD; (*Число изменений состояния "быстрых" дискр. входов за цикл ПЛК*)
0003     t_res:DWORD;     (*Сигнал сброса счетчика t_count*)
0004
0005 END_VAR
0006
0007
0008
```

Рисунок 9.5 – Глобальные переменные, используемые программой обработки высокочастотного таймера («Timer_POU»)

9) Написать основную программу ПЛК. Для этого следует, перейдя в ре-

жим «POU», выбрать в списке объектов основную программу «PLC_PRG». В окне программы (см. рисунок 9.6) – записать вызов функции инициализации прерывания таймера SetIRQ (из библиотеки Timer.lib). Аргументом функции является значение периода вызова прерывания в микросекундах (период должен быть кратен 20). Вызов функции необходимо делать однократно, при старте программы (пример кода представлен в проекте hi_timer.pro, см. рисунок 6.6).
Для передачи информации в обработчик прерывания таймера используются глобальные переменные (см. п. 8). Для них также действует правило, предписывающее запись переменной только в одном POU.

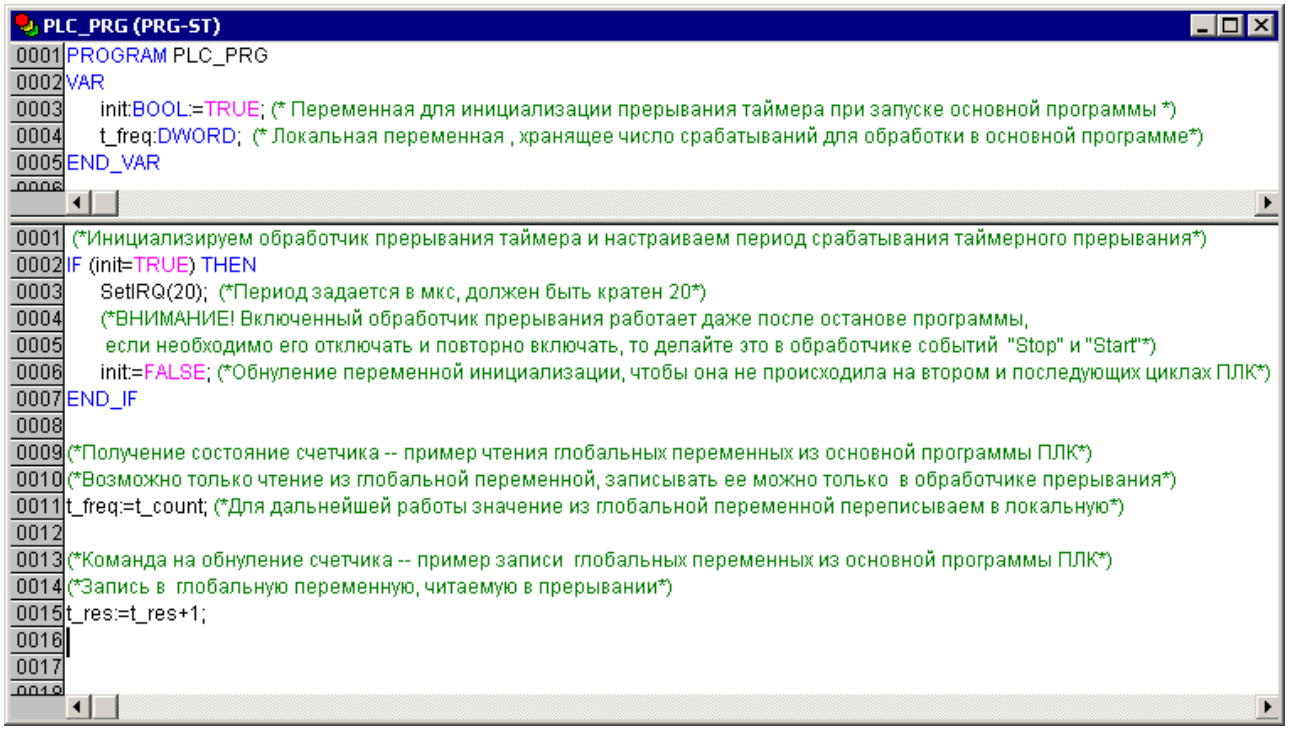


Рисунок 9.6 – Программа ПЛК, обращающаяся к функции инициализации прерывания таймера «SetIRQ»

10 Обновление встроенного ПО микроконтроллера и Target-файлов

Компания ОВЕН совершенствует производимые контроллеры и их программное обеспечение, в том числе – встроенное ПО микроконтроллера и Target-файлы ПЛК.

Обновленное ПО микроконтроллера и Target-файлы следует устанавливать на используемый ПЛК только в том случае, если наблюдаются сбои в работе действующего ПО. Если ПЛК работает без сбоев, то производить обновление программного обеспечения не рекомендуется.

Обновленное ПО и соответствующие ему версии Target-файлов можно скачать с сайта www.owen.ru.

Внимание!
Обновление ПО микроконтроллера и Target-файл ПЛК должны иметь соответствующие версии!

Приступая к обновлению ПО микроконтроллера, следует учесть, что:

Если на ПЛК установлено ПО микроконтроллера версии 2.02.0 или более поздняя, то контроллер подготовлен к возможности перепрограммирования ядра ПЛК в процессе эксплуатации ПЛК без его разборки (без снятия корпуса).

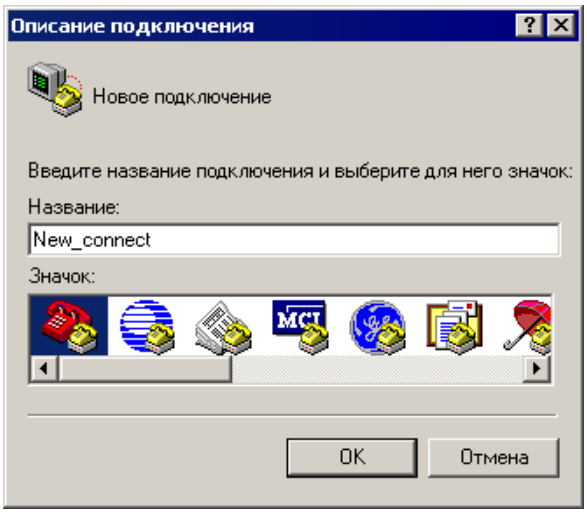
10.1 Определение актуальной версии ПО микроконтроллера

Определение актуальной версии ПО микроконтроллера может быть выполнено двумя способами.

10.1.1 Определение версии ПО микроконтроллера с использованием «гипертерминала»

Для определения актуальной версии ПО микроконтроллера с использованием «гипертерминала» следует:

1) Соединить ПК с контроллером через последовательный порт «RS-232 DEBUG».
2) Выбрать команду «Пуск | Программы | Стандартные | Связь | HyperTerminal».
3) В открывшемся окне «Описание подключения» (см. рисунок 10.1, а) – в поле «Название» задать имя нового подключения и нажать кнопку «ОК»
4) В открывшемся окне «Подключение» (см. рисунок 10.1, б) – в поле «Подключаться через» выбором из раскрывающегося списка задать COM-порт, к которому подключен ПЛК, и нажать кнопку «ОК».
5) В открывшемся окне «Свойства: COM» (см. рисунок 10.2) – в поле «Скорость (бит/с» выбором из раскрывающегося списка задать значение «115200», в поле «Управление потоком» выбором из раскрывающегося списка задать значение «Нет», нажать кнопку «Применить», затем – нажать кнопку «ОК».
6) Нажатием кнопки «Сброс» перезагрузить ПЛК.
7) В открывшемся окне «HyperTerminal» (см. рисунок 10.3) в строке «Binary Version» отобразится номер актуальной версии ПО микроконтроллера.

а) 

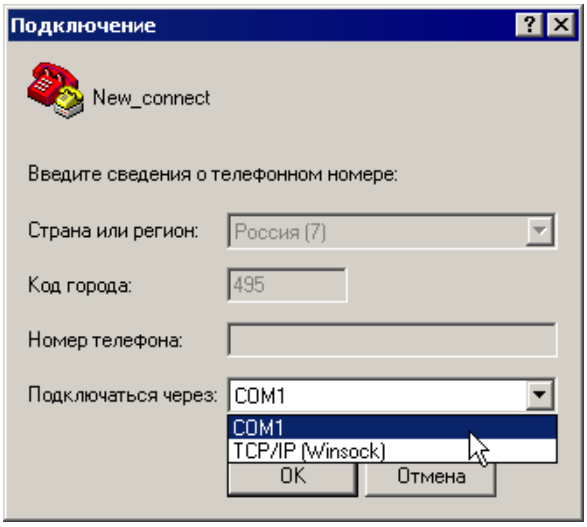
б) 

Рисунок 10.1 – Окна «Описание подключения» (а) и «Подключение» (б)

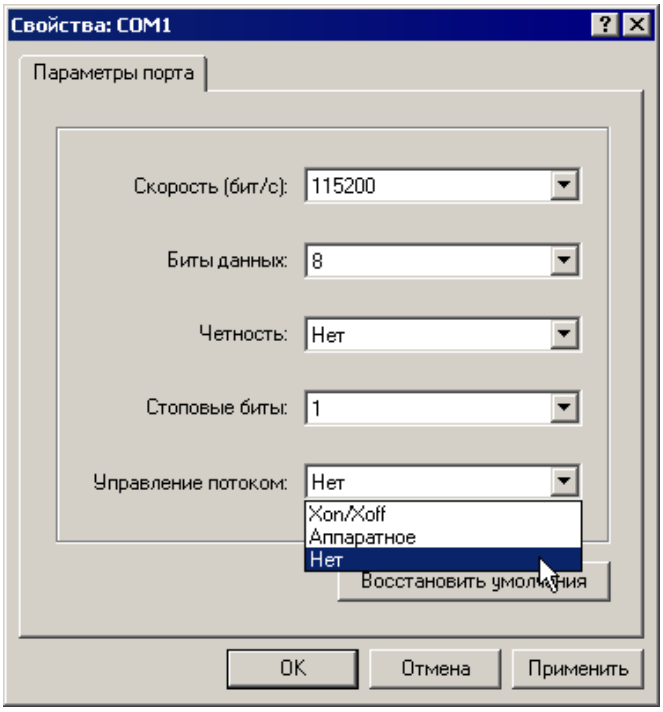


Рисунок 10.2 – Окно «Свойства COM-порта»

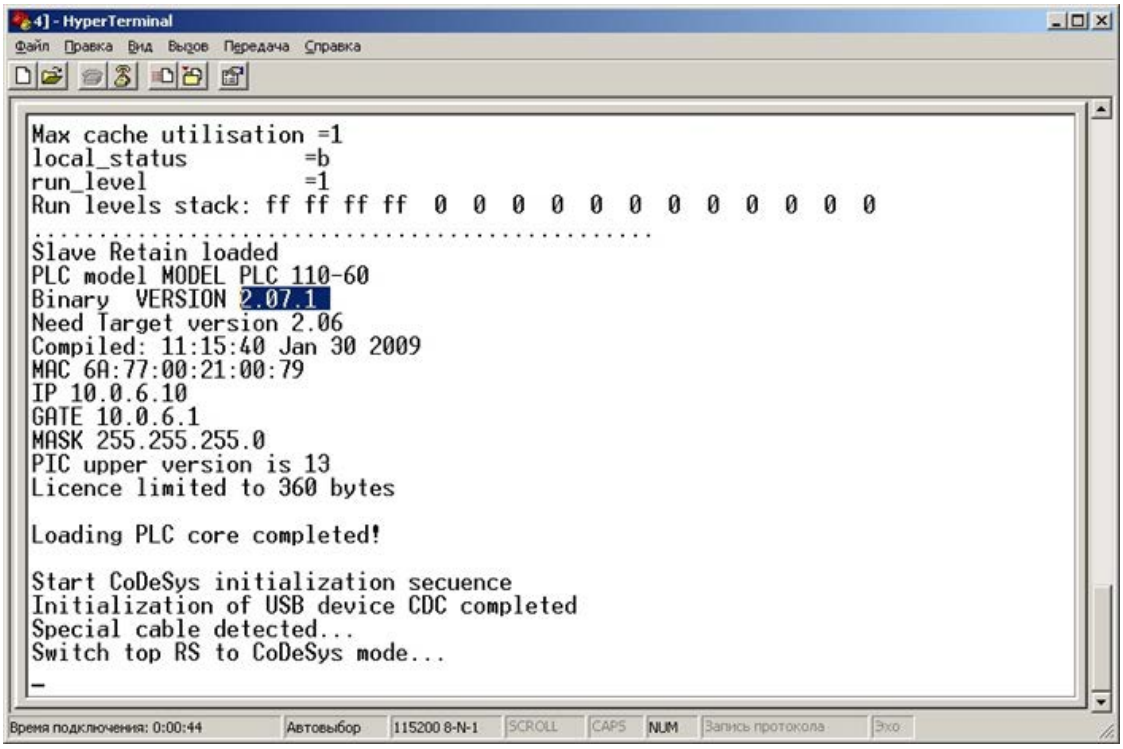


Рисунок 10.3 – Окно «HyperTerminal»

10.1.2 Определение версии ПО микроконтроллера с использованием ПО CODESYS

Для определения актуальной версии ПО микроконтроллера с использованием ПО CODESYS следует:

1) Соединить ПК с контроллером через любой из портов для программирования (COM-порт, порт Ethernet или USB).
2) Запустить ПО CODESYS.
3) Выбрать команду «Онлайн | Подключение (Online | Login)» главного меню.
4) Перейти на вкладку «Ресурсы» организатора объектов и войти в режим «ПЛК Браузер (PLC-Browser)» (подробнее о режиме см. приложение Ж).
5) Выбрать команду «PLCInfo». В поле отображения реакции ПЛК на введенную команду отобразится информация о ПЛК. В строке «Binary Version» отобразится номер актуальной версии ПО микроконтроллера (см. рисунок 10.4).

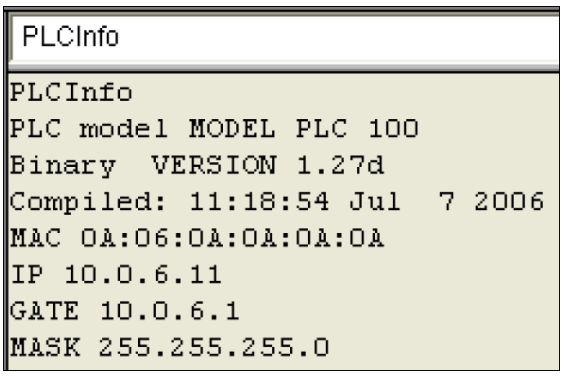


Рисунок 10.4 – Окно «ПЛК Браузер (PLC-Browser)»

10.2 Обновление ПО микроконтроллера

Обновление ПО микроконтроллера может быть выполнено двумя способами.

Первый способ – с использованием специализированного файла обновленного ПО микроконтроллера (например, UpdatePLC110_60.bin) и стандартных функций ПО CODESYS.

Второй способ – использование утилиты «Перепрошивка ПЛК <Версия>.exe» (имеющейся на дистрибутивном диске ПЛК). Второй способ рекомендуется использовать, если сбой в работе ПЛК привел к нарушению связи ПЛК и ПО CODESYS.

И специализированные файлы обновленного ПО микроконтроллера, и утилиты «Перепрошивка ПЛК» расположены в папке «Patching» дистрибутивного диска.

10.2.1 Обновление ПО микроконтроллера с использованием ПО CODESYS

Внимание!

Если на ПЛК записана прошивка версии более ранней, чем 2.02.0, то ПЛК следует предварительно подготовить к возможности перепрограммирования ядра:

1) снять корпус контроллера;
2) удалить перемычку на средней плате ПЛК;
3) установить корпус контроллера.

Для обновления ПО микроконтроллера с использованием ПО CODESYS следует:

1) Записать на жесткий диск ПК файл обновленного ПО микроконтроллера (например, UpdatePLC110_60.bin).
2) Подключить питание ПЛК.
3) Соединить ПК с контроллером через любой из портов для программирования (COM-порт, порт Ethernet или USB).
4) Запустить ПО CODESYS.
5) Выбрать команду «Онлайн | Подключение (Online | Login)» главного меню.
6) Записать в память ПЛК файл обновленного ПО микроконтроллера: выбрать команду «Онлайн | Записать файл в ПЛК (Online | Write File to PLC)». Требуемый файл будет опознан ПО автоматически [11] .
7) Перейти на вкладку «Ресурсы» организатора объектов и войти в режим «ПЛК Браузер (PLC-Browser)» (подробнее о режиме см. приложение Ж).
8) Выбрать команду «UpdateCore». В поле отображения реакции ПЛК на введенную команду отобразится сообщение «UpdateOK» (см. рисунок 10.5).
9) Нажать кнопку «Сброс».
10) На этом процедура обновления ПО микроконтроллера завершается. Далее следует перейти к установке target-файла.

[11] Для сохранения в памяти ПЛК файла обновленного ПО микроконтроллера можено также воспользоваться утилитой «PLC_IO.exe» (имеющейся на дистрибутивном диске ПЛК). Для этого следует:
1) скопировать на жесткий диск ПК утилиту«PLC_IO.exe» и файл «UpdatePLCxxx.bin»для конкретного ПЛК;
2) в командном файле PLC_IO.bat – модифицировать IP адрес или номер COM-порта, по которому утилита будет загружать файл в ПЛК.
3) выполнить командный файл PLC_IO.bat. После запуска, если ПЛК правильно подсоединен и правильно указаны его настройки в PLC_IO.bat,программа соединится с контроллером и запишет соответствующий файл обновления.

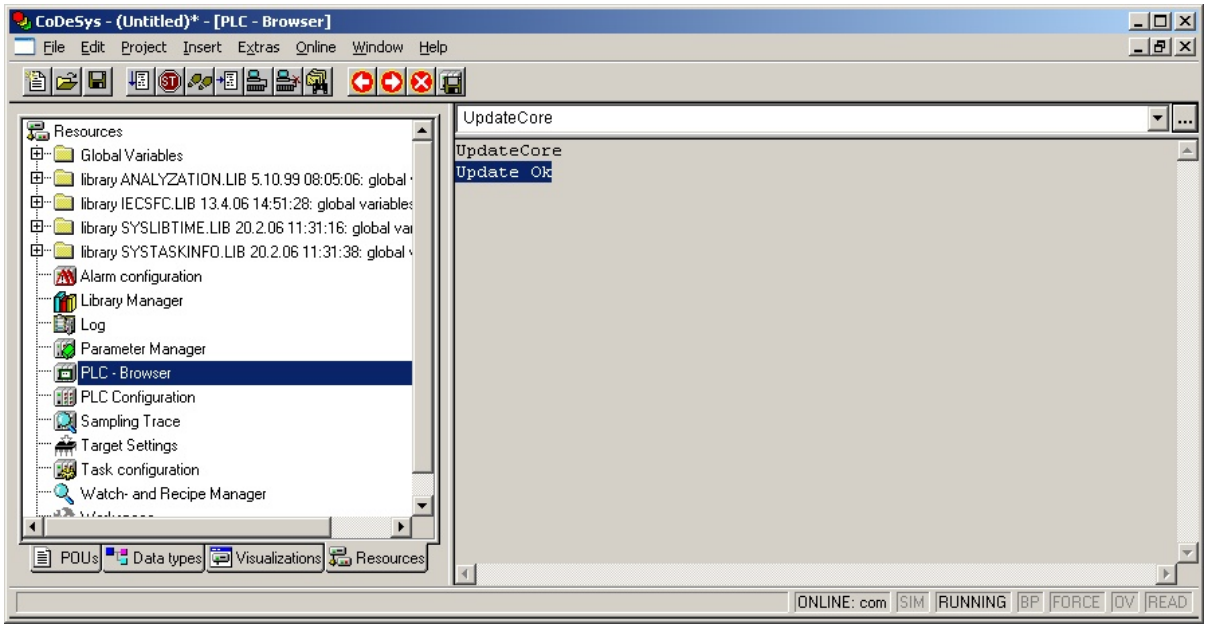


Рисунок 10.5 – Окно «ПЛК Браузер (PLC-Browser)»

10.3 Обновление Target-файла

Для обновления Target-файла ПЛК следует:

1) Скопировать на жесткий диск ПК требуемый Target-файл для конкретного ПЛК. При выборе Target-файла следует обратить внимание на то, что имя Target-файла не полностью совпадает с наименованием контроллера: в наименовании контроллера использована кириллица (например, ПЛК110), а в названии Target-файла – латиница (например, PLC110). Для каждой модификации ПЛК в поставку включен соответствующий Target-файл. Так, для ПЛК110 на дистрибутивном диске размещены Target-файлы PLC110.30-L_v2, PLC110.30-M_v2, PLC110.32-L_v2, PLC110.32-M_v2, PLC110.60-L_v2, PLC110.60-M_v2.
2) Извлечь Target-файл (и входящие в его комплект файлы) из архива.
3) Удалить старую версию файла.
4) Установить требуемую версию Target-файла.
   Процедуры удаления (деинсталляции) и установки (инсталляции) Target-файлов описаны в разделе 3.2 .

Внимание!
После обновления ПО микроконтроллера ПЛК продолжает функционировать в прежнем режиме, т.е. использует настройки старого ПО. Новое ПО вступит в силу только после перезагрузки ПЛК. Перезагрузку производить отключением питания с повторным включением через пять и более секунд.

Приложение А. Интерфейс ПО CODESYS

А.1 Основные режимы (Редакторы) ПО CODESYS

Элементы управления ПО CODESYS приведены в таблице А.1.

Таблица А.1 – Элементы управления программой

Команда меню	Кнопка панели инструментов	Горячие клавиши	Описание команды
Файл (File)			
Создать (New)			Создает новый проект. Новый проект получает имя «Untitled».
Создатьпошаблону (New from template)			Открывает окно выбора файла, в котором следует выбрать требуемый файл проекта (*.pro), который послужит шаблоном нового проекта. Новый проект получает имя «Untitled».
Открыть (Open)		<Ctrl+O>	Открывает ранее сохраненный проект. Если в момент вызова этой команды какой-то проект уже открыт и в него были внесены изменения, то программа предложит сохранить этот проект
Закрыть (Close)			Закрывает открытый в данный момент проект. Если с момента открытия в проект были внесены изменения, то программа предложит сохранить этот проект
Сохранить (Save)		<Ctrl+S>	Сохраняет файл проекта
Сохранить как… (Save as…)			Сохраняет проект или библиотеку под новым именем. При этом исходный файл не изменяется
Сохранить/отправить архив (Save/Mail Archieve…)			Создает архив проекта. Все файлы, которые используются проектом CODESYS, сохраняются и сжимаются в файл с расширением *.zip, который удобно хранить и пересылать по электронной почте.
Печать (Print)		<Ctrl+P>	Печатает содержание активного окна
Параметры печати (Printer Setup…)			Открывает окно настройки печати
Выход (Exit)			Закрывает CODESYS. Если в момент вызова этой команды открыт проект, то программа предложит его сохранить
Правка (Edit)			
Отменить (Undo)		<Ctrl+Z>	Отменяет последнее изменение, сделанное в открытом редакторе или в Организаторе объектов. Используя эту команду, вы можете отменить все изменения, выполненные после открытия окна
Вернуть (Redo)		<Ctrl+Y>	Возвращает последнее изменение, отмененное в открытом редакторе или в Организаторе объектов командой Undo
Cut – Вырезать		<Ctrl+X>	Перемещает выделенный элемент в буфер. При этом выделенный элемент удаляется из окна редактора
Copy – Копировать		<Ctrl+C>	Копирует выделенный элемент в буфер, содержимое окна редактора при этом не изменяется

Paste – Вставить		<Ctrl+V>	Вставляет содержимое буфера, начиная с текущей позиции курсора в окне редактора. В графических редакторах команда выполняется только если содержимое буфера соответствует выбранному элементу
Delete – Удалить		<Delete>	Удаляет выбранную область, содержимое буфера при этом не изменяется
Find… – Найти		<Ctrl+F>	Находит введенный текст в активном окне редактора. Открывает диалог поиска
Find next – Найти далее		<F3>	Начинает поиск введенного текста с текущей позиции и далее
Replace – Найти и заменить			Находит заданный текст и заменяет его на введенный. Вызов команды открывает диалог поиска и замены выбранного текста
Input Assistant – Ассистент ввода			Открывает диалог выбора элемента, который можно ввести в текущей позиции. В левом столбце следует выбрать категорию элементов, в правом – требуемый элемент, а затем нажать OK
Auto Declare – Автоматическое объявление переменных			Открывает диалог объявления переменных
Next Error – Следующая ошибка			Показывает следующую ошибку, если проект скомпилирован с ошибками. При этом открывает соответствующий редактор в том месте, где допущена ошибка, а в окне сообщений отображается краткое описание ошибки
Previous Error – Предыдущая ошибка			Показывает предыдущую ошибку, если проект скомпилирован с ошибками. При этом открывает соответствующий редактор в том месте, где допущена ошибка, а в окне сообщений отображается краткое описание ошибки
Macros – Макросы			Показывает список всех определенных в проекте макрокоманд (макросов). При выполнении макроса открывается окно «Process Macro», в котором выводится имя макроса и имя активной команды
Project – Проект			
Build – Компилировать измененные POU			Компилирует только POU, которые были изменены
Rebuild all – Компилировать весь проект			Компилирует весь проект, даже если он не был изменен
Clear all – Очистить все			Стирает всю информацию о предыдущей компиляции и загрузке проекта в контроллер
Load Download-Information – Загрузить информацию о загрузке кода			Загружает информацию о загрузке кода в контроллер, если она была сохранена в директории, отличной от той, в которой находится проект
Translate into another language – Перевести текст проекта на другой язык			Переводит текст проекта на другой национальный язык (используется вспомогательный текстовый файл, созданный в CODESYS и переведенный в текстовом редакторе на требуемый язык)
Document – Создать документ для печати			Создает версию проекта для печати

Export… – Экспортировать проект			Экспортирует проект из одного инструмента МЭК программирования в другой. Можно экспортировать POU, типы данных, визуализации, описания подключенных к проекту библиотек (но не сами библиотеки) и другие ресурсы
Import – Импортировать данные из файла в проект			Импортирует в проект данные из выбранного файла
Siemens Imports – Импортировать из файлов Siemens			Импортирует переменные и POU из файлов Siemens-STEP5 и STEP7
Merge – Слияние проектов			Сливает два проекта
Compare – Сравнить			Сравнивает два проекта или разные версии одного и того же проекта
Project info – Информация о проекте			Сохраняет дополнительную информацию о проекте
Global Search – Глобальный поиск		<Ctrl+F>	Находит заданный текст в POU, типах данных или разделе глобальных переменных проекта
Global Replace – Заменить текст			Находит заданный текст в POU, типах данных или в глобальных переменных проекта и заменяет его другим
Viev Instance – Показать экземпляры функционального блока			Показывает экземпляры выбранного в организаторе объектов функционального блока. Отображает список всех экземпляров выбранного функционального блока и его реализация
Show Call Tree – Показать дерево вызовов объекта			Показывает дерево вызовов выбранного объекта в новом окне (проект должен быть скомпилирован)
Show Cross Reference – Показать адрес переменной			Открывает диалог, в котором отображены адрес, место расположения (POU, номер строки) переменной (проект должен быть скомпилирован)
Check – Семантический контроль			Команды этого меню используются для дополнительного семантического контроля (проект должен быть скомпилирован без ошибок)
Add Action – Добавить действие			Создает действие, связанное с блоком, выделенным в Организаторе объектов. При этом следует задать имя действия и язык, на котором оно будет описано
User Group Passwords – Пароли групп пользователей			Устанавливает права доступа к объекту для различных групп пользователей
Меню Insert – Вставка			
Declaration keywords – Список ключевых слов			Выводит список ключевых слов для быстрого ввода ключевых слов, допускаемых в разделе объявлений POU. После выбора ключевого слова из списка, оно будет вставлено в текущую позицию курсора
Types – Список типов			Выводит список доступных типов для их быстрого ввода
New declaration – Добавить новую переменную			Добавляет новую переменную в таблицу редактора объявлений
Extras – Опции			Команды данного пункта Меню могут меняться в зависимости от режима работы

Online – Подключение к контроллеру			
Login – Подключиться к контроллеру			Устанавливает соединение ПО CODESYS с контроллером (или запускает программу эмуляции) и включает режим «Online»
Logout – Отключить соединение с контроллером			Разрывает соединение с контроллером или заканчивает работу программы (в режиме эмуляции). Включает режим «Offline»
Download – Загрузить код проекта в контроллер			Загружает код проекта в контроллер
Run – Запустить программу в контроллере/режим эмуляции			Запускает программу на выполнение в контроллере или в режиме эмуляции
Stop – Остановить выполнение программы			Останавливает программу при ее выполнении в контроллере или в режиме эмуляции
Reset – Сброс			Заново инициализирует все переменные, за исключением VAR RETAIN. Если определены начальные значения переменных, они будут присвоены (включая VAR PERSISTENT). Прочие переменные приобретут стандартные значения по умолчанию (например, 0 для целых типов). Данный сброс аналогичен выключению и включению питания ПЛК, при работающей программе
Reset (cold) – Холодный сброс			Холодный сброс. Выполняет те же действия, что и при команде «Reset», и дополнительно выполняет инициализацию энергонезависимой области памяти RETAIN
Reset (original) – Заводской сброс			Выполняет «Reset Cold». Происходит инициализация области PERSISTENT и удаление программы пользователя. Восстанавливаются заводские настройки контроллера
Toggle Breakpoint – Установить точку останова			Устанавливает точку останова в текущей позиции активного окна. Если в этой позиции уже стоит точка останова, то она будет удалена
Breakpoint Dialog – Открыть диалог управления точками останова			Открывает диалог управления точками останова в проекте; в нем указаны все заданные точки останова
Step over – Выполнить одну инструкцию программы			Выполняет одну инструкцию программы. Если это инструкция вызова POU, то данный POU выполняется целиком, затем программа останавливается
Step in – Выполнить программу по шагам			Выполняет программу по шагам, с заходом в вызываемые блоки. Вызываемые POU открываются в отдельных окнах
Single Cycle – Выполнить один цикл			Выполняет один рабочий цикл контроллера. Команду можно повторять многократно при отслеживании работы программы по рабочим циклам
Write values – Записать значение в переменную			Перед началом рабочего цикла присваивает переменной (или нескольким переменным) заранее введенные значения
Force values – Зафиксировать значение переменной			Фиксирует значения одной или нескольких переменных. Запись заданного значения осуществляется в начале и в конце каждого управляющего цикла: 1. Чтение входов, 2. Фиксация переменных, 3. Выполнение кода программы, 4. Фиксация переменных, 5. Запись выходов

Release force – Отменить фиксацию значений переменных			Отменяет фиксацию значений переменных
Write/Force-Dialog – Показать список записываемых и фиксируемых переменных			Открывает окно, содержащее таблицы записываемых (Writelist) и фиксируемых (Forcelist) переменных. В левом столбце таблиц отображаются имена переменных, в правом – их установленные значения
Show Call Stack – Показать список вызванных POU			Показывает список вызванных POU, когда программа остановлена в точке останова
Display Flow control – Показать контроль потока исполнения			Включает режим контроля потока исполнения. Если данная возможность поддерживается целевой платформой, то каждая строка или цепь программы, которая была выполнена в контроллере в предыдущем управляющем цикле, будет выделена
Simulation – Включить режим эмуляции			«Старт». Включает режим эмуляции, программа будет выполнена в ПК. Если режим эмуляции выключен, программа будет запущена в контроллере
			«Стоп». Останавливает программу при ее выполнении в контроллере или в режиме эмуляции
Communication Parameters – Параметры соединения			Выводит диалог настройки параметров связи ПК и ПЛК (если используется OPC или DDE сервер, то эти параметры можно настроить из их конфигурации)
Sourcecode download – Загрузить в контроллер исходные тексты проекта			Загружает исходные тексты проекта в контроллер (именно исходные тексты проекта – не код проекта, который создается при компиляции)
Create bootproject – Автоматически загружать код проекта при перезапуске ПЛК			Делает код проекта автоматически загружаемым при перезапуске контроллера: проект будет выполняться автоматически при перезапуске ПЛК
Write file to PLC – Записать файл в контроллер			Записывает в ПЛК выбранный файл (любого типа); размер файла ограничивается размером карты памяти контроллера
Read file from PLC – Считать файл из контроллера			Считывает ранее сохраненный в контроллере файл и сохраняет его в указанную директорию на ПК
Window – Работа с окнами			
Tile Horizontal – Упорядочить окна по горизонтали			Упорядочивает размещение окон по горизонтали так, чтобы они не перекрывали друг друга и полностью занимали рабочую область
Tile Vertical – Упорядочить окна по вертикали			Упорядочивает размещение окон по вертикали так, чтобы они не перекрывали друг друга и полностью занимали рабочую область
Cascade – Упорядочить окна каскадом			Упорядочивает окна каскадом – каждое следующее поверх остальных
Arrange Symbols – Выстроить свернутые окна			Выстраивает свернутые окна в ряд в нижней части Рабочего окна
Close All – Закрыть все окна			Закрывает все окна

Messages – Открыть окно сообщений			Открывает окно сообщений, которое содержит информацию о предыдущей компиляции, проверке или сравнении проекта
Library Manager – Открыть менеджер библиотек			Открывает окно менеджера библиотек.
Log – Открыть бортжурнал			Открывает «бортжурнал» – детальный протокол последовательности действий, которые были выполнены во время «Online»-сессии. Бортжурнал записывается в двоичный файл формата *.log
Help – Помощь			
Contents – Содержание справочной документации			Открывает окно системы оперативной помощи
Search – Осуществить поиск			Переход к контекстному поиску по текстам оперативной помощи
About – Показать информацию о программе			Открывает окно с информацией о программе CODESYS

A.2 Основные режимы (Редакторы) ПО CODESYS

CODESYS предоставляет встроенные специализированные редакторы для всех пяти языков МЭК 61131-3 и дополнительный CFC редактор:

- Список Инструкций (IL);
- Функциональные блоковые диаграммы (FBD);
- Релейно-контактные схемы (LD);
- Структурированный текст (ST);
- Последовательные функциональные схемы (SFC):

  мониторинг времени исполнения шагов;
  автоматический анализатор причин ошибок;
  набор управляющих флагов: сброс, разрешение мониторинга, фиксация переходов и т.д.

- Непрерывные функциональные диаграммы (CFC):

  автоматическая расстановка и соединение;
  макро опция для структурирования больших диаграмм.

Два специальных редактора управляют прикладной средой исполнения:

- Конфигуратор задач задает:

  циклические задачи и задачи, исполняемые по событиям;
  параметры сторожевого таймера;
  настройку событий.

- Конфигуратор ввода-вывода обеспечивает:

  Profibus конфигурирование на основе GSD файлов;
  CANopen конфигурирование на основе EDS файлов;
  ASI конфигурирование.

Приложение Б. Примеры настройки опроса переменных (ОВЕН)

В Приложении представлены примеры настройки опроса переменных модуля «Owen (Master)» для некоторых наиболее часто встречающихся случаев применения ПЛК (см. таблицы Б.1-Б10). Эти примеры могут быть использованы и при настройке модулей «Owen(Slave)» и «Owen(Spy)».

Примечание.Следует помнить, что при настройке модулей Owen (Spy) и Owen (Slave) не используется параметр «Polling Time» (период опроса). При настройке модуля Owen (Slave) также не используются параметры «Address Type» (тип адреса) и «Address» (адрес).

Б.1 Пример 1

Таблица Б.1 – Описание переменной

Характеристика	Значение
Прибор	Все приборы с аналоговыми входами
Тип переменной	Число с плавающей точкой и циклическим временем
Имя параметра	Read
Описание	Значение с измерителя
Диапазон	В соответствии с диапазоном аналогового входа
Диапазон индекса	0 .. 6

Таблица Б.2 – Настройки модуля Owen (Master)

Характеристика	Значение
Тип переменной	Float variable + time
Направление опроса	Чтение
Address Length	8/11 бит в зависимости от настроек сети
Address	Адрес прибора + номер канала с нуля
Hash name	Read
Index	Индекс содержится в адресе прибора
Use a index?	No
Float type	Float
Precision	Не используется
Polling time	В мс, в зависимости от необходимого периода опроса
Вставленные подмодули	Отсутствуют

Б.2 Пример 2

Таблица Б.3 – Описание переменной

Характеристика	Значение
Прибор	ТРМ133, МВА8, ТРМ151, ТРМ148
Тип переменной	Тип данных с числом с фиксированной точкой
Имя параметра	itrL
Описание	Период опроса датчика
Диапазон	0,1 … 30
Диапазон индекса	0 … 6

Таблица Б.4 – Настройки модуля Owen (Master)

Характеристика	Значение
Тип переменной	Float variable
Направление опроса	Чтение/Запись (в зависимости от потребностей)
Address Length	8/11 бит в зависимости от настроек сети
Address	Адрес прибора
Hash name	itrL
Index	0-6 (в зависимости от номера канала измерения)
Use a index?	Yes
Float type	Fix point binary
Precision	1
Polling time	В мс, в зависимости от необходимого периода опроса
Вставленные подмодули	Отсутствуют

Б.3 Пример 3

Таблица Б.5 – Описание переменной

Характеристика	Значение
Прибор	ТРМ133, МВА8, ТРМ151, ТРМ148
Тип переменной	Тип данных со знаковым/беззнаковым целым значением
Имя параметра	in-t
Описание	Тип датчика
Диапазон	Не ограничен
Диапазон индекса	0 … 6

Таблица Б.6 – Настройки модуля Owen (Master)

Характеристика	Значение
Тип переменной	Unsigned variable
Направление опроса	Чтение/Запись (в зависимости от потребностей)
Address Length	8/11 бит в зависимости от настроек сети
Address	Адрес прибора
Hash name	in-t
Index	0…6 (в зависимости от номера канала измерения)
Use a index?	Yes
Polling time	В мс, в зависимости от необходимого периода опроса
Вставленные подмодули	
1	8 bits

Б.4 Пример 4

Таблица Б.7 – Описание переменной

Характеристика	Значение
Прибор	Любой прибор ОВЕН
Тип переменной	Тип данных с текстовой информацией
Имя параметра	Dev
Описание	Название прибора
Диапазон	Строка
Диапазон индекса	Нет индекса

Таблица Б.8 – Настройки модуля Owen (Master)

Характеристика	Значение
Тип переменной	String variable
Направление опроса	Чтение (константное значение)
Address Length	8/11 бит в зависимости от настроек сети
Address	Адрес прибора
Hash name	dev
Index	Не используется
Use a index?	No
Polling time	В мс, в зависимости от необходимого периода опроса
Вставленные подмодули	Отсутствуют

Б.5 Пример 5

Таблица Б.9 – Описание переменной

Характеристика	Значение
Прибор	ТРМ133
Тип переменной	Тип данных с информацией о времени
Имя параметра	t.val
Описание	Время полного хода 3-х позиционного ИМ, мин:сек
Диапазон	1 … 999
Диапазон индекса	0 … 7

Таблица Б.10 – Настройки модуля Owen (Master)

Характеристика	Значение
Тип переменной	Timevariable
Направление опроса	Чтение/Запись (в зависимости от потребностей)
Address Length	8/11 бит в зависимости от настроек сети
Address	Адрес прибора
Hash name	t.val
Index	0-7 (в зависимости от номера канала)
Use a index?	Yes
Less time field	Seconds
Polling time	В мс, в зависимости от необходимого периода опроса
Вставленные подмодули	
1	Seconds
2	Minutes

Приложение В. Сообщения об ошибках в ПЛК

В Приложении представлены коды и причины возможных ошибок, возникающих в процессе работы модулей, и хранящиеся в переменных этих модулей:

- «Мастер» (ОВЕН, ModBus, DCON);
- «Архиватор»;
- архивации информации в файл.

В.1 Коды ошибок модулей «Мастер»

В.1.1 Модуль ModBus (Мастер)

В модуле **ModBus (Мастер)** используются два канала для отображения статуса Мастера и возникающих ошибок:

- **«Last Address»** – содержит адрес последнего опрошенного мастером устройства (адрес последовательного устройства или IP-адрес, в зависимости от режима работы универсального устройства **ModBus**);
- **«Last error»** – содержит код ошибки из таблицы В.1.

Таблица В.1 – Ошибки работы модуля ModBus (Мастер)

Краткое наименование	Код ошибки		Причины ошибок
	(hex)	(dec)	
OK	0x0000	0	Нет ошибок
NO_DEVICE	0x0051	81	Превышен таймаут ожидания ответа
NO_SOCKET	0x0054	84	Нет свободного сокета для устройства TCP/IP
SOCKET_ERROR	0x0055	85	Ошибка при приеме/передаче по сети TCP/IP

В.1.2 Модуль ОВЕН (Мастер)

В модуле ОВЕН (Мастер) используются три канала для отображения статуса мастера и возникающих ошибок:

- **«Last error»** – содержит код ошибки (см. таблицу В.2)
- **«Last Address»** – содержит адрес последнего опрошенного Мастером устройства
- **«Last HASH»** – содержит Hash-код переменной, которая была опрошена последней.

Таблица В.2 – Ошибки работы модуля ОВЕН (Мастер)

Краткое наименование	Код ошибки		Причины ошибок
	(hex)	(dec)	
OK	0x0000	0	Нет ошибок
NO_DEVICE	0x0051	81	Превышен таймаут ожидания ответа
N_ERR	0x0057	87	В ответе устройства содержится Hash-код ошибки **n.Err (0x0233)**. Уточнение ошибки содержится в старшем байте кода ошибки (см. таблицу В.4)
BAD_HASH	0x4000	16384	Hash-код в ответе не соответствует ожидаемому*
BAD_ADDRESS	0x8000	32768	Адрес в ответе не соответствует ожидаемому*
* Эти коды ошибок могут быть вызваны совместной работой нескольких мастеров в одной сети			

Таблица В.3 – Коды ошибок приборов в сети ОВЕН

Краткое наименование	Код ошибки в старшем байте		Причины ошибок
	(hex)	(dec)	
Определение констант ошибок приема			
OK	0	0	Безошибочный прием кадра
Ошибки записи параметров и атрибутов функцией modific			
PDOT	2	2	Задано положение точки, превышающее 3
EROM	3	3	Попытка модификации ROM-параметра
ESTR	4	4	Не целое число при записи индекса строки или времени
EDOT	5	5	Неверно задано положение точки (при фиксированной точке)
ERNG	6	6	Значение мантиссы превышает ограничения дескриптора
Ошибки записи атрибутов функциями modAllPermis() и modEditPermis()			
EOWNER	7	7	Несанкционированная попытка редактирования Атрибутов (попытка изменить атрибут пользователем, не являющимся хозяином параметра).
EPERM	8	8	У запрошенного параметра отсутствуют признаки
Стандартные ошибки, присущие протоколу обмена			
AFE	0x21	33	Аппаратная ошибка кадрирования
B8E	0x22	34	Ошибка в 8-ом бите посылки
B9E	0x23	35	Ошибка в 9-ом бите посылки
SBE	0x24	36	Ошибка приема стоп-байта (стоп пришел не вовремя)
OVB	0x25	37	Ошибка переполнения буфера
ERS	0x26	38	Принят недопустимый символ
CRCE	0x27	39	Неверная контрольная сумма кадра
EDESC	0x28	40	Не найден дескриптор
NFNC	0x29	41	Не найдена сетевая функция, хотя дескриптор найден. В нормально функционирующем приборе **эта ошибка встречаться НЕ ДОЛЖНА!**
Стандартные ошибки, общие для всех модулей			
EDGT	0x30	48	Мантисса двоично-десятичного параметра содержит ошибку
SZE	0x31	49	Размер поля данных не соответствует ожидаемому
EASK	0x32	50	Значение бита запроса не соответствует ожидаемому
EACC	0x33	51	Редактирование параметра запрещено индивидуальным атрибутом

Таблица В.3 – Продолжение

Краткое наименование	Код ошибки в старшем байте		Причины ошибок
	(hex)	(dec)	
IDXOVF	0x34	52	Недопустимо большой линейный индекс
IDXLIM	0x35	53	Индекс параметра превышает ограничитель индекса
EXTROM	0x36	54	Индекс параметра превышает ограничитель индекса
RESERVED	0x37	55	Данный код не используется
Стандартные ошибки, общие для всех модулей			
EDGT	0x30	48	Мантисса двоично-десятичного параметра содержит ошибку
SZE	0x31	49	Размер поля данных не соответствует ожидаемому
EASK	0x32	50	Значение бита запроса не соответствует ожидаемому
EACC	0x33	51	Редактирование параметра запрещено индивидуальным атрибутом
IDXOVF	0x34	52	Недопустимо большой линейный индекс
IDXLIM	0x35	53	Индекс параметра превышает ограничитель индекса
EXTROM	0x36	54	Индекс параметра превышает ограничитель индекса
RESERVED	0x37	55	Данный код не используется
«Запись запрещена групповым атрибутом уровня»			
LEVGRATT	0x38	56	Запрещающий групповой атрибут находится на уровне 0 (в корне)
LEVGRATT1	0x39	57	Запрещающий групповой атрибут находится на уровне 1
LEVGRATT2	0x3A	58	Запрещающий групповой атрибут находится на уровне 2
LEVGRATT3	0x3B	59	Запрещающий групповой атрибут находится на уровне 3
LEVGRATT4	0x3C	60	Запрещающий групповой атрибут находится на уровне 4
LEVGRATT5	0x3D	61	Запрещающий групповой атрибут находится на уровне 5
LEVGRATT6	0x3E	62	Запрещающий групповой атрибут находится на уровне 6
LEVGRATT7	0x3F	63	Запрещающий групповой атрибут находится на уровне 7

Таблица В.3 – Окончание

Краткое наименование	Код ошибки в старшем байте		Причины ошибок
	(hex)	(dec)	
Состояния COMMON-сегмента			
__LOCKSEG	0x41	65	Выполняется другая задача (Сегмент COMMON занят)
__FREESEG	0x42	66	Задача еще не запущена (Сегмент COMMON свободен)
__READYSEG	0x43	67	Запрошенная задача уже выполняется
__DEBUGSEG	0x44	68	Программе неизвестна запрошенная функция
__NOWHATCOM	0x45	69	В программе стоит заглушка функции WhatCOMState()
__NORUNCOM	0x46	70	В программе стоит заглушка функции RunCOMTask()
	0x47	71	Недопустимое сочетание значений параметров (Изменение параметра было запрещено функцией Valid)
	0x48	72	Ошибка при чтении EEPROM
Ошибки при редактировании графиков			
	0x49	73	Нарушена упорядоченность узлов X по возрастанию
	0x4A	74	Попытка записи X при ненулевом числе узлов графика
	0x4B	75	Ошибка выполнения функции PrevWriteActions()
Ошибки мостов и ретрансляторов			
GATE _OVR	0x50		Переполнение буфера моста или ретранслятора
GATE_DERR	0x51	81	Превышение тайм-аута ответа, потеря пакета в дочерней сети (сети, в которую ретранслируется пакет)
GATE_NONET	0x52	82	Запрошенная дочерняя подсеть не доступна (в случае ретрансляции в одну из нескольких дочерних подсетей)
GATE_MERR	0x53	83	Ответ из дочерней сети не может быть ретранслирован в материнскую сеть.
***Примечание**. Коды ошибок приборов в сети ОВЕН приведены в соответствие с описанием протокола приборов ПО ОВЕН по RS-485*			

В.1.3 Модуль DCON (Мастер)

В модуле **DCON (Мастер)** используется один канал для отображения статуса Мастера и возникающих ошибок: **«Last error»**; содержит код ошибки из таблицы В.4.

Таблица В.4 – Ошибки работы модуля DCON (Мастер)

Краткое наименование	Код ошибки		Причины ошибок
	(hex)	(dec)	
OK	0x0000	0	Нет ошибок
NO_DEVICE	0x0051	81	Превышен таймаут ожидания ответа

Информация о работе каждого отдельного устройства **DCON** выводится в поле «Status» универсального устройства DCON. Коды ошибок приведены в таблице В.5.

Таблица В.5 – Ошибки работы универсального устройства DCON

Краткое наименование	Код ошибки		Причины ошибок
	(hex)	(dec)	
NOT_INITIALIZED	0x0000	0	Модуль универсального устройства DCON не был проинициализирован корректно
REQUEST	0x0001	1	Послан запрос к устройству
BAD_REQUEST_FORMAT	0x0041	65	Неправильный формат строки запроса
BAD_REQUEST_DATA	0x0081	129	Данные для запроса не соответствуют по формату строке запроса
NO_DEVICE	0x0051	81	Превышен таймаут ожидания ответа
UNINDENTIFIED_ANSVER	0x0021	33	Ответ не распознан
OK_ANSVER	0x0003	3	Пришел ответ, соответствующий строке формата для случая положительного ответа
OK_ANSVER_BAD_FORMAT	0x0043	67	Неправильный формат строки разбора положительного ответа
OK_ANSVER_BAD_DATA	0x0083	131	Данные для разбора положительного ответа не соответствуют по формату строке разбора
NEG_ANSVER	0x0023	35	Пришел ответ, соответствующий строке формата для негативного ответа
NEG_ANSVER_BAD_FORMAT	0x0063	99	Неправильный формат строки разбора негативного ответа
NEG_ANSVER_BAD_DATA	0x00A3	163	Данные для разбора негативного ответа не соответствуют по формату строке разбора

Примечание. *Переменная* ***Status*** *предназначена также для управления работой мастера DCON при настройке универсального устройства DCON в режиме* ***«Опрос по команде» (Work mode=«By command»).*** *Для однократного запуска опроса необходимо записать в переменную значение* ***0xFF****.*

В.2 Коды ошибок подмодуля «Modem»

Вподмодуле**«Modem»** используетсяодинканалдля отображениявозникающихошибок: **«Modem fault»**; содержит код ошибки из таблицы В.6.

Таблица В.6 – Ошибки работы подмодуля «Modem»

Краткое наименование	Код ошибки		Причины ошибок
	(hex)	(dec)	
OK	0x0000	0	Нет ошибок, модем исправен
MODEM_FAULT	0x0001	1	Подмодуль «**Modem**» зафиксировал отказ подключенного модема или его отсутствие

В.3 Коды ошибок модуля «Архиватор»

В модуле **«Архиватор»** используется один канал для отображения статуса и возникающих ошибок: **«Status»**; содержит код ошибки из таблицы В.7.

Таблица В.7 – Ошибки работы модуля «Архиватор»

Краткое наименование	Код ошибки		Причины ошибок
	(hex)	(dec)	
STANDBY	0x0000	0	Модуль «**Архиватор**» остановлен
RUN	0x0001	1	Модуль «**Архиватор**» запущен и работает без ошибок
DEVICE_ERROR	0x0002	2	Ошибка при записи в устройство
NO_DEVICE	0x0004	4	Устройство вывода отсутствует или неправильно проинициализировано

Примечание. При записи команды в переменную «**Status**» архивирование может быть запущено или остановлено. Коды команд даны в таблице В.8.

Таблица В.8 – Коды команд управления модулем «Архиватор»

Краткое наименование	Код команды		Описание команды
	(hex)	(dec)	
ARCHIVING_STARTED	0x00FF	255	Запуск архивирования
ARCHIVING_STOPED	0x00FE	254	Остановка архивирования

В.4 Коды ошибок модуля архивирования информации в файл

Для подмодуля архивирования информации в файл дополнительно в поле **«File Status»** выводится информация о работе подмодуля. Коды ошибок приведены в таблице В.9.

Таблица В.9 – Ошибки работы модуля архивирования информации в файл

Краткое наименование	Код ошибки		Причины ошибок
	(hex)	(dec)	
OK	0x0000	0	Модуль функционирует нормально
STOPED	0x0001	1	Вывод в файл запрещен
CANT_OPEN	0x0002	2	Невозможно открыть файл
TOO_LARGE	0x0003	3	Размер записи превышает размер файла. Увеличить файл или разбить данные на несколько файлов
FILE_FULL	0x0004	4	Файл заполнен. Перезапись или сдвиг запрещены
GENERAL_ERROR	0x0005	5	Ошибка инициализации

Приложение Г. Примеры настройки опроса (DCON, Master)

В Приложении представлены примеры настройки модуля **DCON (Master)** для опроса устройств ввода/вывода.

Г.1 Опрос модулей аналоговых входов IPC-7033

Производится опрос первых трех входов нескольких модулей аналоговых входов IPC-7033.

Описание формата обмена с модулем IPC-7033:

Формат запроса:

#AA[CRC][CR]

где:	#	–	разделитель;
	AA	–	адрес прибора;
	CRC	–	контрольная сумма;
	CR	–	перевод строки.

Формат ответа:

>+0255.12+013.45+150.11[CRC][CR]

где:	**>**	–	разделитель в случае положительного ответа;
	+025.12	–	данные одного канала (5 чисел + знак + точка), итого семь символов;
	CRC	–	контрольная сумма;
	CR	–	перевод строки.

Окно конфигурирования модуля **DCON (Master)** с подключенным к нему устройством **Universal DCON device**, настроенным для последовательного опроса нескольких модулей IPC-7033 с различными адресами, проиллюстрировано на рисунке Г.1.

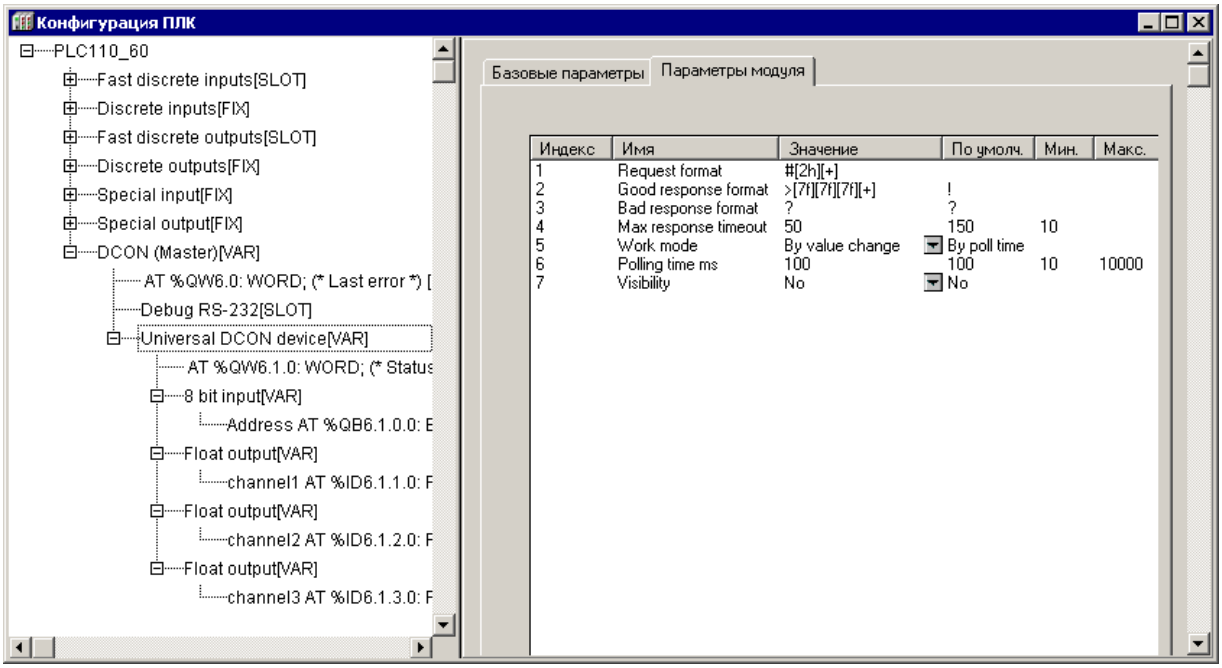


Рисунок Г.1 – Параметры подмодуля «UniversalDCONdevice» модуля DCON (Master) для опроса входов модулей IPC-7033

Для задания адреса опрашиваемого прибора используется 8-битовая входная переменная **«Address»**. Значения со входов опрашиваемого модуля IPC-7033 отображаются в трех выходных переменных типа **Float (REAL).**

Модуль **Universal DCON device** настраивается следующим образом:

Request format – формат строки запроса – **#[2h][+],**

где: **#** – символ разделителя команды опроса входов;
[2h] – спецкоманда, указывающая, что в это место запроса подставляется шестнадцатиричный двухсимвольный адрес, значение которого должно быть взято из входной переменной;
[+] – спецкоманда подсчета и добавления в конец запроса контрольной суммы «по модулю 256».

Внимание! Символ возврата каретки вставляется автоматически!

Good response format – формат положительного ответа – **>[7f][7f][7f][+]**,

где: **>** – символ разделителя в случае положительного ответа;
[7f] – спецкоманда, указывающая на то, что семь символов ответа должны быть преобразованы в число с плавающей точкой и результат преобразования должен быть помещен в первую выходную переменную, которая имеет формат **Float.**

Внимание! Для следующих спецкоманд **[7f]** применяется то же правило преобразования, но результаты помещаются во вторую и третью выходные переменные, соответственно.

[+] – спецкоманда, указывающая на то, что должно быть проанализировано правильность контрольной суммы в принятой посылке. Результат записывается в переменную «**Status».**
CR – перевод строки.

Bad response format – формат отрицательного ответа – **?,**

где: **?** – начальный символ строки отрицательного ответа. В рассматриваемом случае отрицательный ответ не содержит значащей информации, для его идентификации достаточно одного первого символа.

Max response timeout – максимальное время ожидания ответа – 50 мс. Задается в соответствии с рекомендациями производителя прибора.

Work mode – режим работы – **by change value** (по изменению значения одной из входных переменных). Этот режим позволяет генерировать запросы при изменении адреса опрашиваемого прибора. Для генерации одного запроса необходимо записать значение, отличающееся от текущего, во входную переменную **«Adress»**. После этого по значению переменной «**Status»** определяется окончание обмена данными с опрашиваемым прибором и корректность данных в выходных переменных.

Внимание! *Чтение переменной «**Status»**а и ее анализ должны производиться на следующем цикле работы ПЛК после записи нового адреса. Остальные параметры, в данном режиме работы не существенны.*

Г.2 Установка выходного значения модуля IPC-7021

Установка выходного значения модуля аналогового вывода IPC-7021 с периодичностью 1 сек и при необходимости изменения значения следующим образом: производится запись одного выходного значения у модуля IPC-7021 с шестнадцатиричным адресом 18.

Описание формата обмена с модулем IPC-7021:

Формат запроса:

#AA(данные)[CRC][CR]

где: # – разделитель;
AA – адрес прибора;
+025.12 – выходное значение (5 чисел + знак + точка), итого семь символов
CRC – контрольная сумма;
CR – перевод строки.

Формат ответа:

![CRC][CR]

где: **!** – разделитель в случае положительного ответа;
CRC – контрольная сумма;
CR – перевод строки.

Параметры модуля DCON (Master) с подключенным к нему модулем Universal DCON device, настроенным для периодической записи выходных значений в модуль IPC-7021, отображены на рисунке Г.2.

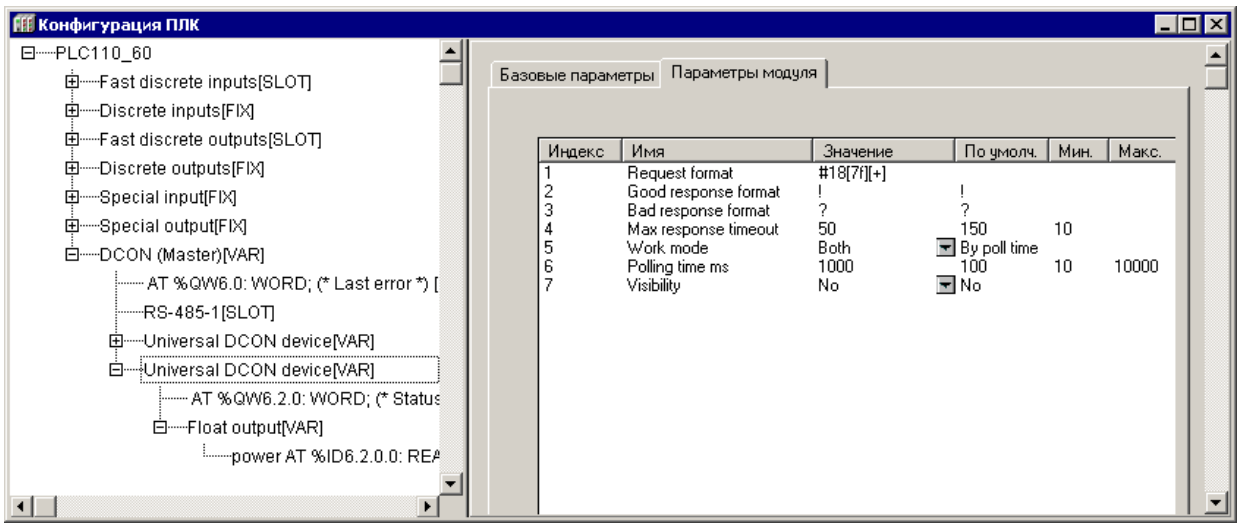


Рисунок Г.2 – Параметры подмодуля Universal DCON device модуля DCON (Master) для периодической записи выходных значений в модуль IPC-7021

Данные, посылаемые в модуль IPC 7021, задаются во входной переменной **«power»** типа **Float (REAL)** модуля **Universal DCON device**.

Модуль **Universal DCON device** настраивается следующим образом:

Request format – формат строки запроса – **#18[7f][+],**

где: **#** – символ разделителя команды опроса входов;
18 – адрес прибора в шестнадцатиричном формате (**для букв используется верхний регистр!);**
[7f] – спецкоманда, указывающая на то, что семь символов запроса должны быть сформированы в виде числа в формате **[знак]число..число.число..число**. Данные должны быть взяты из входной переменной, которая должна иметь формат **float;**
[+] – спецкоманда подсчета и добавления в конец запроса контрольной суммы «по модулю 256».

Внимание! *Символ возврата каретки вставляется автоматически!*

Good response format – формат положительного ответа – **!,**

где: **!** – начальный символ строки положительного ответа. В рассматриваемом случае положительный ответ не содержит значащей информации, для его идентификации достаточно одного первого символа.

Bad response format – формат отрицательного ответа – **?,**

где: **?** – начальный символ строки отрицательного ответа. В рассматриваемом случае отрицательный ответ не содержит информации, для его идентификации достаточно одного первого символа.

Max response timeout – максимальное время ожидания ответа – 50 мс. Задается в соответствии с рекомендациями производителя прибора.

Work mode – режим работы – Both (по времени опроса и смене значения одной из входных переменных). Этот режим позволяет генерировать запросы по таймеру (параметр Polling time) и при изменении значения входной переменной модуля.

Polling time – время опроса – 1000 мс.

Задает период записи значения в модуль IPC-7021.

Приложение Е. Использование OPC-сервера

Приложение предназначено для ознакомления пользователя с технологией подключения ПЛК к ПК через OPC-серверы – разработанные компанией 3S-Software и технологией подключения приборов, разработанных компанией ОВЕН, к ПК через OPC-драйверы.

Для работы оборудования с современными SCADA системами, поддерживающими спецификацию OPCDA, требуются OPC-драйверы (OPC–сервер) для приборов, реализующие спецификацию Data Access (DA).

Прочесть и записать данные может пользовательская программа на языке, полноценно поддерживающем COM технологию Microsoft (Visual Basic, C++, Java, Delphi и т.д.). Получение данных возможно также из приложений, поддерживающих доступ к COM объектам. Например, приложений пакета MS Office, что позволяет пользователю получить, например, в таблице Excel набор технологических параметров, изменяющихся в реальном масштабе времени.

Е.1 Использование OPC-сервера 3S-Software

OPC-сервер, разработанный компанией 3S-Software, предназначен для подключения ПЛК к системам SCADA. OPC-сервер соответствует спецификации OPC DA 2.0, в частности, просмотр списка имен переменных подключенного ПЛК.

Для подключения ПЛК к ПК следует:

1) Загрузить проект в ПО CODESYS и проверить, не подключен ли ПЛК к ПК. Если подключен, то ПЛК следует отключить выбором команды «Онлайн | Отключение (Online | Logout)» главного меню.
2) Выбором пункта «Настройки целевой платформы (Target Settings)» на вкладке «Ресурсы (Resources)» Организатора объектов CODESYS перейти в режим Настройки целевой платформы (Target Settings)».
3) В открывшемся окне режима Настройки целевой платформы (Target Settings)»(см. рисунок Е.1), на вкладке «Общие (General)», установить флажок переключателя «Загружать символьный файл (Download Symbol File)»и нажать кнопку «ОК» окна режима.

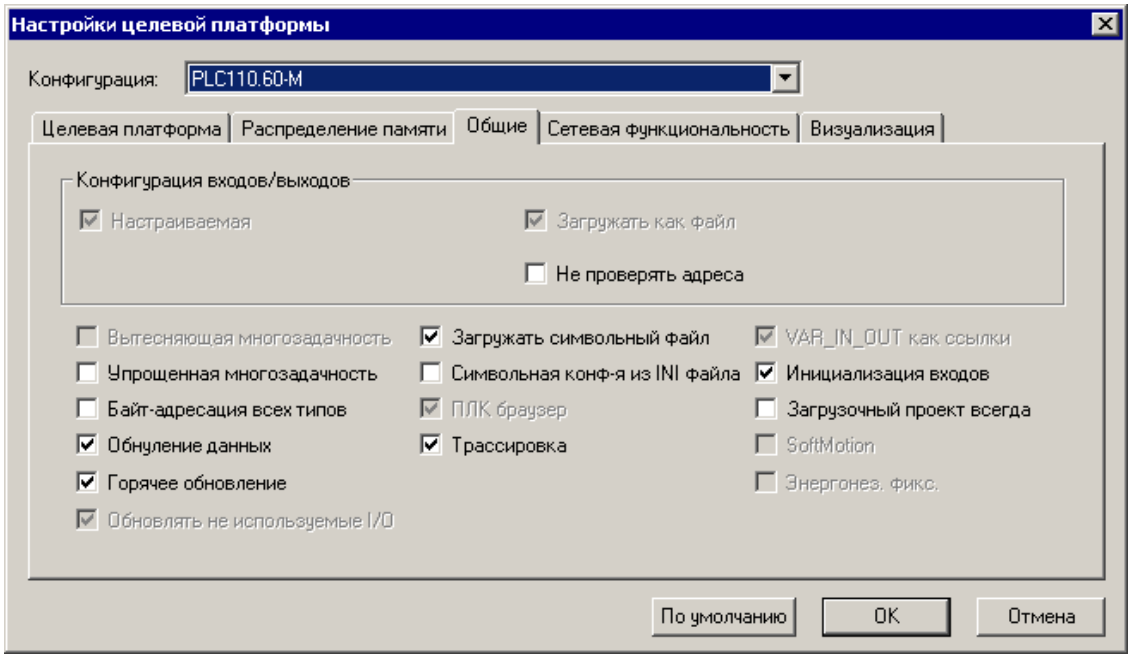


Рисунок Е.1 – Окно режима «Target Settings». Вкладка «General»

4) Выбрать команду «Проект | Опции (Project | Options)» главного меню ПО CODESYS.

5) В открывшемся окне режима «Опции (Options)» (рисунок Е.2), в списке «Категория (Category)», выделить строку «Символьная конфигурация (Symbol Configuration)».
В правой части окна – установить флажок в поле переключателя «Создавать описания (Dump symbol entries)» и нажать кнопку «Настроить символьный файл (Configure symbol file)».

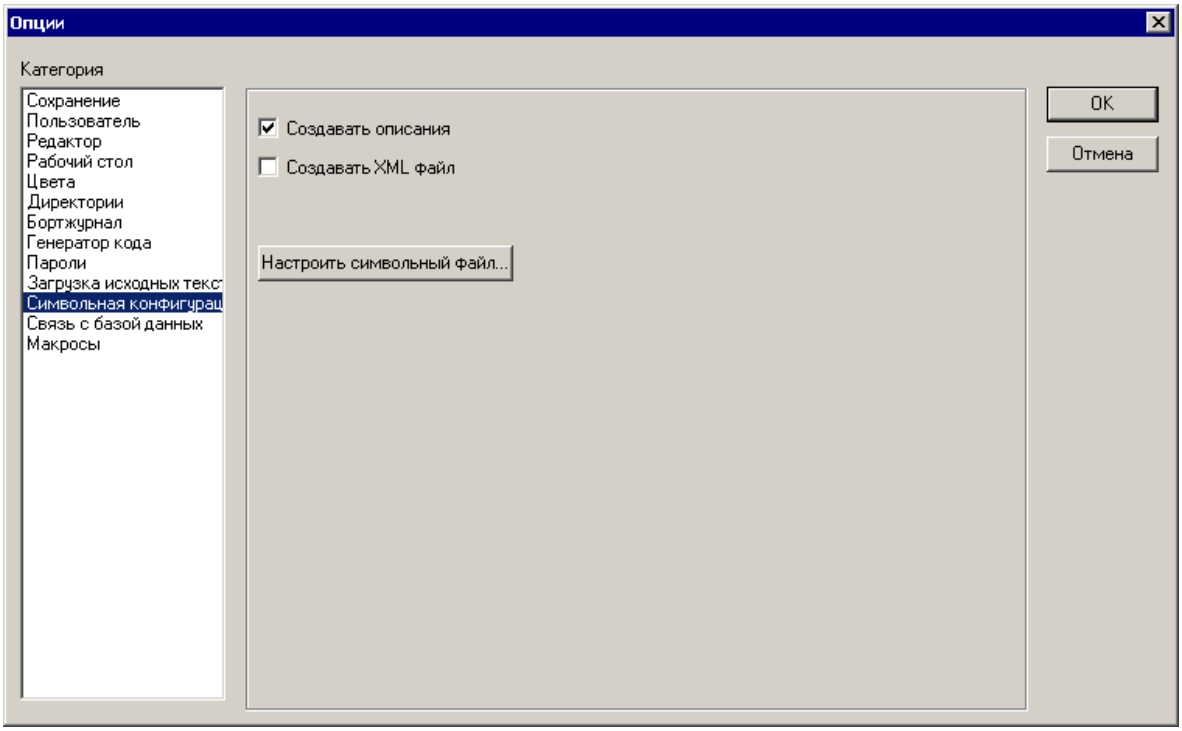


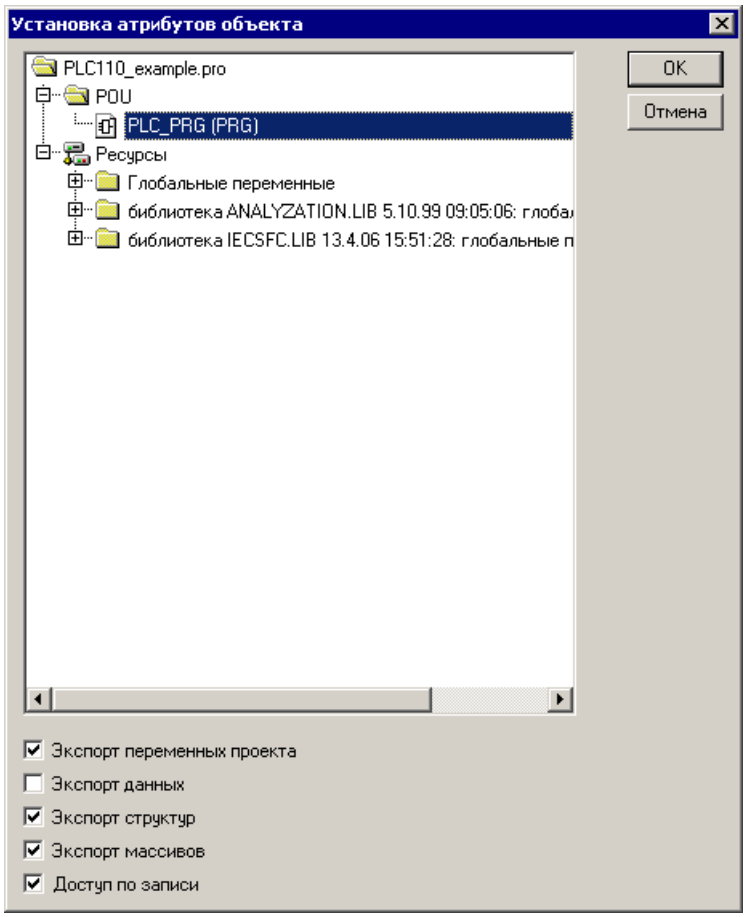
Рисунок Е.2 – Окно режима «Опции (Options)»

6) В открывшемся окне «Установка атрибутов объекта (Set object attributes)» (рисунок Е.3) отображается список параметров переменных проекта.
В этом списке – последовательно выбирая те объекты проекта, из которых требуется экспортировать переменные – установить для каждого из них флажки в полях переключателей в нижней части окна. Для обеспечения экспорта переменных в пространство имен OPC-сервера следует установить флажок в поле переключателя «Экспорт переменных проекта (Export variables of object)». Если требуется изменять значения переменных, то следует установить флажок в поле переключателя «Доступ по записи (Write access)».

Внимание! *При выборе экспортируемых переменных рекомендуется выбирать только требуемые переменные, а не включать в список экспортируемых все переменные проекта.*
Нажать кнопку «ОК» окна «Установка атрибутов объекта (Set object attributes)» и кнопку «ОК» окна «Опции (Options)».

Примечание. *Выполнение процедуры изменения (при необходимости) списка экспортируемых переменных описано в разделе Е.2.*

7) Выбором команды «Файл | Сохранить (File | Save)» главного меню ПО CODESYS или нажатием кнопки «Сохранить (Save)» ( ) панели инструментов – сохранить проект.
8) Выбором команды «Проект | Компилировать все (Project | Rebuild all…)» главного меню ПО CODESYS перекомпилировать проект.
9) Выбором команды «Online | Login» главного меню ПО CODESYS – загрузить проект в ПЛК.



РисунокЕ.3 – Окно «Set object attributes». Выборпараметровпеременныхпроекта

10) Выбором команды «Пуск | Программы | 3S Software | Communication | CODESYS OPC Configurator» – запуститьПО «OPC Configurator».

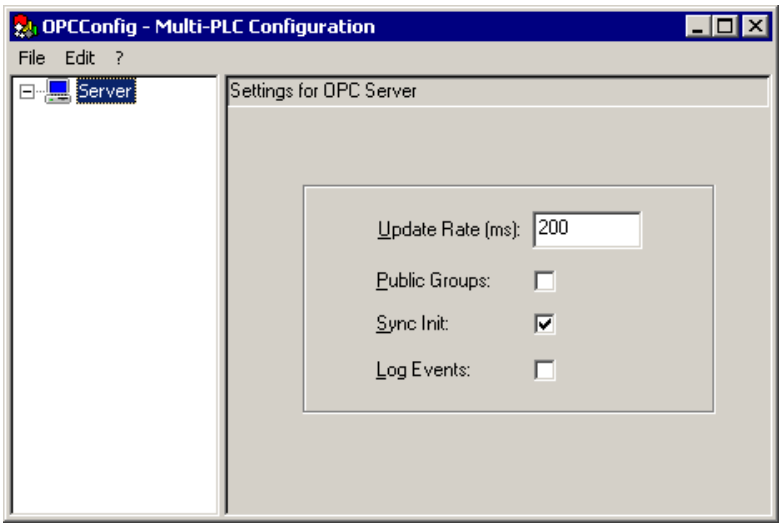
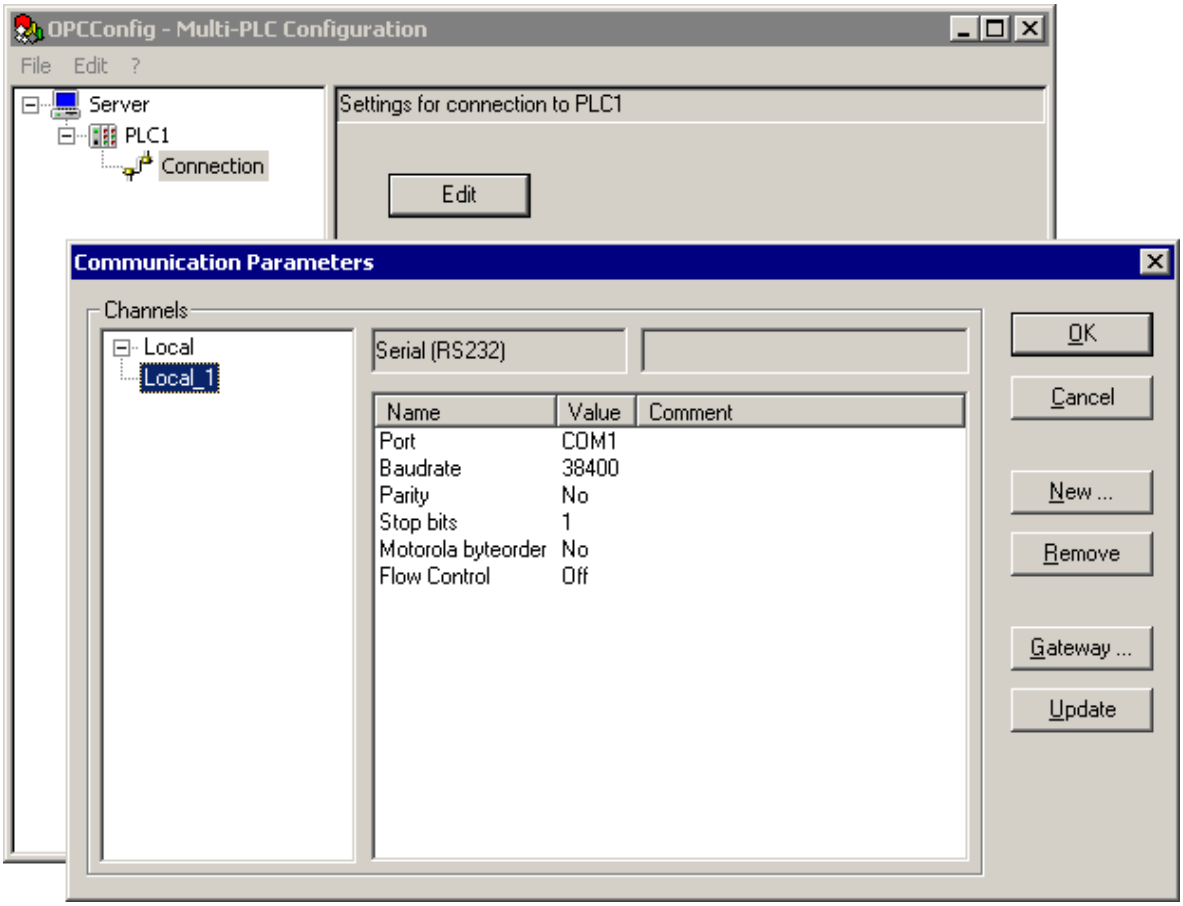


Рисунок Е.4 – Окно OPCConfigurator

11) В открывшемся окне «OPC Config – Multi-PLC Configuration» (рисунок Е.4) в левой части окна отображается иерархический список, исходно содержащий одну строку: «Server».
Выделить строку «Server» и в поле параметров (в правой части окна) – установить время обновления данных (в миллисекундах), введя требуемое значение в поле «Update Rate (ms)».
12) В контекстном меню строки «Server» выбрать команду «Append PLC».
В иерархический список в левой части окна добавляются строки «PLC1» и «Connection».
13) В иерархическом списке (в левой части окна) – выбрать пункт «Connection».
В поле параметров (в правой части окна) – нажать кнопку «Edit»
14) В открывшемся окне «Communication Parameters» – установить параметры подключения ПЛК, см. рисунок Е.5.



РисунокЕ.5 – Окно «Communication Parameters»

15) Нажать кнопку «**ОК»** окна «**Communication Parameters**», в открывшемся окне запроса подтверждения операции («Save changes in Multi-PLC Configuration») – нажать кнопку «Да» для подтверждения произведенных настроек.

После этого ПО «OPC-сервер» сконфигурировано и готово к работе под управлением SCADA-системы.

Е.2 Изменение списка экспортируемых переменных

Список экспортируемых переменных сохраняется в памяти прибора, и для изменения этого списка следует предварительно удалить текущий список из памяти прибора.

Для выполнения удаления текущего списка экспортируемых переменных из памяти прибора следует:

1) Загрузить проект в ПО CODESYS и проверить, не подключен ли ПЛК к ПК. Если подключен, то ПЛК следует отключить выбором команды «Онлайн | Отключение (Online | Logout)» главного меню.
4) Выбрать команду «Проект | Опции (Project | Options)» главного меню ПО CODESYS.
5) В открывшемся окне режима «Опции (Options)» (рисунок Е.2), в списке «Категория (Category)», выделить строку «Символьная конфигурация (Symbol Configuration)».
   В правой части окна – установить флажок в поле переключателя «Создавать описания (Dump symbol entries)» и нажать кнопку «Настроить

символьный файл (Configure symbol file)».

6) В открывшемся окне «Установка атрибутов объекта (Set object attributes)» (рисунок Е.3) – снять флажок переключателя «Экспорт переменных проекта (Export variables of object)». При этом остальные переключатели становятся недоступными для редактирования.
Нажать кнопку «ОК» окна «Установка атрибутов объекта (Set object attributes)» и кнопку «ОК» окна «Опции (Options)».

7) Выбором команды «Проект | Очистить все (Project | Clean all)» главного меню ПО CODESYS очистить сохраненные параметры проекта.

8) Выбором команды «Проект | Компилировать все (Project | Rebuild all…)» главного меню ПО CODESYS перекомпилировать проект.

9) Выбором команды «Файл | Сохранить (File | Save)» главного меню ПО CODESYS или нажатием кнопки «Сохранить (Save)» () панели инструментов – сохранить проект.

После этого память прибора очищена от списка экспортируемых переменных. Процедура создания нового списка переменных (конфигурирования ПО «OPC-сервер») описана в разделе Е.1.

Е.3 Использование OPC-драйверов «ОВЕН»

OPC-драйверы, разработанные компанией ОВЕН, предназначены для подключения приборов фирмы ОВЕН к системам SCADA.

Драйверы реализованы в виде двух модулей: OWEN-RS232 и OWEN-RS485. Они применяются для приборов фирмы ОВЕН, поддерживающих сетевой интерфейс «токовая петля» (для преобразования в сеть RS-232 используется адаптер АС2) и поддерживающих сетевой интерфейс RS-485.

Для преобразования в сеть RS232 или USB можно использовать адаптеры фирмы ОВЕН – АС3, АС3-М, АС4,– или других производителей.

При работе могут быть использованы протоколы OWEN, ModBus-RTU или ModBus-ASCII.

Перед началом работы пользователь должен задать конфигурацию своих приборов и режим работы порта.

К адаптеру АС-2 можно подключить до 8 приборов. К одной сети RS485 подключается до тридцати двух приборов (шлейфом, без применения репитера).

Список приборов, которые можно подключить к серверам:

1) OWEN-RS232:

Задатчик-регулятор МПР51
Измеритель ТРМ0 PiC
Измеритель УКТ38-В
Измеритель УКТ38-Щ4
Измеритель регулятор ТРМ1 PiC
Измеритель регулятор ТРМ10 PiC
Измеритель регулятор ТРМ12 PiC
Измеритель регулятор ТРМ5 PiC
Многоканальный регулятор ТРМ32
Многоканальный регулятор ТРМ33
Многоканальный регулятор ТРМ34
Многоканальный регулятор ТРМ38

2) OWEN-RS485:

Многоканальный регулятор ТРМ138
Универсальный двухканальный программный ПИД-регулятор ТРМ151

Счетчик импульсов СИ8
Прибор контроля положения ПКП1
Модуль ввода аналоговый МВА8
Модуль вывода управляющий МВУ8
ПИД регулятор с универсальным входом ТРМ101
Измеритель двухканальный с универсальными входами ТРМ200
Измеритель-регулятор одноканальный с универсальным входом ТРМ201
Измеритель-регулятор двухканальный с универсальными входами ТРМ202
Контроллер приточной вентиляции ТРМ133

Для установки модулей OWEN-RS232 и OWEN-RS485 требуется запустить программу-инсталлятор (файл OwenOPC-setup.exe), содержащуюся на дистрибутивном диске.

Примечание. *Начиная с версии 1.0.0.5, OPC-сервера OWEN-RS232 добавлен тег, управляющий обменом на внешней шине (флаг активности OPC-сервера). Имя тега «Status/active», тип BOOL. Запись в этот тег 1 (единицы) разрешает обмен по внешней шине, запись 0 (нуля) запрещает обмен.*

Е.3.1 Установка OPC-драйверов фирмы ОВЕН

Для установки модулей OWEN-RS232 и OWEN-RS485 требуется:

1) Запустить программу-инсталлятор (файл OwenOPC-setup.exe).
2) В открывшемся окне программы-инсталлятора – нажать кнопку «Далее». В последовательно открывающихся окнах мастера установки – выполнять инструкции, отображаемые в окне.

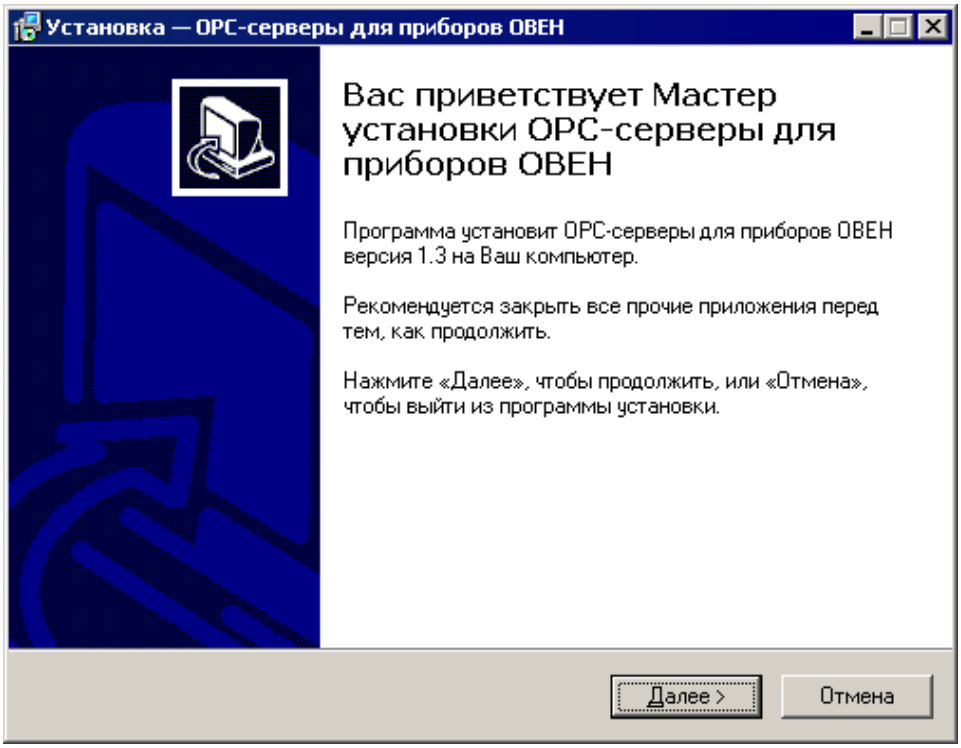


Рисунок Е.6 – Первое окно программы – инсталлятора OPC-сервера для приборов ОВЕН

Приложение Ж. Режим «ПЛК-Браузер» ПО CODESYS

Приложение предназначено для ознакомления пользователя с технологией настройки и мониторинга функционирования программируемого логического контроллера ПЛК, реализованной в режиме «PLC-Browser (ПЛК-Браузер)» ПО CODESYS.

Режим «PLC-Browser (ПЛК-Браузер)» предоставляет пользователю возможность:

- вводить команды в виде текстовых строк;
- передавать команды в ПЛК;
- получать в качестве реакции ПЛК запрошенную информацию или отчет о результатах выполнения команд.

Таким образом, режим предназначен для мониторинга (диагностики) состояния ПЛК и настройки его функционирования.

Ж.1 Вход в режим

Работа в режиме «ПЛК-Браузер (PLC-Browser)» возможна только после физического подключения ПЛК к компьютеру и установки связи с контроллером (связь устанавливается выбором команды «Онлайн | Подключение (Online | Login)» главного меню).

Для входа в режим «ПЛК-Браузер (PLC-Browser)»следует перейти на вкладку «Ресурсы (Resources)» Организатора объектов ПО CODESYS и выбрать строку «ПЛК-Браузер (PLC-Browser)»организатора объектов.

Окно режима представлено на рисунке Ж.1.

Рабочее поле окна режима (расположенное справа от Организатора объектов) разделено на две части: в верхней части окна отображается строка вводимых пользователем команд, в нижней части – поле отображения реакции ПЛК на введенную команду.

Кнопкой  в правой части строки команд вызывается раскрывающийся список, содержащий список (стек) всех ранее введенных со времени запуска проекта команд, автоматически дополняемый впервые вводимыми в рамках проекта командами, список сохраняется до закрытия проекта.

Кнопкой с тремя точками, расположенной справа от строки команд, вызывается перечень доступных команд, в котором можно выбрать требуемую команду и нажатием клавиши <Enter> перенести ее в строку команд.

Введенная пользователем команда передается в контроллер нажатием клавиши <Enter>. В нижней части рабочего поля отображается реакция ПЛК на введенную команду.

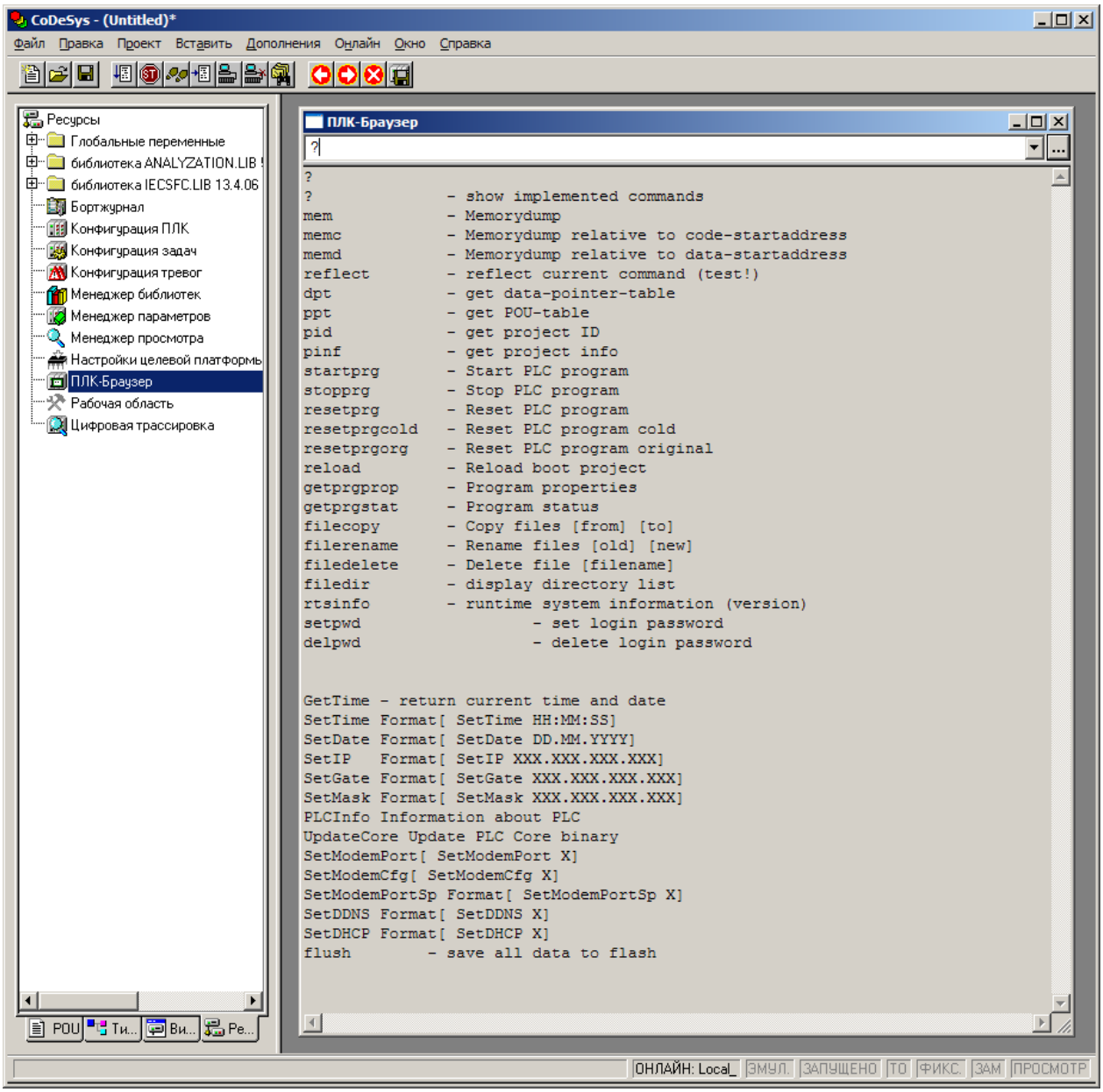


Рисунок Ж.1 – Окно режима «ПЛК-Браузер (PLC-Browser)»

Ж.2 Команды PLC-Browser

Команды режима PLC-Browser обеспечивают выполнение функций манипулирования памятью, файлами, управления программами и информационных функций системы исполнения.

Синтаксис команд: **<команда><пробел><параметры>**

Список параметров определяется типом команды. Переданная команда повторяется в окне отображения вместе с ответом контроллера.

При открытии проекта, список доступных команд режимаможно получить, введя команду «?» (знак вопроса), см. рисунок Ж.1.

Команды режима, применяемые в ПЛК, представлены в таблицах Ж.1 и Ж.2.

Примечание. *Команды перечня стандартных команд PLC-Browser «saveretain» (запись сохраняемых (retain) переменных) и «restoreretain» (чтение сохраняемых (retain) переменных) в ПЛК не используются.*

Таблица Ж.1 – Перечень команд PLC-Browser, применяемых в ПЛК

Команда	Содержание	Описание
?	show implemented commands	Запрос у системы исполнения актуального списка всех поддерживаемых команд
mem	Memory dump	Hex-дампобластипамяти. Синтаксис 1: mem <start address><end address> Синтаксис 2: mem <start address>-<end address> Адрес вводится в виде десятичного или шестнадцатеричного числа (префикс 16#)
memc	Memory dump relative to code-startaddress	Относительный Hex-дамп области кода (аналогична команде mem, адрес задается от начала области кода)
memd	Memory dump relative to data- startaddress	Относительный Hex-дамп области данных (аналогична команде mem, адрес задается от начала области данных)
reflect	reflect current command (test!)	Возврат строки (для тестирования соединения)
dpt	get data-pointer-table	Чтение таблицы указателей данных
ppt	get POU-table	Чтение таблицы POU
pid	get project ID	Чтение идентификатора проекта
pint	get project info	Чтение информации о проекте
startprg	Start PLC program	Запуск программы ПЛК
stopprg	Stop PLC program	Остановка программы ПЛК
resetprg	Reset PLC program	Сброс программы ПЛК – инициализируются только не энергонезависимые переменные
resetprgcold	Reset PLC program cold	Холодный сброс программы ПЛК инициализируется все, в том числе, и энергонезависимые переменные
resetprgorg	Reset PLC program original	Заводской сброс программы ПЛК – полная очистка областей кода и данных
reload	Reload boot project	Перезапись загрузочного кода проекта
getprgprop	Program properties	Свойства программы
getprgstat	Program status	Статус программы
filecopy	Copy files [from] [to]	Копирование файла [из] [в]
filerename	Rename files [old] [new]	Переименование файла [старое имя] [новое имя]
filedelete	Delete file [filename]	Удаление файла [имя файла]
filedir	display directory list	Файловая команда dir (дает лист перечня файлов)
setpwd	set login password	Установка пароля на контроллер. Синтаксис: setpwd <password> [level], где level может быть «0» (по умолчанию), действительный для подключения системы программирования, или «1», действительный для всех приложений
delpwd	delete login password	Удалить пароль

Таблица Ж.2 – Перечень команд разработчика ПЛК

Команда	Содержание	Описание
GetTime	return current time and date	Возврат текущего времени и даты
SetTime	Format [SetTime HH:MM:SS]	Установка времени в формате: часы, минуты, секунды
Set Date	Format [SetDate DD.MM.YYYY]	Установка даты в формате: день, месяц, год
SetIP	Format [SetIP XXX .XXX .XXX .XXX]	Установка IP-адреса в сети Ethernet
SetGate	Format f SetGate XXX .XXX .XXX .XXX]	Установка адреса шлюза в сети Ethernet
SetMask	Format [SetMaskXXX .XXX .XXX .XXX]	Установка маски в сети Ethernet
SetModemCfg	Format [SetModemCfg X]	Настройка конфигурации подключенного модема, гдеX может быть: – «0» (по умолчанию) – модем не подключен или будет использоваться пользовательским программным обеспечением через PLC-Configuration; – «1» – к порту (порт задаётся командой SetModemPort) подключен последовательный модем (используется в режиме прямого соединения с другим модемом); – «2» – к порту подключен последовательный модем (используется в режиме соединения с сетью Интернет GPRS - подключение).
PLC Info	Information about PLC	Информация о типе ПЛК и его настройках
UpdateCore	Обновление прошивки	Команда на перепрошивку контроллера

Примеры ввода команд и реакции контроллера представлены на рисунке Ж.2.

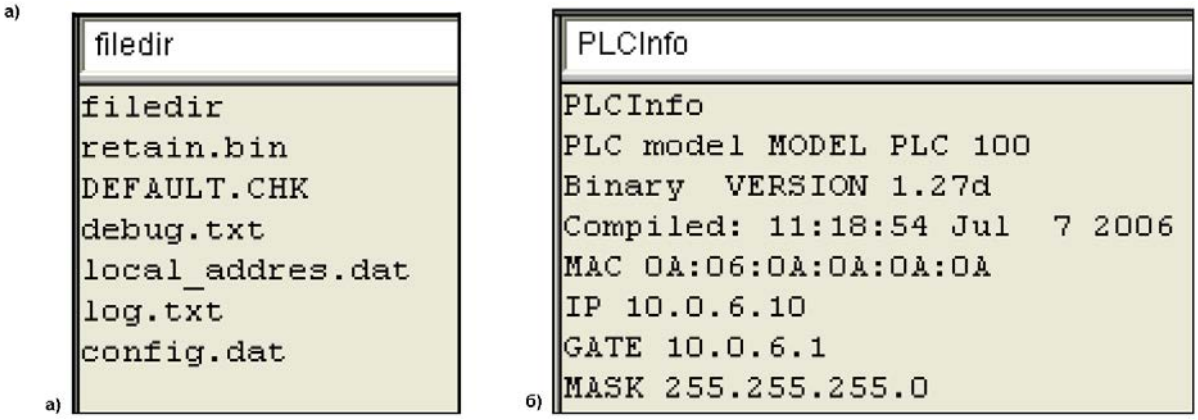


Рисунок Ж.2 – Ввод команд (а) и реакция контроллера (б)

Если команда не распознана контроллером (введена с ошибкой), то в области отображения результата появится сообщение: «Keyword not found» – ключевое слово не обнаружено.

Ж.3 Вспомогательные команды режима PLC-Browser

В режиме PLC-Browser в пункте «Extras» главного меню ПО CODESYS отображаются вспомогательные команды режима, а в панели инструментов – вспомогательные кнопки:

Команды (и кнопки) просмотра истории «Вперед (History forward)» и «Назад (History backward)» дают возможность «прокрутить» результаты выполненных команд вперед и назад по списку. Запись истории сохраняется до закрытия проекта.

Команда (и кнопка) «Cancel» прерывает начатый запрос.

Команда (и кнопка) «Save history list» сохраняет результаты выполненных команд в файле с расширением *.bhl. (Browser History List).

Команда Print last command открывает стандартный диалог печати. При ее инициировании на печать будет выведен текущий запрос и его результат.

Ж.4 Настройка ПЛК: изменение сетевых настроек

Использование нескольких контроллеров в одной Ethernet-сети требует, чтобы их IP-адреса были уникальными. В режиме PLC-Browser можно, при работающем программном соединении, узнать имеющиеся сетевые настройки ПЛК и внести в них требуемые изменения.

Для этой цели физическое и программное соединение со всеми контроллерами фирмы ОВЕН удобнее устанавливать через COM-порт компьютера.

Порядок действий следующий:

1) **Физическое соединение устройств.** Интерфейсным кабелем из комплекта поставки контроллера связывается COM-порт компьютера с гнездом (RS232), расположенным на лицевой панели контроллера (при этом должно быть включено питание);
2) **Выбор вида программного соединения.** В ПО CODESYS запускается созданный проект программы, в главном меню ПО выбирается команда **Online | Communication parameters**, в открывшемся окне нажимается кнопка «New…», в в открывшемся окне новому соединению присваива-

ется имя (например, «Owen_RS232») и выбирается из перечня вид соединения: «Serial (RS232)», в параметре «Baudrate» устанавливается скорость 115200 бит/с.

3) **Включение соединения с контроллером.** В главном меню ПО CODESYS выбирается команда **Online | Login**, подтверждается загрузка программы в контроллер.
4) **Вход режима «PLC-Browser».** В ПО CODESYS, на вкладке «Ресурсы (Resources)» Организатора объектов выбирается пункт «PLC-Browser», открывается окно режима «PLC-Browser».
5) **Получение информации о настройках контроллера.** В командной строке режима «PLC-Browser» вводится команда «PLCInfo», запрос передается ПЛК нажатием на клавиатуре кнопки <Enter>. В ответе контроллера отображаются значения действующих параметров для IP-адреса (IP), маски (MASK) и шлюза (GATE) подсети, см. рис Ж.2 (б).
6) **Изменение IP-адреса контроллера.** В командной строке режима «PLC-Browser» вводится команда и нужный адрес, например: «SetIP 10.0.6.11», команда передается ПЛК нажатием на клавиатуре кнопки <Enter>, в поле отображения реакции ПЛК на введенную команду появляется ответ контроллера, подтверждающий исполнение команды (см. рисунок Ж.3).

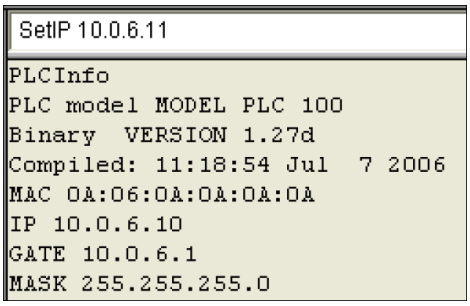


Рисунок Ж.3 – Пример ввода команды для изменения IP-адреса контроллера в локальной сети

7) **Изменение для контроллера значений маски (MASK) и шлюза (GATE) в подсети.** Выполняется командами «SetGate» и «SetMask». Последовательность выполнения аналогична последовательности п. 6.После смены значения параметра GATEнеобходима перезагрузка контроллера. Перезагрузку лучше производить отключением питания контроллера с последующим включением через пять и более секунд.
8) **Перезагрузка контроллера.** Осуществляется переводом тумблера на передней панели контроллера в положение «Сброс» и удержанием его в течение пяти или более секунд, либо отключением питания на время не менее пяти секунд с последующим включением; рекомендуется использовать перезагрузку выключением питания (при этом программное соединение разрывается).
9) **Включение соединения с контроллером.** В главном меню ПО CODESYS выбирается команда **Online | Login**, (предварительно может потребоваться подтверждение выбора соединения с контроллером через COM-порт компьютера – см. п. 2).

10) **Контроль сделанных изменений.** В командной строке режима «PLC-Browser» вводится команда «PLCInfo». Реакция ПЛК представлена на рисунке
Ж.4.

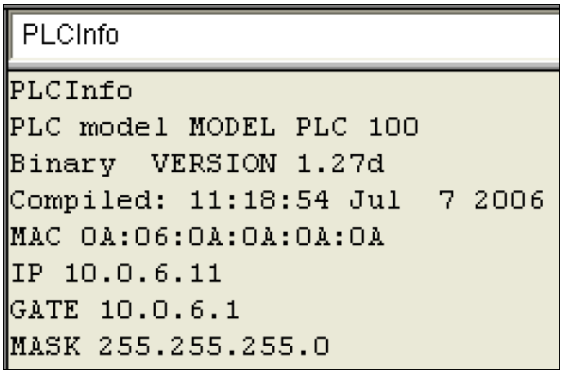


Рисунок Ж.4 – Информация о новых настройках контроллера для локальной сети

После этого связь с контроллером может быть установлена по интерфейсу Ethernet, работающему на новых сетевых настройках.

Приложение И. Перенос проекта CODESYS между несовместимыми версиями встроенного ПО контроллера

Если проект CODESYSсоздан для контроллера со встроенным ПО, версия которого несовместима с имеющейся на доступном контроллере, можно вручную, с помощью двух открытых экземпляров среды разработки CODESYS, преобразовать проект к нужной версии встроенного ПО. Фактически, такое преобразование есть поэтапное копирование различных частей проекта из одного его экземпляра, для несовместимой версии, в другой, вновь созданный, где target-файл выбран верно.

При конвертировании нужно учесть, что причины несовместимости могут быть следующими:

- различие имен и допустимых диапазонов значений некоторых переменных конфигурации, одинаковых по своему назначению,
- различия в обозначении адресов соответствующих входов и выходов,
- разница в наличии необходимых входов и выходов, в том числе служебного назначения,
- разница в объемах доступной памяти.

Перечисленные причины влекут за собой возможную необходимость исправления исходных текстов POU внутри нового проекта, или даже нереализуемость некоторых функций (в случае отсутствия в конфигурации) Обобщенная процедура переноса проекта с одной версии target-файла на другую описана ниже. Проект, который нужно перенести далее будет называться *«старый проект»,* а проект, в который происходит копирование – *«новый проект».*

1. Открыть старый проект и его копию (новый проект) для переноса

Перед переносом необходимо установить target-файлы для тех версий встроенного ПО, для которых существует старый вариант проекта и будет сформирован его новый вариант. Открыть два экземпляра CODESYS; в обоих открыть старую версию проекта, одно окно далее будет использоваться как источник данных по проекту («старый проект»), второй – как приемник («новый проект»). Рекомендуется в окне с проектом-приемником произвести сохранение проекта под другим именем.

2. В новом проекте изменить конфигурацию оборудования: выбрать таргет-файл, соответствующий контроллеру и версии прошивки, которые планируется использовать далее

Это можно сделать, перейдя на вкладку «Ресурсы» (рис. И.1, метка «1»), выбрав пункт списка «Настройки целевой платформы» (рис. И.1, метка «2»), и нажав на кнопку селектора, обозначенного как («Конфигурация» рис. И.1, метка «3»). В появившемся списке выбрать необходимый target-файл.

3. В новом проекте, в пункте меню «Дополнения» выбрать пункт «Стандартная конфигурация» (рис. И.2)

Выбрав пункт списка «Конфигурация ПЛК», в главном меню выбрать подменю «Дополнения», в подменю выбрать пункт «Стандартная конфигурация» и в появившемся окошке с сообщением щёлкнуть мышью на подтверждении того, что необходимо сбросить конфигурацию проекта до стандартной.

4. Последовательно, сохраняя порядок, создать в новом проекте все те модули, что были в старом проекте

В старом проекта открыть конфигурацию ПЛК, (пример см. рис. И.3) сравнить ее с базовой конфигурацией нового проекта, затем, сравнивая две конфигурации,

добавить в новый проект (рис. И.4) те модули, которых, по сравнению со старой конфигурацией, в нем не хватает. Добавлять следует в том, порядке, в котором они в старом проекте перечислены сверху вниз. Для добавления необходимо на вершине дерева конфигурации ПЛК (там, где на рис. И.3 надпись “PLC110_60”) нажать правую кнопку мыши и в появившемся меню выбрать «Добавить подэлемент», далее, в меню второго уровня тип подэлемента (рис. И.4).

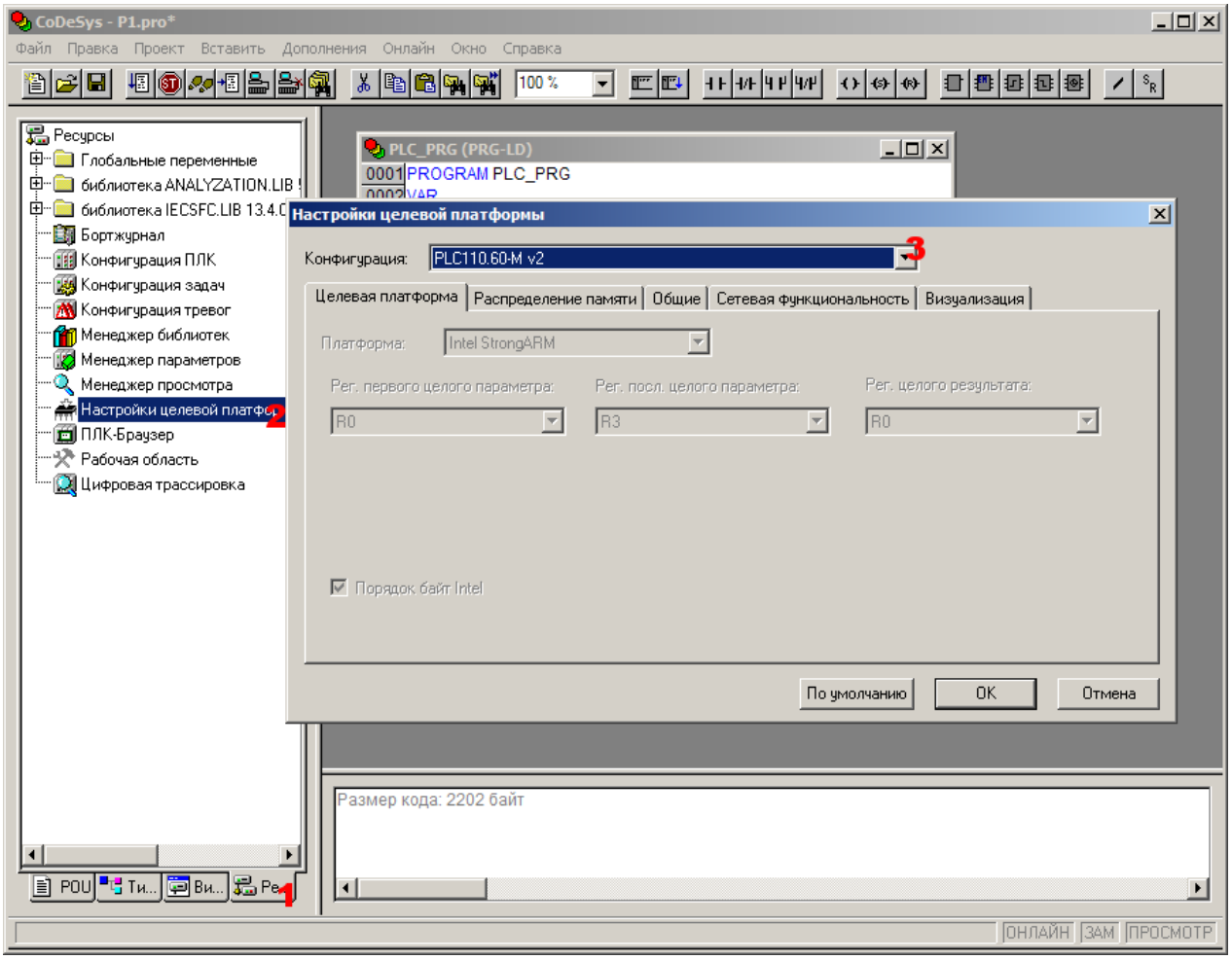


Рисунок И.1 – Смена target-файла в настройках нового проекта.

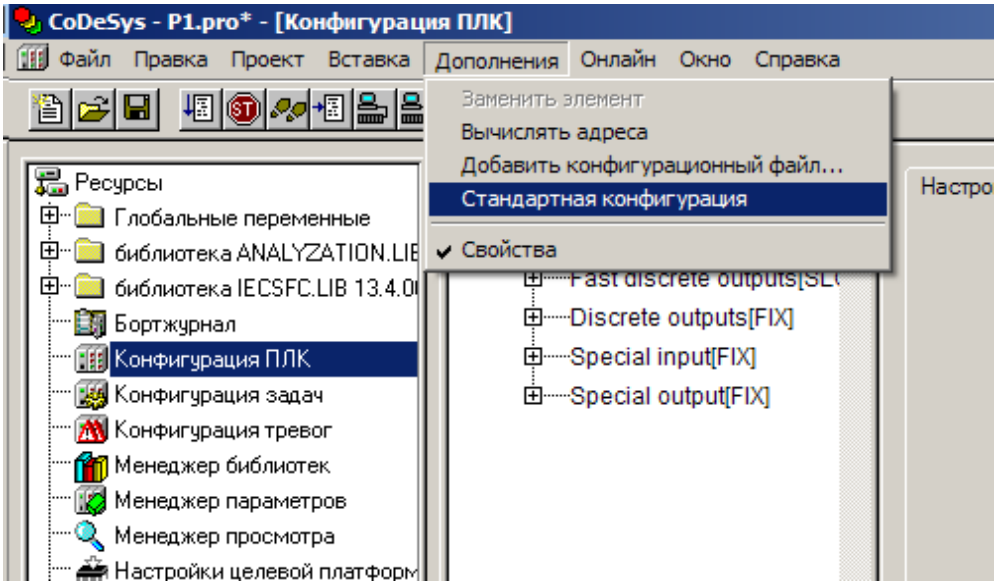


Рисунок И.2 – Сброс конфигурации проекта на стандартную

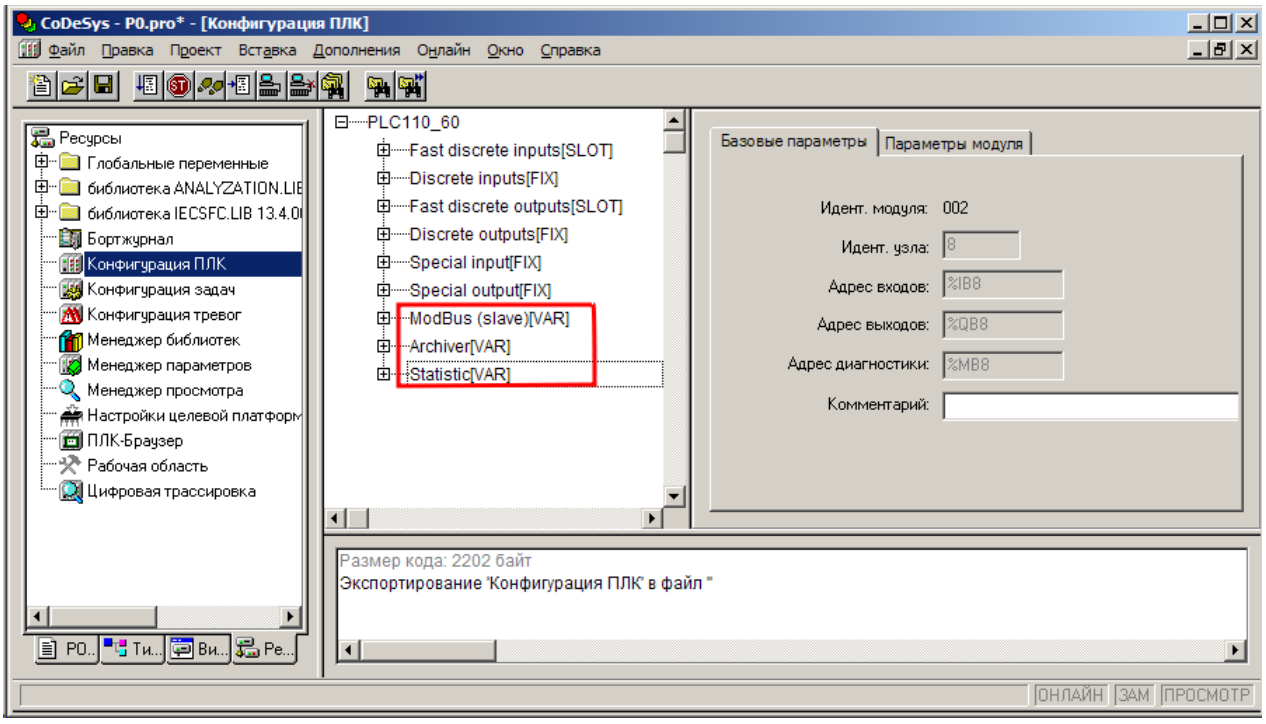


Рисунок И.3 – Конфигурация устройства; красной рамкой выделены добавленные модули

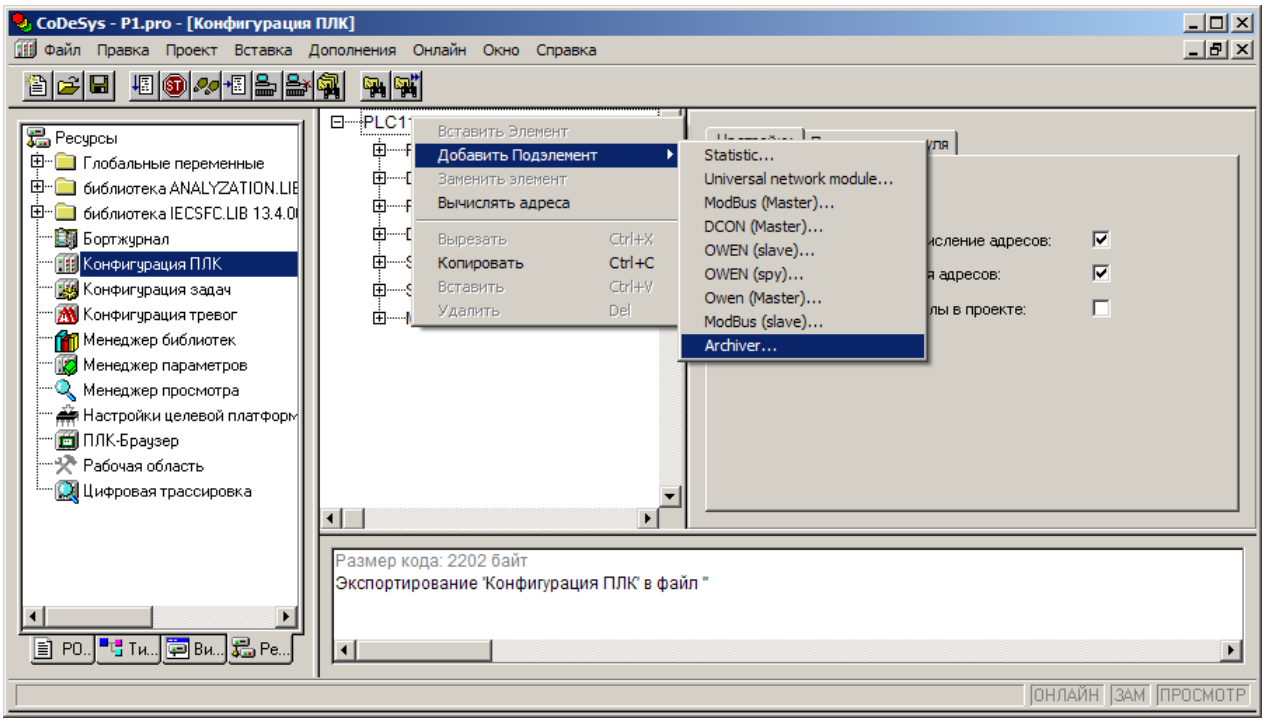


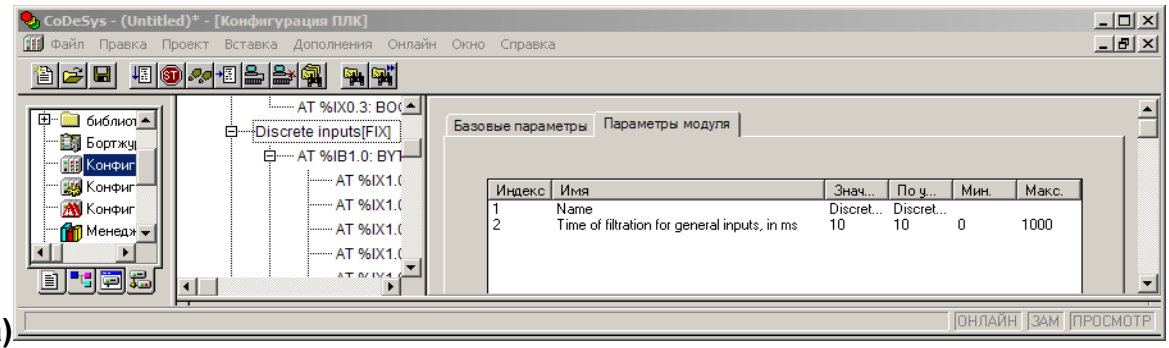
Рисунок И.4 – Добавление нового модуля в конфигурацию устройства в новом проекте

5. Задать модулям параметры в соответствии со старым проектом, с учётом различий в требуемых значениях временных и иных параметров в новой версии конфигурации

После создания в новом проекте всех модулей, которые есть в старом, нужно сконфигурировать модули. При конфигурировании следует учитывать, что некоторые параметры могут отсутствовать[12], а у других параметров может смениться способ их представления или диапазоны значений[13] (см. рис. И.5 а, б), в частности по-другому стали задаваться временные параметры модуля ШИМ.

[12] Например, могут отсутствовать некоторые параметры портов передачи данных RS-232, RS-485, или дискретных входов и выходов, например, в некоторых случаях, такой параметр, как Visibility.
[13] К таким параметрам относится постоянная времени фильтра быстрых входов-выходов. Постоянная времени фильтра уменьшилась в связи с увеличением быстродействия этого фильтра.

а)



б)

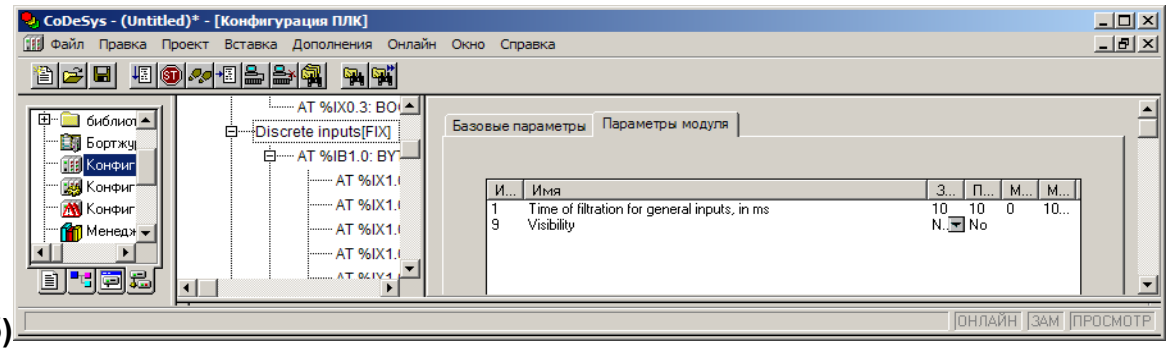
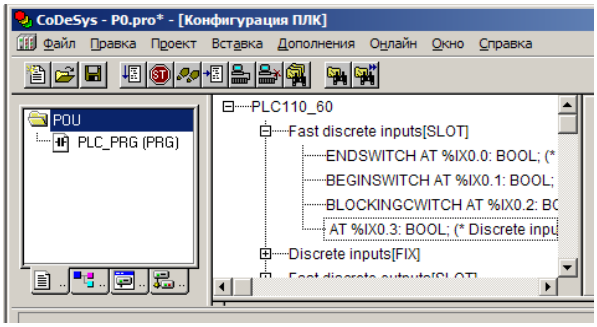


Рисунок И.5 – Пример различающихся параметров модулей в старом (а) и новом (б) проектах.

6. Задать каналам ввода-вывода имена в соответствии со старым проектом

Возможность задавать каналам ввода-вывода символические имена и использовать их в разработанной программе, избегая таким образом использования адресов, существенно облегчает перенос проекта между платформами. Если в старом проекте использовались символические имена, следует задать их в настройках объектов точно в том же виде и на тех же местах, как это было в старом проекте (см. рис. И.6).

а)



б)

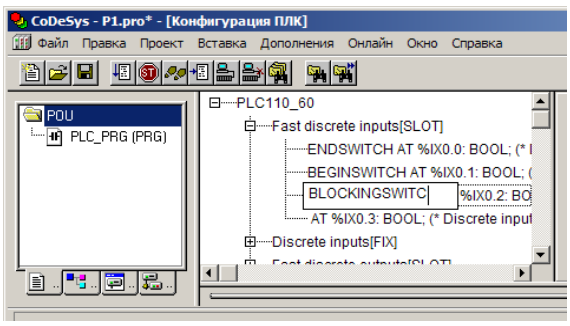


Рисунок И.6 – Пример символических имен каналов в старом (а) и новом (б) проектах.

7. Проверить распределение памяти входов/выходов

Конфигурация входов и выходов могла измениться. Новая версия (M02) контроллера ПЛК110 (target-файл версии 3.04) имеет следующие изменения:

- отсутствуют некоторые переменные в модуле «Статистика»: нет внутренней температуры контроллера и переменной оставшегося времени работы при работе от батареи,
- не требуется создание подмодулей при использовании переменных типа unsigned,

– все адреса ModBus как для устройства master, так и для устройства slave обозначаются, как **%Q**.

Таким образом, если в пользовательской программе использовались переменныевнутренней температуры контроллера и оставшегося времени работы при работе от батареи, подпрограммы, их использующие, необходимо реализовать другим способом, или не задействовать при компиляции проекта.

Подпрограммы, которые используют переменные типа unsigned, и в которых имеется обращение к переменным по адресам, могут нуждаться в модификации в связи с тем, что вид адресов переменных типа unsigned изменится. Также изменится вид адресов входов и выходов в модулях ModBus. Если данные адреса присутствуют в программе в чистом виде (например, в блоке объявления переменных, хотя могут и во всём тексте программы), то необходимо произвести замену адреса там, где это необходимо.

Если в конфигурации старой версии проекта входам и выходам присвоены имена, как показано выше (см. рис. И.6), и в тексте программы обращение к входам и выходам осуществляется только по присвоенным именам, то достаточно будет изменить конфигурацию входов и выходов контроллера, введя соответствующие имена соответствующим ячейкам памяти, связанным с вводом-выводом.

9. Провести коррекцию кода программы, если используются значения из модуля статистики (см. п.4 списка отличий) или модуля ШИМ (см. п.3 списка отличий)

Изменения, описанные в п. 5 могут коснуться не только конфигурации ПЛК, но и текста программы. Поэтому, при наличии обращениий к параметрам модуля ШИМ или модуля статистики в тексте конвертируемой программы необходимо проконтролировать каждый из фрагментов кода, содержащий данные обращения и модифицировать код, с целью сохранения его работоспособности, либо исключить его из проекта с целью сохранения работоспособности остального проекта.

10. Скомпилировать программу, настроить канал связи и загрузить её в ПЛК

Данные операции описаны в других главах инструкции, никаких специальных действий дополнительно описывать не требуется.

Приложение К. Установка драйвера подключения ПЛК по USB-device

Для связи контроллера с ПК необходимо произвести следующие действия:

- соединить кабелем «USB A-B» разъем «USB B», располагающийся на лицевой панели контроллера, и любой свободный USB-порт ПК;
- подать питание на контроллер и на ПК;
- при первом подключении через интерфейс USB компьютер запросит соответствующий драйвер;
- указать путь расположения драйвера (драйвер расположен на диске, который идет в комплекте с ПЛК, так же Вы можете скачать драйвер с сайта www.owen.ru);
- если драйвер успешно установился, то в диспетчере устройств на ПК появляется новый порт «PLC110 USB virtual serial port»; таким образом подключение ПЛК к ПК распознается как добавление нового COM-порта к ПК с присвоением порту индивидуального порядкового номера (например, COM4);
- далее подключение к контроллеру осуществляется так же, как если бы контроллер подключался через физический COM-порт ПК.

Если драйвер не установился по умолчанию, или возникли проблемы с его установкой, то его установку в дальнейшем можно выполнить вручную. После неудачной попытки установки в диспетчере устройств ПК можно будет увидеть, что в систему добавлено неизвестное устройство, которое не работает (см. рисунок К.1), это и есть новый последовательный порт, которым контроллер подключен к ПК.

Для того, чтобы порт стал работоспособным, необходимо установить драйвер подключения контроллера по USB. Драйвер можно найти на диске, поставляющемся с контроллером, в папке “Драйвера”.

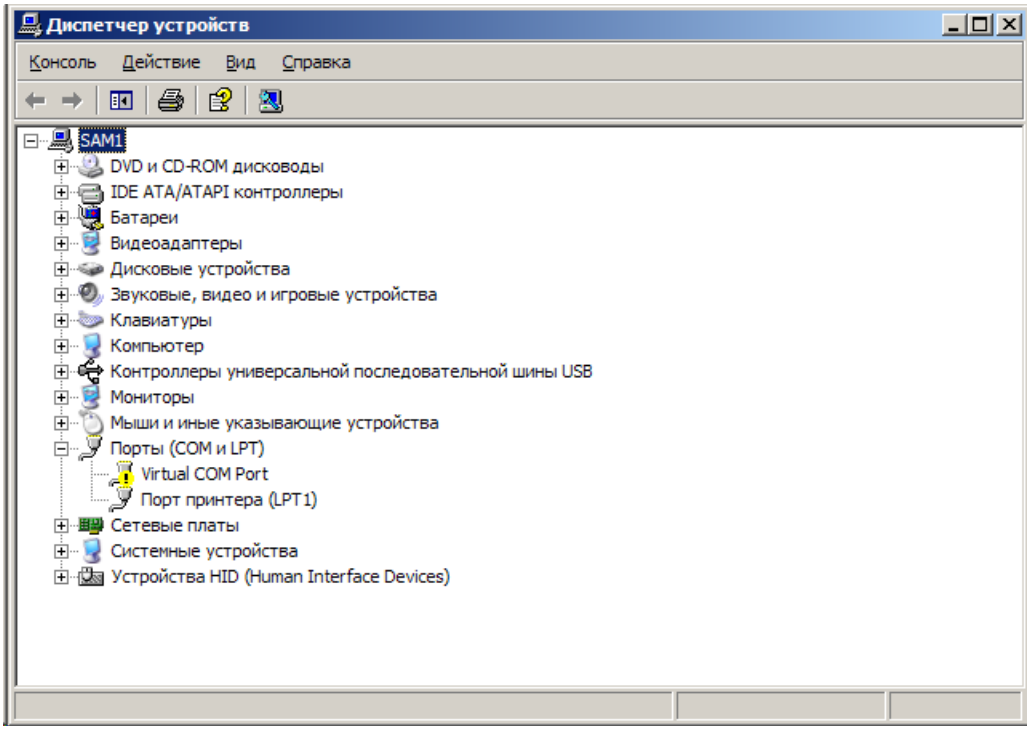


Рисунок К.1 – Диспетчер устройств с обнаруженным, но не установленным подключением к ПЛК

Процесс установки драйвера в Windows XP SP2 и в Windows XP SP3 происходит следующим образом. Сначала щелкнуть в открытом диспетчере устройств правой кнопкой мыши по появившемуся пункту с неработающим устройством «Virtual-COMPort» (с меткой желтого цвета). В выпадающем меню выбрать пункт «Обновить драйвер». На экране появится обычное окно обновления драйверов Windows (рисунок К.2). На вопрос, подключиться ли к узлу “WindowsUpdate” выбрать радио-кнопку «Нет, не в этот раз» и нажать кнопку «Далее».

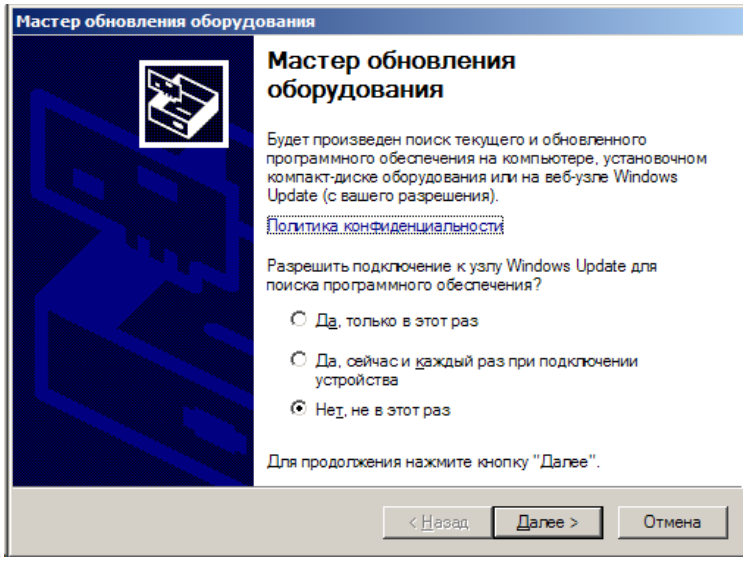


Рисунок К.2 – Начало работы мастера установки оборудования

Так как установка драйвера производится из известного источника, нет необходимости в сканировании дисковых устройств в поисках драйверов. На следующем шаге необходимо выбрать радио-кнопку «Установка из указанного места» (рисунок К.3) и для продолжения нажать кнопку «Далее».

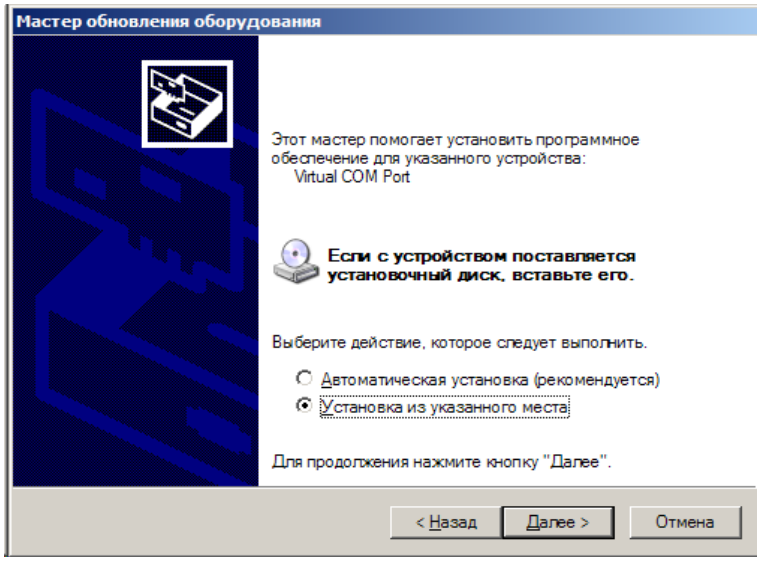


Рисунок К.3 – Выбор источника файлов драйверов

Следующим шагом пользователь отказывается от автоматического выбора драйвера системой, сообщая ей, что драйвер будет выбран вручную, после непосредственного указания пути, по которому расположены файлы, необходимые для

установки драйвера. В окне (рисунок К.4) выбрать радио-кнопку «Не выполнять поиск, Я сам выберу подходящий драйвер» и нажать кнопку «Далее».

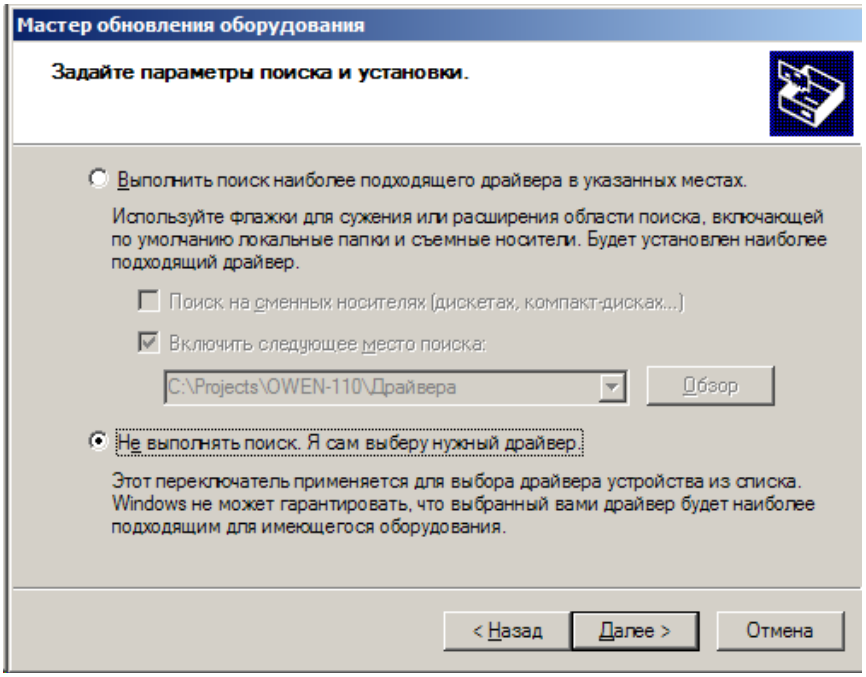


Рисунок К.4 – Определение ручного либо автоматического выбора нужного драйвера

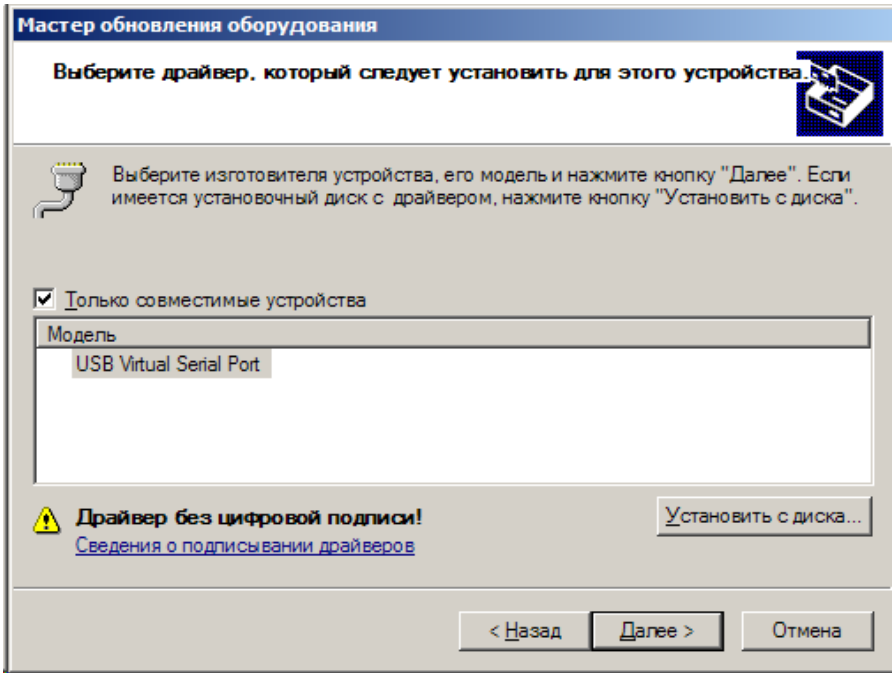


Рисунок К.5 – Окно выбора драйвера

После того, как окно установки примет вид, аналогичный рисунку К.5 (возможны неточности, например, таблица в средней части окна может содержать другие строки, или быть пустой), нажать кнопку «Установить с диска…». На экране появится окно (рисунок К.6), в котором нужно нажать кнопку «Обзор» и в окне выбора файлов и папок (рисунок К.7) выбрать папку, содержащую файл с расширением “.INF” для

установки драйвера. Имя файла – “PLC110 USB COM WIN7.inf”. Нажать на кнопку «Открыть».

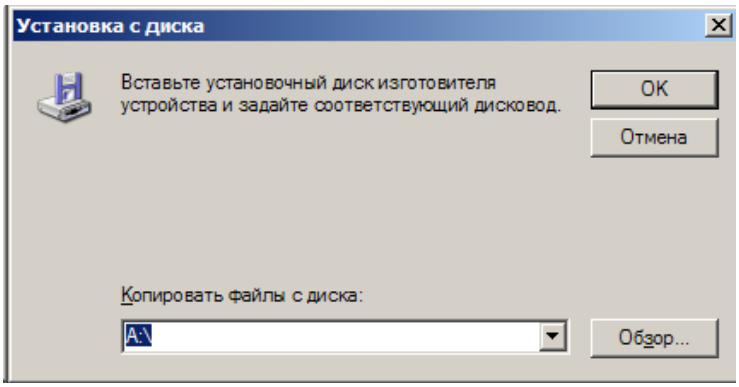


Рисунок К.6 – Окно задания имени файла (пути к файлам)

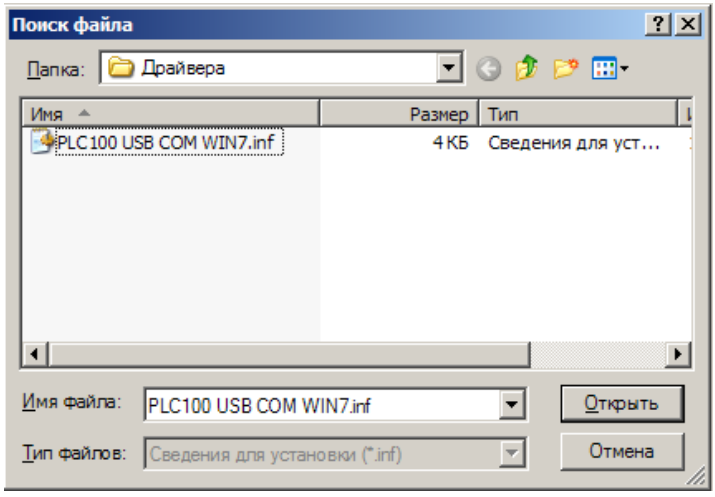


Рисунок К.7 – Окно выбора файлов

После нажатия на кнопку «Открыть» окно выбора файлов исчезнет, и в строке ввода окна (рисунок К.6) появится путь к файлам установки драйверов порта, например “E:\OWEN-110\Драйвера”. Далее следует нажать на кнопку “OK”.

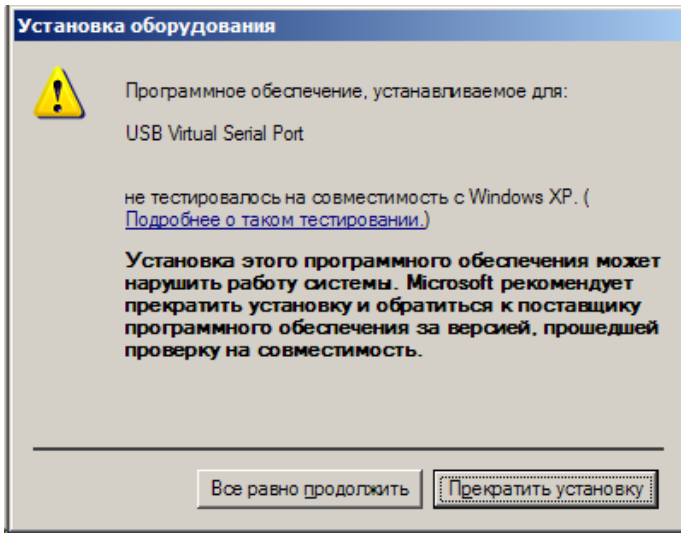


Рисунок К.8 – Предупреждение о драйвере, не сертифицированном Microsoft

Возможно, что в процессе установки появится окно, предупреждающее о том, что устанавливаемый драйвер не сертифицирован компанией Microsoft (рисунок К.8), при этом следует нажать на кнопку «Все равно продолжить». При этом окно исчезнет, и процесс установки драйвера продолжится.

Процесс установки драйвера начинается с появления окна (рисунок К.9), в котором ход копирования файлов, внесения изменений в системные файлы и другие операции по установке отображается на индикаторе; над индикатором указывается имя копируемого файла и полное имя каталога, в который файл будет скопирован.

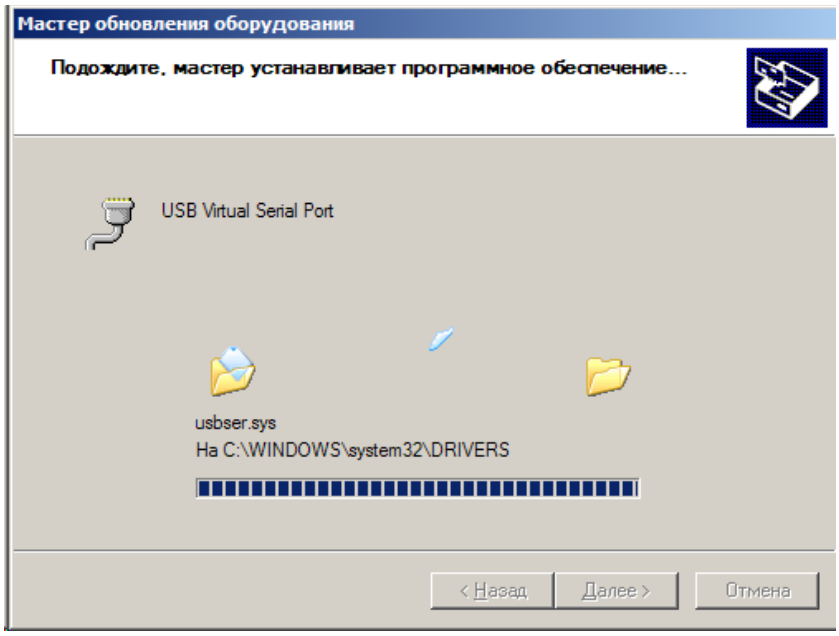


Рисунок К.9 – Ход установки драйвера

После успешного завершения всех операций установки драйвера появляется окно, аналогичное рисунку К.10. Для продолжения работы следует нажать кнопку «Готово».

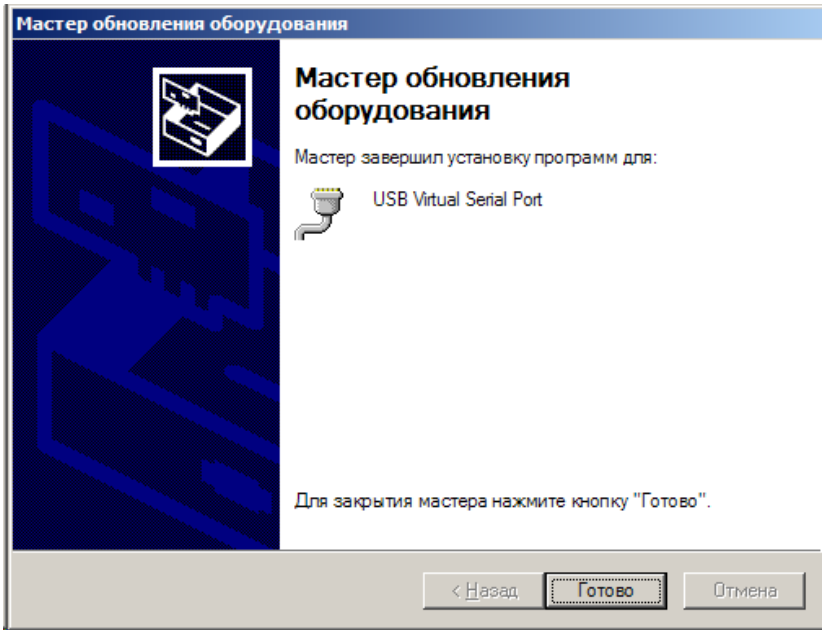


Рисунок К.10 – Конец установки

В случае успешной установки драйвера подключения контроллера с помощью порта “USB B” окно диспетчера устройств отобразит новое устройство так, как показано на рисунке К.11. После установки следует закрыть окно диспетчера устройств, а также окно свойств системы нажатием на кнопку “OK”.

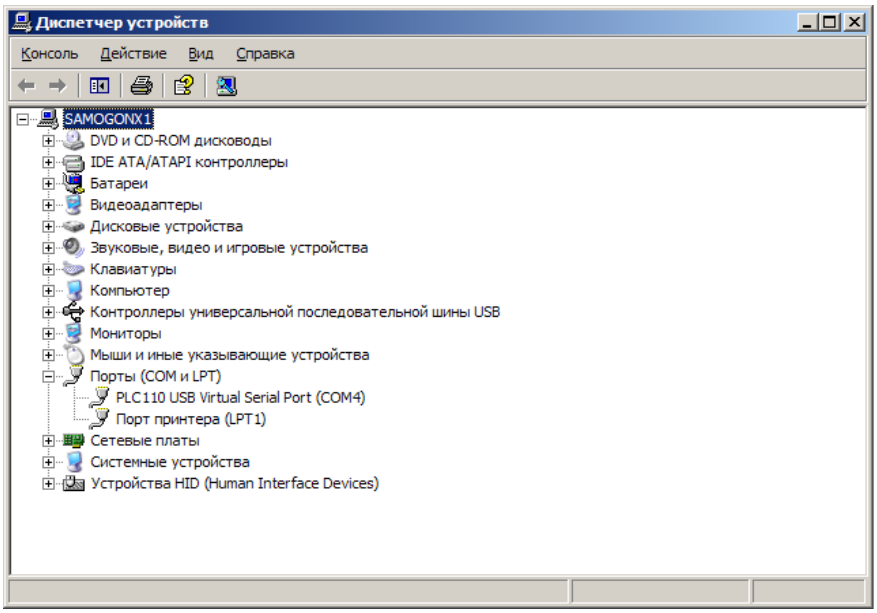


Рисунок К.11 – Диспетчер устройств после успешной установки драйвера.

После установки можно сразу приступать к работе с CODESYS: созданию проекта, настройке подключения, программированию контроллера. При настройке подключения указывать номер COM-порта, соответствующий отображенному в диспетчере устройств (в рассмотренном примере, на рисунке К.11 это порт COM4).

Лист регистрации изменений

Версия документа	Дата выпуска	Содержание изменений