

ООО «ПО Промсвязь»

---

## **СТАБУР: Первый старт в CODESYS V3.5**

Версия документа: 1.0

Дата: Апрель 2026

Производитель: ООО «ПО Промсвязь», г. Екатеринбург

Сайт: [psve.ru](http://psve.ru)

Екатеринбург · [www.psve.ru](http://www.psve.ru)

## Содержание

|                                                                  |    |
|------------------------------------------------------------------|----|
| 1 Введение .....                                                 | 5  |
| 1.1 Назначение документа .....                                   | 5  |
| 1.2 Условные обозначения.....                                    | 5  |
| 1.3 Что такое CODESYS V3.5 .....                                 | 6  |
| Состав среды разработки .....                                    | 6  |
| Архитектура: IDE и рантайм .....                                 | 7  |
| Таргет-файл (device package).....                                | 7  |
| 1.4 Рекомендуемая версия среды и компилятора.....                | 7  |
| 1.5 Линейка приборов СТАБУР с поддержкой CODESYS .....           | 8  |
| 1.6 Что потребуется для начала работы .....                      | 10 |
| 2 Установка программного обеспечения .....                       | 12 |
| 2.1 Установка CODESYS Development System.....                    | 12 |
| 2.2 Установка таргет-файла StaburTarget .....                    | 12 |
| 2.3 Проверка установки .....                                     | 14 |
| 3 Быстрый старт .....                                            | 16 |
| 3.1 Создание проекта.....                                        | 16 |
| 3.2 Подключение по mini-USB (RNDIS) .....                        | 18 |
| 3.3 Подключение по Ethernet.....                                 | 19 |
| 3.4 Установка связи в CODESYS .....                              | 21 |
| 3.5 Компиляция, загрузка и запуск .....                          | 21 |
| 4 Настройки устройства и компонентов в дереве проекта .....      | 24 |
| 4.1 Узел контроллера (Device).....                               | 24 |
| 4.2 Энергонезависимая память (Retain).....                       | 25 |
| 4.2.1 Конфигурация Retain.....                                   | 26 |
| 4.2.2 Каналы Retain (вкладка «Соотнесение входов/выходов») ..... | 26 |
| 4.2.3 Служебные вкладки Retain .....                             | 27 |
| 4.3 Управление индикаторами (LEDS).....                          | 27 |
| 4.4 Корзина модулей и добавление модулей ввода-вывода .....      | 28 |
| 4.5 Узел Application и задачи.....                               | 31 |
| 5 Интерфейс CODESYS .....                                        | 33 |
| 5.1 Главное окно среды .....                                     | 33 |

|                                                  |    |
|--------------------------------------------------|----|
| 5.2 Дерево устройств .....                       | 33 |
| 5.3 Редакторы программ .....                     | 34 |
| 5.4 Панель сообщений.....                        | 35 |
| 5.5 Меню и горячие клавиши .....                 | 35 |
| 6 Кратко о программировании .....                | 38 |
| 6.1 Языки МЭК 61131-3 .....                      | 38 |
| 6.2 Типы данных .....                            | 39 |
| 6.2.1 Логический тип .....                       | 39 |
| 6.2.2 Целые типы .....                           | 39 |
| 6.2.3 Байтовые типы .....                        | 40 |
| 6.2.4 Вещественные типы .....                    | 40 |
| 6.2.5 Строки.....                                | 40 |
| 6.2.6 Временные типы .....                       | 41 |
| 6.2.7 Составные типы .....                       | 41 |
| 6.3 Переменные и области видимости.....          | 42 |
| 6.3.1 Блоки объявления переменных .....          | 42 |
| 6.3.2 Модификаторы переменных.....               | 43 |
| 6.3.3 Именованые переменных .....                | 44 |
| 6.4 Программные объекты (POU) .....              | 45 |
| 6.4.1 PROGRAM – Программа .....                  | 46 |
| 6.4.2 FUNCTION_BLOCK – Функциональный блок ..... | 47 |
| 6.4.3 FUNCTION – Функция.....                    | 47 |
| 6.4.4 Сравнение типов POU .....                  | 48 |
| 6.5 Простой пример на ST .....                   | 48 |
| 7 Создание визуализации.....                     | 51 |
| 7.1 TargetVisu и WebVisu .....                   | 51 |
| 7.1.1 Visualization Manager.....                 | 51 |
| 7.1.2 Настройки Visualization Manager .....      | 52 |
| 7.1.3 Настройки Таргет-визуализации.....         | 53 |
| 7.1.4 Настройки Веб-визуализации.....            | 54 |
| 7.1.5 Задание размера экрана визуализации .....  | 55 |
| 7.2 Создание экрана визуализации .....           | 56 |

|                                                       |    |
|-------------------------------------------------------|----|
| 7.3 Элементы и привязка переменных .....              | 57 |
| 7.4 Фреймы .....                                      | 59 |
| 8 Дополнительные материалы и проверка результата..... | 60 |
| 8.1 Чек-лист после первой загрузки.....               | 60 |
| 8.2 Диагностика модулей.....                          | 60 |
| 8.3 Куда обращаться и полезные ссылки .....           | 61 |

# 1 Введение

## 1.1 Назначение документа

Настоящий документ – практическое руководство для инженера, который впервые начинает работу с приборами серии СТАБУР (ПЛК и панели оператора) в среде программирования CODESYS V3.5.

Документ последовательно проводит через все этапы: от установки программного обеспечения и таргет-файла до создания первого проекта, подключения к прибору, загрузки программы и проверки её работоспособности. Помимо этого рассмотрены структура дерева устройств, интерфейс среды разработки, основы программирования по стандарту МЭК 61131-3 и создание визуализации на экране прибора.

Документ не заменяет полное Руководство по эксплуатации (РЭ) и Руководство по программированию CODESYS (РП), а дополняет их – давая концентрированный маршрут для быстрого старта.

### Сопутствующие документы:

Таблица 1.1 – Сопутствующие документы

| Документ                                     | Содержание                                                                               |
|----------------------------------------------|------------------------------------------------------------------------------------------|
| Руководство по эксплуатации ПЛК СТАБУР (РЭ)  | Технические характеристики, монтаж, модули, обновление прошивки, настройка сети          |
| Руководство по программированию CODESYS (РП) | Таргет-файл, описание устройств, модули I/O в CODESYS, Modbus, CANopen, загрузка проекта |
| Справка CODESYS (F1 в среде)                 | Полное описание IDE, языков, визуализации, библиотек                                     |

Все актуальные версии руководств и дистрибутивов доступны на сайте [psve.ru](http://psve.ru).

## 1.2 Условные обозначения

В настоящем документе применяются два вида выделенных блоков.

**Блок «Примечание»** (синяя рамка) содержит дополнительную информацию, советы и пояснения, которые полезны, но не являются обязательными для выполнения. Пример:

**Примечание.** Это пример информационного блока. Текст выделен синей рамкой и синим фоном. Такие блоки содержат советы, уточнения и ссылки на дополнительные источники.

**Блок «Внимание»** (красная рамка) содержит предупреждения о действиях, которые могут привести к неправильной работе оборудования, потере данных или повреждению прибора. Пример:

**! Внимание.** Это пример блока предупреждения. Текст выделен красной рамкой и красным фоном. Такие блоки необходимо выполнять строго.

Нумерованные перечни в документе обозначают **обязательную последовательность шагов** и нумеруются с 1 в каждом разделе.

### 1.3 Что такое CODESYS V3.5

CODESYS (Controller Development System) – интегрированная среда разработки (IDE) для программируемых логических контроллеров, разработанная немецкой компанией 3S-Smart Software Solutions. Среда является платформонезависимой: один и тот же инструмент используется для программирования контроллеров сотен производителей по всему миру, что делает знания и навыки работы в CODESYS универсальными.

CODESYS реализует международный стандарт МЭК 61131-3, определяющий языки программирования ПЛК, типы данных, структуру программ и правила обмена данными. Соответствие стандарту обеспечивает переносимость кода между проектами и платформами.

#### Состав среды разработки

Таблица 1.3 – Компоненты среды разработки CODESYS

| Компонент                           | Назначение                                                           |
|-------------------------------------|----------------------------------------------------------------------|
| Редакторы языков МЭК 61131-3        | Разработка программ на ST, LD, FBD, SFC, IL, CFC                     |
| Конфигуратор устройств              | Настройка дерева проекта: контроллер, модули, полевые шины           |
| Менеджер задач (Task Configuration) | Управление циклическими и событийными задачами, приоритетами         |
| Средства отладки и мониторинга      | Точки останова, пошаговое выполнение, принудительная запись значений |
| Трассировка                         | Запись изменений переменных во времени с отображением графиков       |
| Редактор визуализации               | Разработка операторских экранов (TargetVisu, WebVisu)                |
| Менеджер библиотек                  | Подключение стандартных и пользовательских библиотек функций         |
| Менеджер пакетов                    | Установка таргет-файлов производителей и дополнительных компонентов  |
| CODESYS Gateway                     | Коммуникационный сервис для связи IDE с контроллерами по сети        |

## Архитектура: IDE и рантайм

CODESYS состоит из двух независимых частей:

- **CODESYS Development System** – IDE, устанавливаемая на компьютер инженера. Здесь создаётся, компилируется и отлаживается проект.
- **CODESYS Runtime** – исполняющая система, встроенная в прошивку контроллера. Она получает скомпилированный байт-код из IDE и выполняет его в реальном времени.

На приборах СТАБУР рантайм CODESYS работает под управлением операционной системы Linux на процессоре Rockchip RK3568.

## Таргет-файл (device package)

Для того чтобы CODESYS «узнала» конкретный контроллер, производитель предоставляет **таргет-файл** – пакет формата .package. Он содержит описание аппаратных ресурсов устройства: доступные модули ввода-вывода, интерфейсы, объём Retain-памяти, индикаторы, параметры соединения и прочее.

После установки таргет-файла через **Инструменты** → **Менеджер пакетов** в каталоге устройств CODESYS появляются модели приборов СТАБУР. Без установленного таргет-файла создать проект для конкретного прибора невозможно.

**Примечание.** Актуальные версии таргет-файла StaburTarget, прошедшие внутреннее тестирование, публикуются на сайте [psve.ru](http://psve.ru). Рекомендуется использовать версии оттуда.

## 1.4 Рекомендуемая версия среды и компилятора

Таблица 1.4 – Рекомендуемая версия среды и компилятора

| Параметр                   | Значение                                                                                        |
|----------------------------|-------------------------------------------------------------------------------------------------|
| Среда разработки           | CODESYS Development System V3, версия <b>3.5.16 SP4</b>                                         |
| Версия компилятора проекта | <b>3.5.16.40</b> (задаётся в Project Settings → Compiler Version)                               |
| Таргет-файл                | StaburTarget[версия].package – скачать актуальную версию с <a href="http://psve.ru">psve.ru</a> |

Версия CODESYS может быть новее, чем указано, однако в свойствах проекта необходимо зафиксировать версию компилятора на 3.5.16.40 – это версия, под которую собран таргет-файл.

**! Внимание.** Использование компилятора более ранней версии с текущим таргетом не поддерживается и приведёт к ошибкам при сборке или работе проекта.

## 1.5 Линейка приборов СТАБУР с поддержкой CODESYS

Все приборы серии СТАБУР, работающие с CODESYS, построены на единой аппаратной платформе: 4-ядерный 64-разрядный процессор ARM Cortex-A55 (2 ГГц), ОС реального времени Linux RT, TFT-дисплей с сенсорной панелью. Различия между моделями – в размере экрана, наличии корзин для модулей и комплектации интерфейсами.

**ПЛК СТАБУР** – программируемый логический контроллер с экраном и модульной корзиной. Основное применение – управление технологическим оборудованием.

**Панель оператора (ПО) СТАБУР** – прибор для визуализации, контроля и управления. Аппаратно идентичен ПЛК; программируется в CODESYS по тем же правилам.

Таблица 1.5 – Линейка приборов СТАБУР и базовая комплектация

| Тип прибора                                  | Диагональ экрана                         | Разрешение                                                                         | Корзины модулей                               | Базовая комплектация интерфейсов и питания                                                     |
|----------------------------------------------|------------------------------------------|------------------------------------------------------------------------------------|-----------------------------------------------|------------------------------------------------------------------------------------------------|
| ПЛК СТАБУР                                   | 5" / 7" / 10,1"                          | 800×480 /<br>800×480 /<br>1024×600                                                 | 1 корзина (до 8 модулей)                      | PS24 в слоте 4; модуль ETH232 не является обязательным                                         |
| ПО СТАБУР                                    | 5" / 7" / 10,1"                          | 800×480 /<br>800×480 /<br>1024×600                                                 | 1 корзина (до 8 модулей)                      | ETH232 в слоте 1 и PS24 в слоте 4                                                              |
| ПЛК СТАБУР /<br>ПО СТАБУР<br>(вариант 10,4") | 10,4"                                    | 1024×768                                                                           | Без корзины модулей                           | Ethernet, RS-232, 3xRS-485, USB-host, аудио-вход и выход, а также встроенный блок питания 24 В |
| ПЛК СТАБУР /<br>ПО СТАБУР                    | 12,1" / 15" / 17" /<br>19" / 21,5" / 24" | 1024×768 /<br>1024×768 /<br>1280×1024 /<br>1280×1024 /<br>1920×1080 /<br>1920×1080 | Без корзин / 1 / 2<br>корзины (до 16 модулей) | Ethernet, RS-232, 3xRS-485, USB-host, аудио-вход и выход, а также встроенный блок питания 24 В |

**Примечание.** Для приборов 5"–10,1" ключевое различие комплектации: у ПЛК базово установлен модуль PS24 (слот 4), у панели оператора базово установлены ETH232 (слот 1) и PS24 (слот 4). Вариант 10,4" является отдельным исполнением без корзины модулей: интерфейсы и питание встроены в корпус, установка ETH232 и PS24 не требуется. Для приборов от 12,1" действует тот же принцип встроенных интерфейсов и встроенного питания.

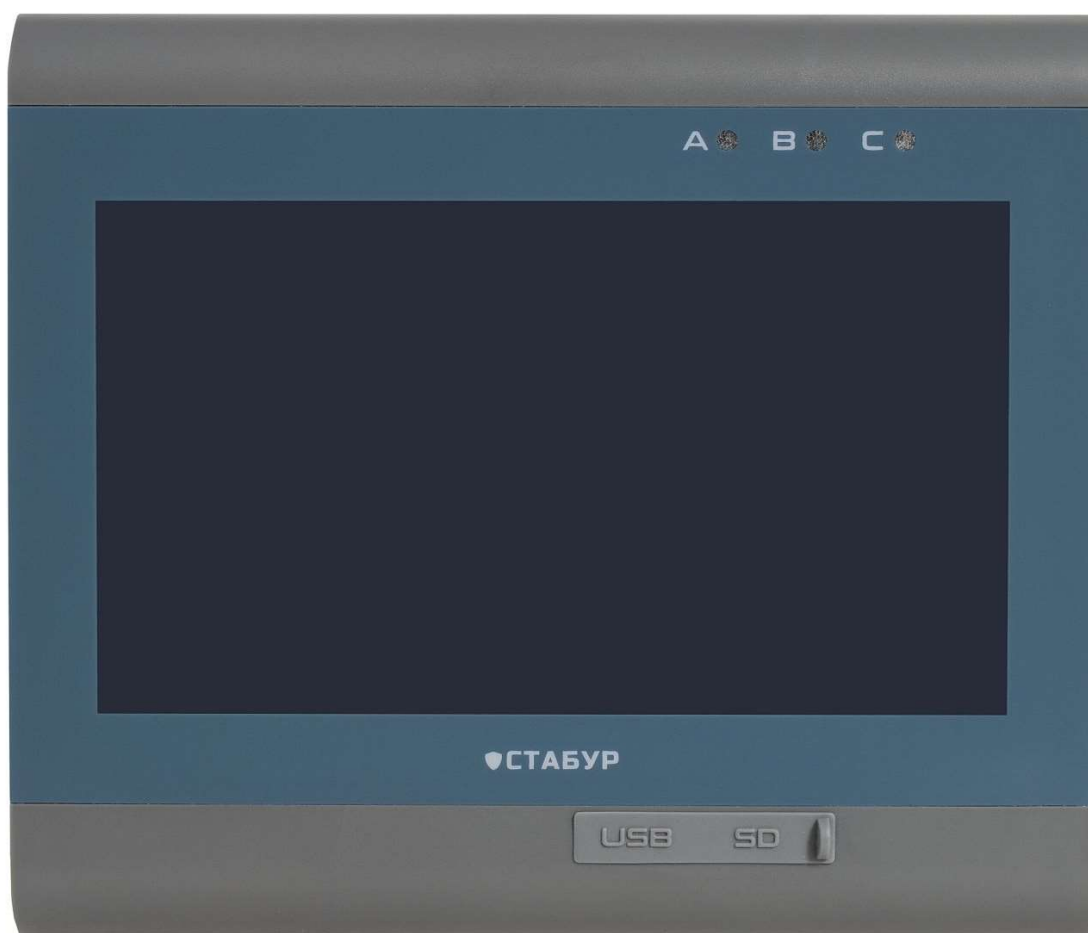


Рисунок 1.1 – Внешний вид ПЛК СТАБУР 07 (лицевая сторона): экран, светодиоды А/В/С, разъём micro-SD и mini-USB OTG под силиконовой заглушкой

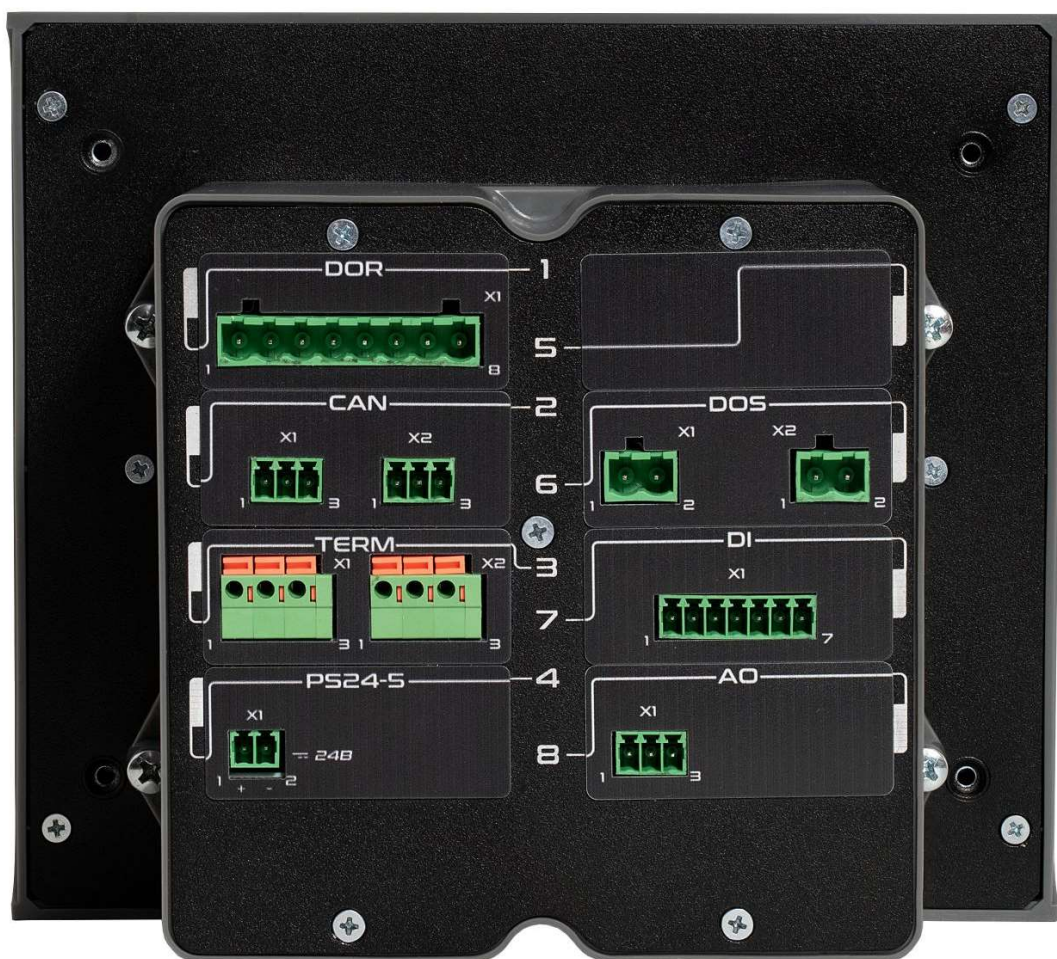


Рисунок 1.2 – Задняя сторона ПЛК СТАБУР 07: слоты модулей, маркировка разъемов

## 1.6 Что потребуется для начала работы

Для прохождения данного руководства подготовьте:

1. **Прибор СТАБУР** (ПЛК или ПО) с актуальной прошивкой.
2. **Персональный компьютер** с ОС Windows 10/11 (64-bit).
3. **Кабель Ethernet** (RJ45) – для подключения по сети; тип и длину кабеля выбирайте по требованиям вашей схемы подключения и документации на сетевое оборудование. Либо **кабель mini-USB** из комплекта поставки – для подключения по USB (RNDIS).
4. **Источник питания 24 В** постоянного тока для прибора. Для приборов 5"-10,1" питание подаётся через модуль PS24 в слоте 4; для варианта 10,4" и приборов 12"+ – через встроенный блок питания.
5. **Установщик CODESYS Development System V3** – с [psve.ru](http://psve.ru).
6. **Таргет-файл** StaburTarget[версия].package – с [psve.ru](http://psve.ru).

**Рекомендация.** Перед началом работы используйте актуальные версии прошивки и таргет-файла с [psve.ru](http://psve.ru), прошедшие внутреннее тестирование перед публикацией. Номер прошивки можно посмотреть в системной утилите прибора или в терминале (например, через PuTTY при входе root без пароля), номер таргет-файла – в имени файла .package.

## 2 Установка программного обеспечения

### 2.1 Установка CODESYS Development System

1. Скачайте установщик CODESYS Development System V3 (версия 3.5.16 SP4, 64-bit) с сайта [psve.ru](http://psve.ru) или из официального магазина CODESYS Store.
2. Запустите установщик **от имени администратора** (правый клик → «Запуск от имени администратора»).
3. В окне выбора компонентов убедитесь, что отмечены:
  - **CODESYS Development System** (среда разработки);
  - **CODESYS Gateway** (шлюз связи с прибором).
4. Завершите установку. Перезагрузка компьютера обычно не требуется.

**Рекомендация.** Добавьте каталог установки CODESYS и рабочую папку проектов в **исключения антивируса**. Это предотвратит замедление компиляции и ложные срабатывания при загрузке проекта.

### 2.2 Установка таргет-файла StaburTarget

Таргет-файл – пакет формата .package, который добавляет в CODESYS описание всех устройств серии СТАБУР: контроллеров, модулей ввода-вывода, интерфейсных модулей, компонентов Retain и LEDS.

#### Порядок установки:

1. Запустите CODESYS.
2. Откройте менеджер пакетов: меню **Инструменты** → **Менеджер пакетов** (или **Tools** → **Package Manager** при английском интерфейсе).

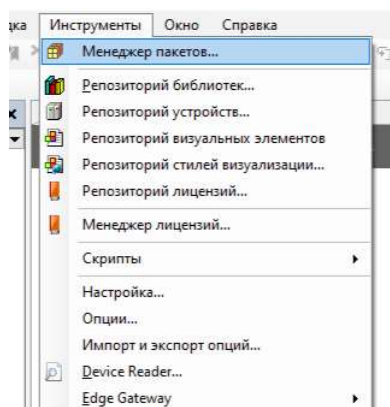


Рисунок 2.1 –Запуск менеджера пакетов: меню Инструменты → Менеджер пакетов

3. В открывшемся окне нажмите кнопку **Установить** (Install) и укажите путь к скачанному файлу, например StaburTarget[1.0.0.7].package.

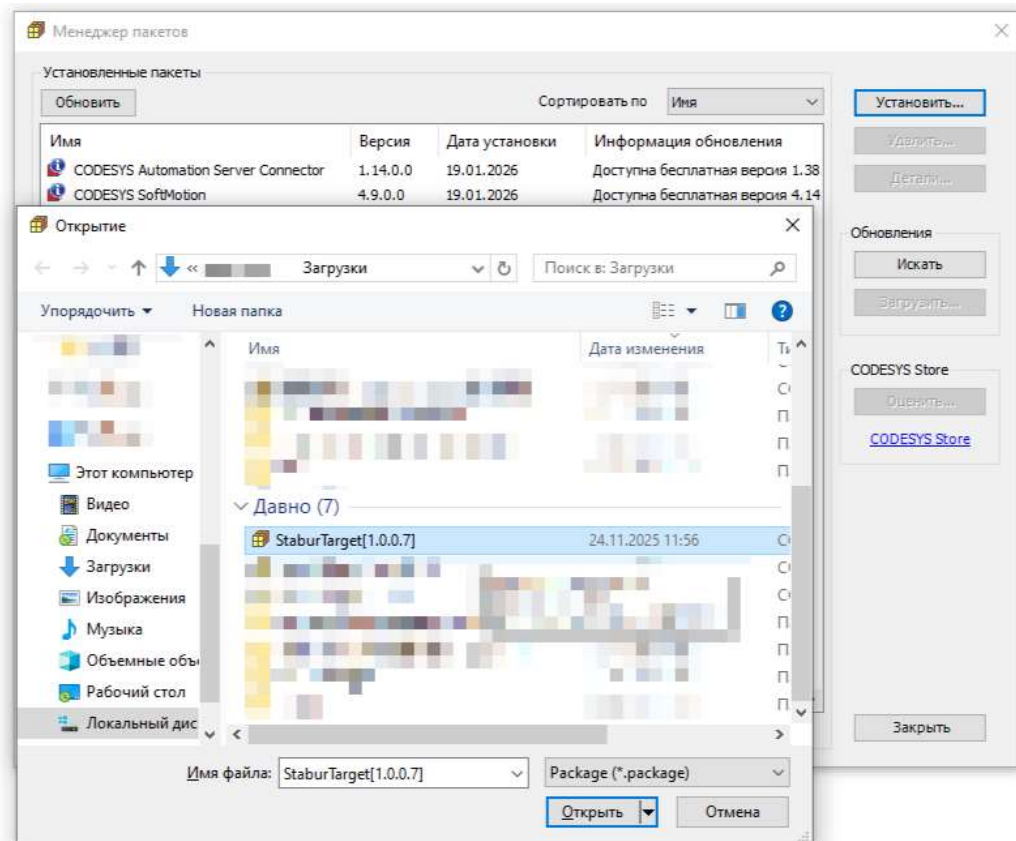


Рисунок 2.2 – Окно менеджера пакетов: выбор файла .package

4. Выберите **Полная установка** (Complete) и нажмите **Next**.

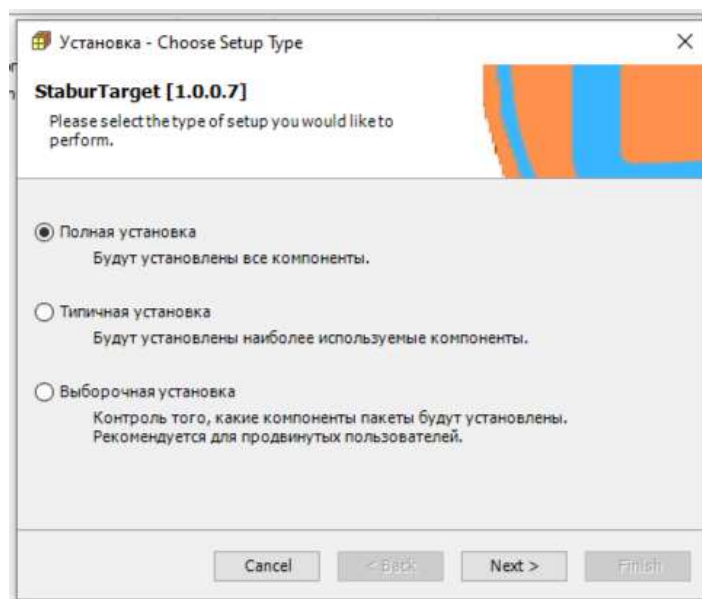


Рисунок 2.3 – Выбор типа установки: «Полная установка»

5. При появлении окна Контроля учётных записей (UAC) подтвердите разрешение.
6. Дождитесь окончания установки и закройте менеджер пакетов.

**Если возникает ошибка доступа (Access Denied):** закройте CODESYS, запустите его заново от имени администратора и повторите установку.

## 2.3 Проверка установки

После установки таргет-файла необходимо убедиться, что всё прошло корректно:

1. Перезапустите CODESYS (закройте и откройте заново).
2. Откройте менеджер пакетов (**Инструменты** → **Менеджер пакетов**). В списке установленных пакетов должен присутствовать **StaburTarget**.

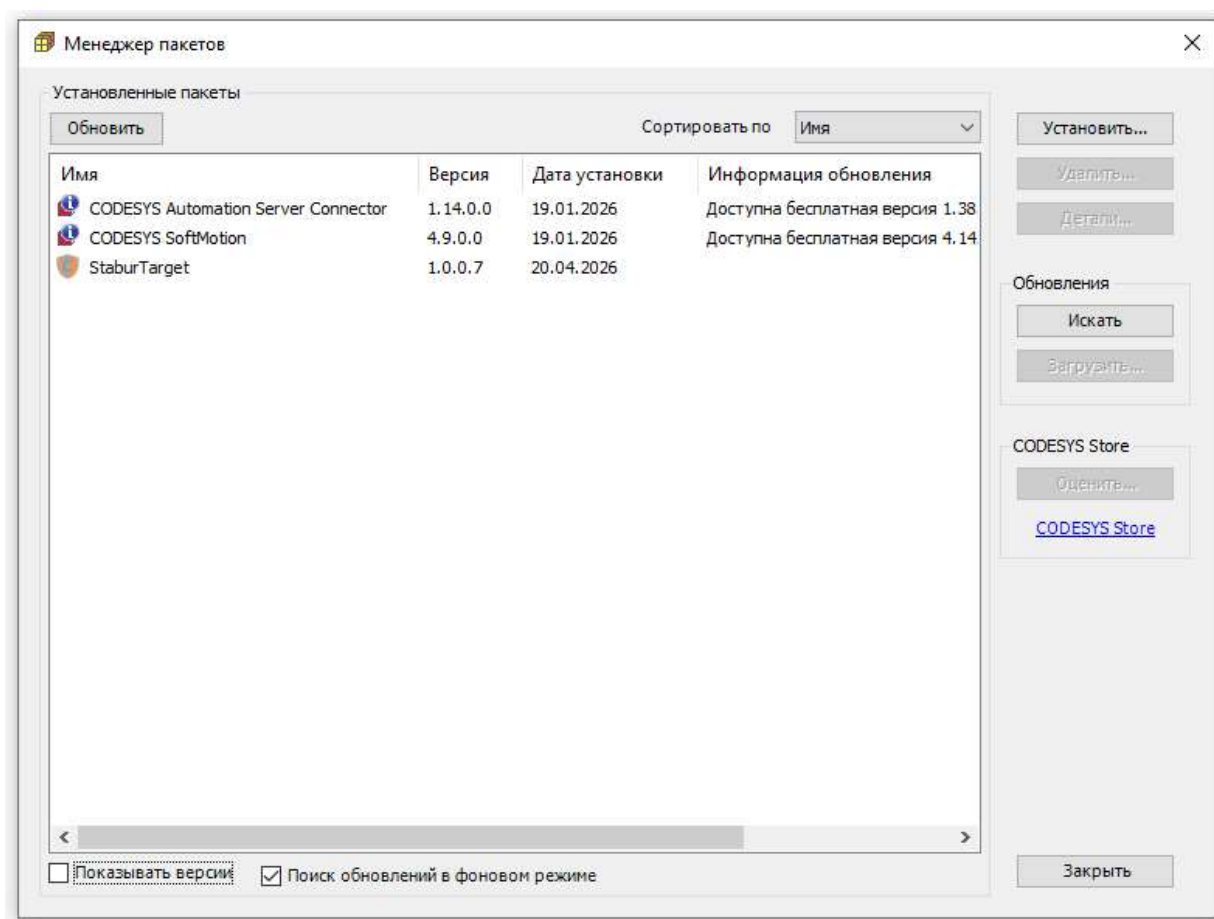


Рисунок 2.4 – Менеджер пакетов: StaburTarget

3. Создайте тестовый проект: **Файл** → **Новый проект** → **Стандартный проект**. В выпадающем списке устройств должны появиться модели СТАБУР (например, **PLC STABUR**).

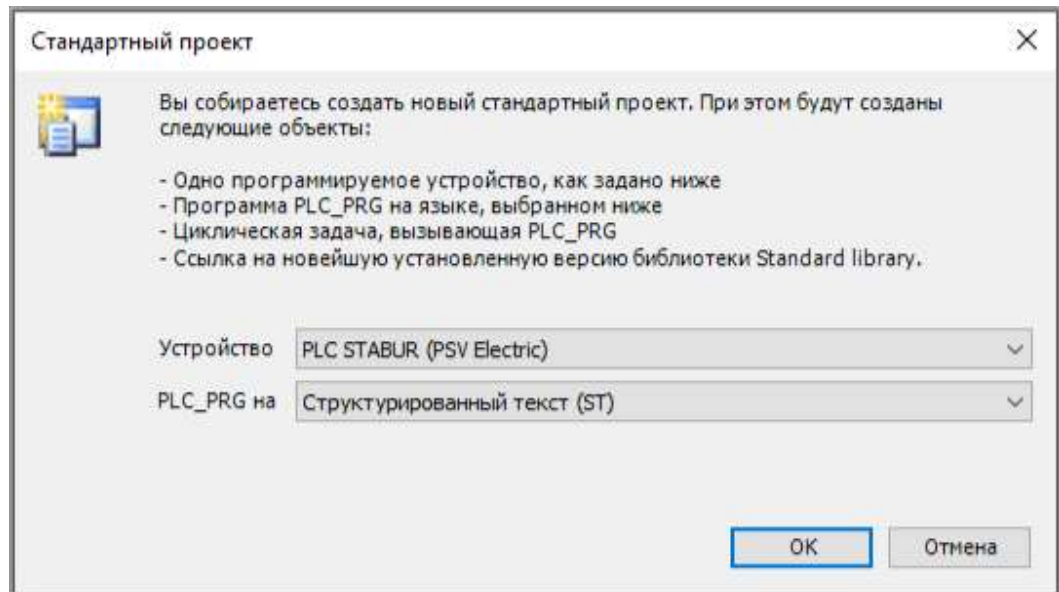


Рисунок 2.5 – Окно создания проекта: в списке устройств видно PLC STABUR

Если устройства СТАБУР в списке отсутствуют:

- убедитесь, что CODESYS был перезапущен после установки пакета;
- проверьте, не установлен ли пакет от другого производителя поверх – удалите лишние через менеджер пакетов;
- повторите установку от имени администратора.

## 3 Быстрый старт

В этом разделе мы создадим проект, подключимся к прибору и загрузим в него первую программу. В качестве примера используется ПЛК СТАБУР 07 (7", одна корзина модулей).

### 3.1 Создание проекта

1. Запустите CODESYS.
2. Выберите **Файл** → **Новый проект** (File → New Project).
3. В окне шаблонов выберите **Стандартный проект** (Standard Project). Укажите имя проекта и каталог для сохранения.

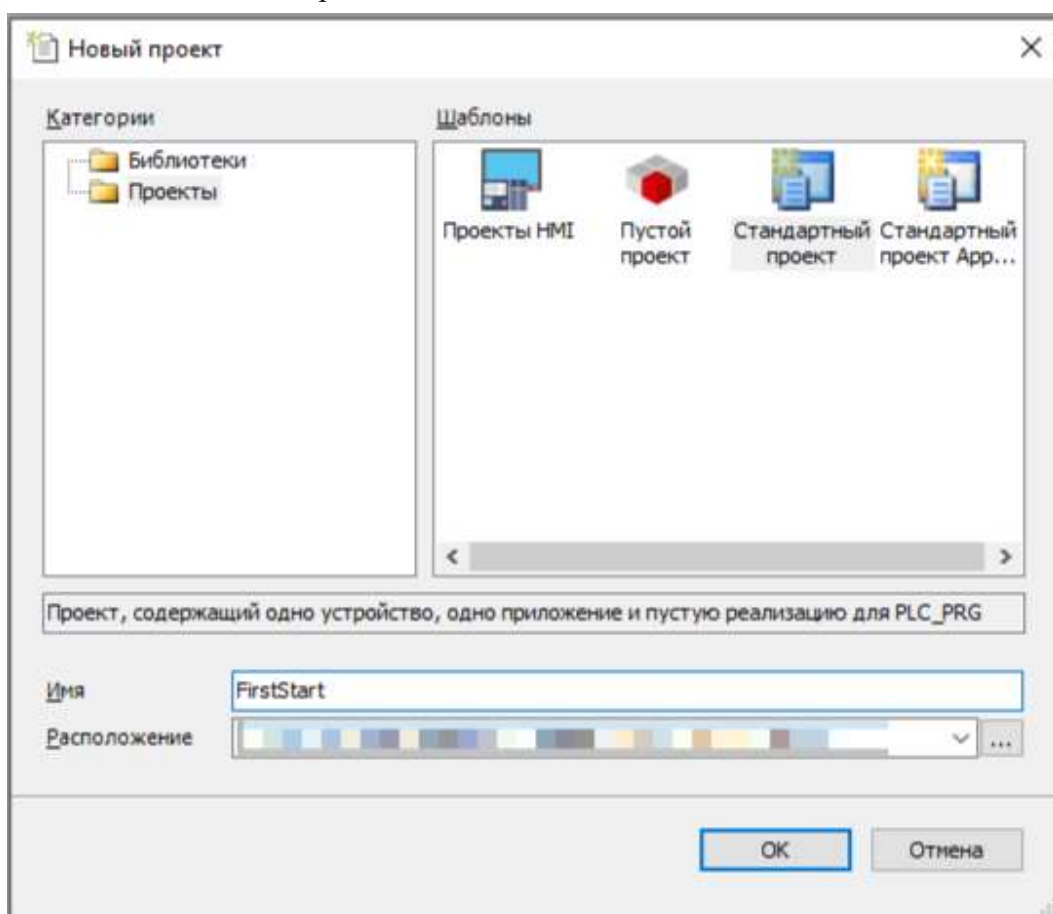


Рисунок 3.1 – Создание нового проекта: выбор шаблона «Стандартный проект»

**! Важно.** Путь к проекту не должен содержать кириллицу и не должен быть слишком длинным. Например: C:\Projects\Stabur\_Test.

4. В следующем окне выберите устройство – **PLC STABUR** (базовый вариант для старта проекта на устройствах СТАБУР с любой диагональю).

5. Выберите язык программирования для главной программы. Рекомендуем **Structured Text (ST)** – это универсальный текстовый язык, удобный для большинства задач. При необходимости язык любой программы можно изменить позже.
6. Нажмите **ОК**. Проект создан.

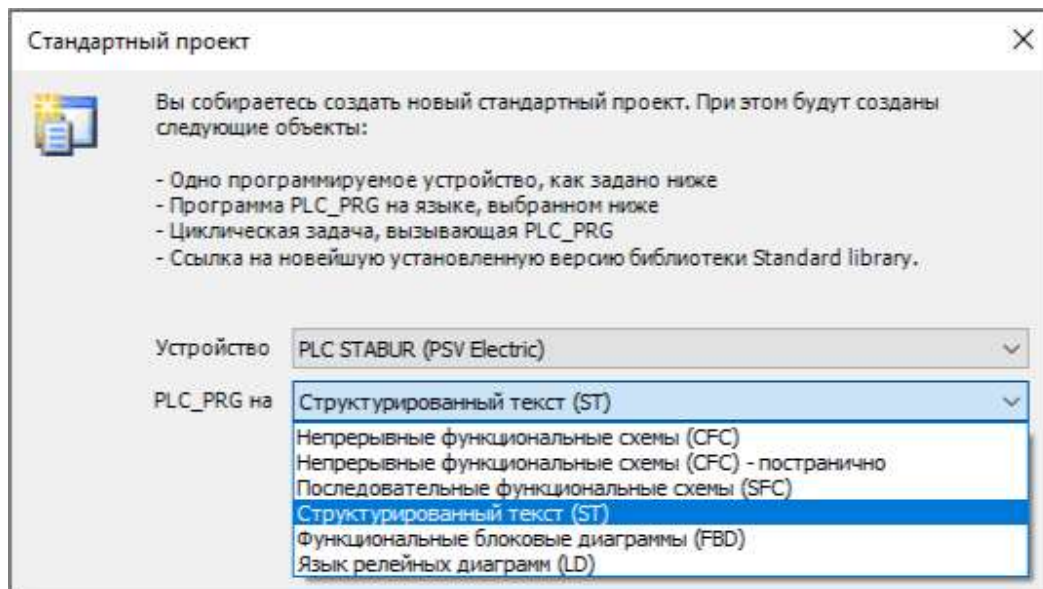


Рисунок 3.2 – Настройка шаблона: выбор устройства PLC STABUR и языка ST

После создания проекта в дереве устройств автоматически появятся:

- корневой узел контроллера (**Device**);
- системные устройства: **Retain** (энергонезависимая память), **LEDS** (управление индикаторами);
- узел **Application** с программой **PLC\_PRG** и **Конфигурация задач**.

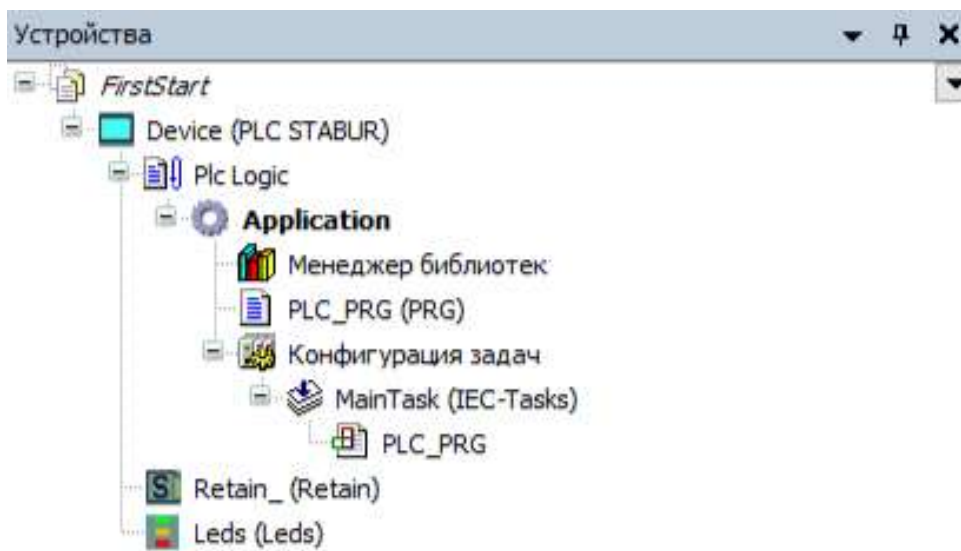


Рисунок 3.3 – Дерево устройств нового проекта: Device, Retain, LEDS, Application, PLC\_PRG

## 3.2 Подключение по mini-USB (RNDIS)

Для первичного подключения используйте порт **mini-USB OTG**: он есть на всех устройствах СТАБУР. На некоторых новых приборах порт может быть расположен на задней панели. При подключении USB-кабелем прибор определяется в Windows как сетевой адаптер RNDIS.

### Сетевые настройки интерфейса RNDIS на приборе (заводские):

Таблица 3.2 – Параметры подключения по mini-USB (RNDIS)

| Параметр        | Значение              |
|-----------------|-----------------------|
| IP-адрес (usb0) | 10.0.0.1              |
| Маска подсети   | 255.255.255.252 (/30) |

Порядок подключения:

1. Включите прибор и **дождитесь полной загрузки** (на экране отображается рабочий интерфейс или заставка ПО).
2. Подключите кабель mini-USB из комплекта поставки к порту OTG на приборе и к USB-порту ПК.
3. Windows автоматически установит драйвер RNDIS и в «Сетевых подключениях» появится новый сетевой адаптер.
4. Проверьте связь: в терминале ПК выполните ping 10.0.0.1 (логин – root без пароля).

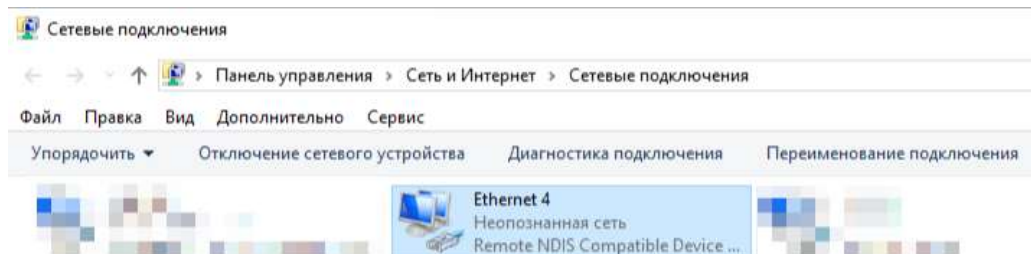
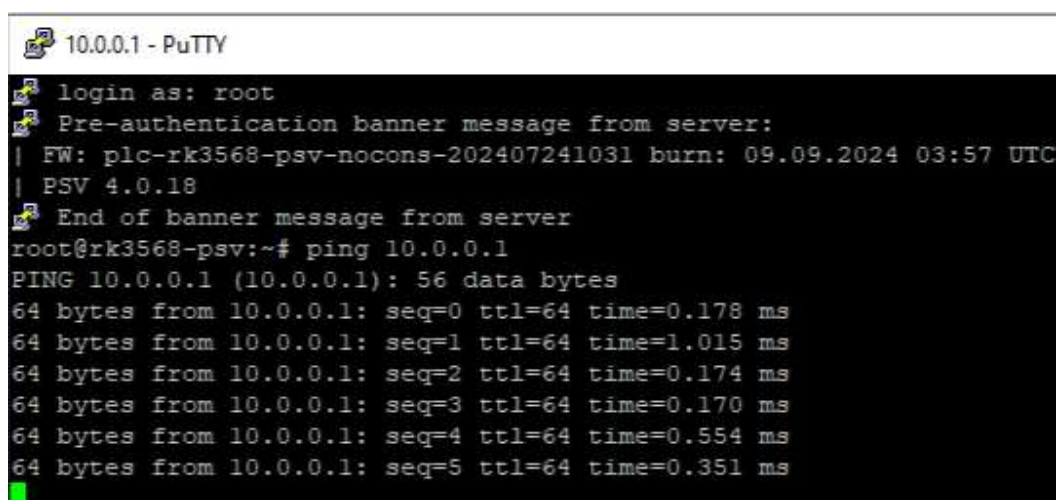


Рисунок 3.4 – Сетевые подключения Windows: новый адаптер RNDIS после подключения прибора



```
10.0.0.1 - PuTTY
login as: root
Pre-authentication banner message from server:
| FW: plc-rk3568-psv-nocons-202407241031 burn: 09.09.2024 03:57 UTC
| PSV 4.0.18
End of banner message from server
root@rk3568-psv:~# ping 10.0.0.1
PING 10.0.0.1 (10.0.0.1): 56 data bytes
64 bytes from 10.0.0.1: seq=0 ttl=64 time=0.178 ms
64 bytes from 10.0.0.1: seq=1 ttl=64 time=1.015 ms
64 bytes from 10.0.0.1: seq=2 ttl=64 time=0.174 ms
64 bytes from 10.0.0.1: seq=3 ttl=64 time=0.170 ms
64 bytes from 10.0.0.1: seq=4 ttl=64 time=0.554 ms
64 bytes from 10.0.0.1: seq=5 ttl=64 time=0.351 ms
```

Рисунок 3.5 – Проверка связи через PuTTY

5. Если ping 10.0.0.1 не проходит, назначьте адаптеру RNDIS статический адрес из подсети прибора:
- IP: 10.0.0.2
  - Маска: 255.255.255.252

**! Внимание.** Интерфейсы mini-USB и RS-232 **не имеют гальванической развязки**. Все подключаемое к прибору оборудование (компьютер, коммутатор и др.), имеющее клеммы заземления, **должно быть надёжно заземлено**. Несоблюдение этого требования может привести к повреждению прибора.

**Если Windows не распознаёт устройство:** проверьте, что прибор полностью загрузился. Попробуйте другой USB-кабель. Откройте «Диспетчер устройств» – если устройство отображается с восклицательным знаком, установите драйвер вручную (обычно стандартный драйвер RNDIS Windows 10/11).

### 3.3 Подключение по Ethernet

Также подключение к прибору можно выполнять не только по mini-USB, но и по Ethernet. Подключение выполняется **кабелем Ethernet (RJ45)** – напрямую между ПК и прибором или через коммутатор; конкретный тип кабеля определяется вашей схемой подключения.

#### Сетевые настройки прибора по умолчанию:

Интерфейс Ethernet (eth0) по умолчанию настроен на получение IP-адреса **по DHCP**. Если в сети есть DHCP-сервер, прибор получит адрес автоматически.

Узнать текущий IP-адрес прибора можно:

- через **системную утилиту**;
- из консоли прибора командой `ifconfig`.

**Если DHCP-сервера в сети нет** (например, при прямом соединении ПК–прибор кабелем):

Задайте прибору статический IP-адрес. Для этого отредактируйте на приборе файл /etc/systemd/network/10-eth.network:

```
[Network]
DHCP=no
Address=192.168.0.10/24
Gateway=192.168.0.1
```

После этого настройте сетевой адаптер ПК в той же подсети, например:

- IP: 192.168.0.2
- Маска: 255.255.255.0

**Проверка связи:** откройте командную строку на ПК и выполните ping 192.168.0.10 – должны идти ответы.

На разъёме Ethernet прибора (RJ45 на модуле ETH232 или встроенный порт) есть индикаторы:

- **Зелёный (Link)** – физическое соединение установлено; мигание – идёт обмен данными.
- **Жёлтый** – светится при скорости 100 Мбит/с.

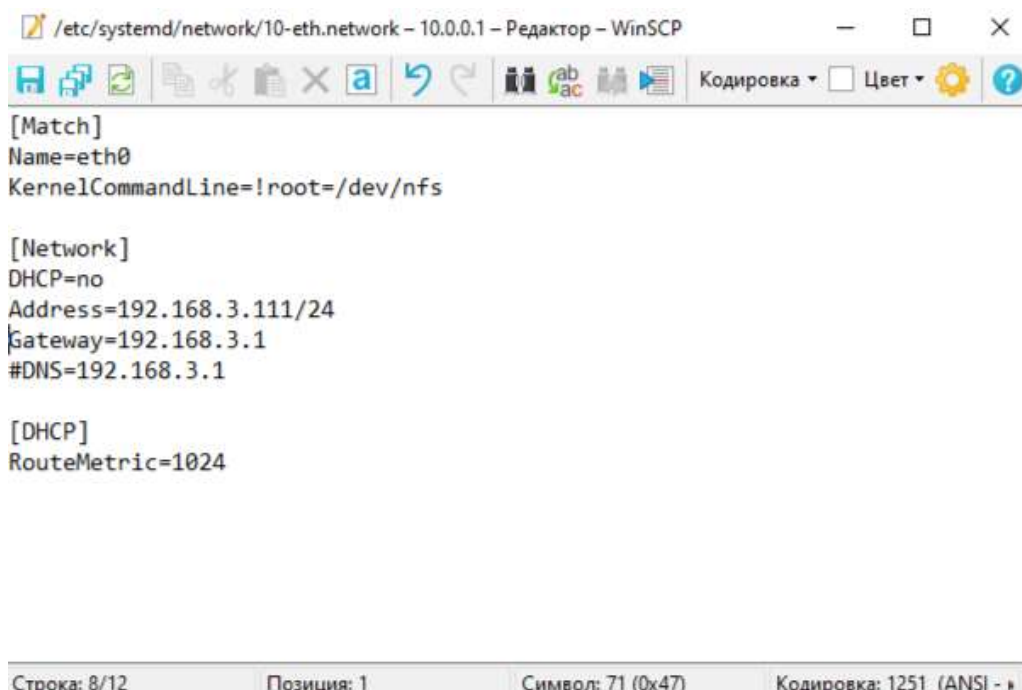


Рисунок 3.6 – Пример назначения IP-адреса

### 3.4 Установка связи в CODESYS

После физического подключения прибора к ПК (по Ethernet или USB) настройте связь в среде CODESYS:

1. В дереве устройств дважды щёлкните по корневому узлу контроллера (**Device (PLC STABUR)**). Откроется редактор устройства.
2. Перейдите на вкладку **Установки соединения** (Communication Settings).

Рисунок 3.6 – Вкладка «Установки соединения» узла Device

3. Нажмите кнопку **Сканировать сеть** (Scan network). CODESYS через шлюз Gateway опросит доступные сети и выведет список найденных приборов.

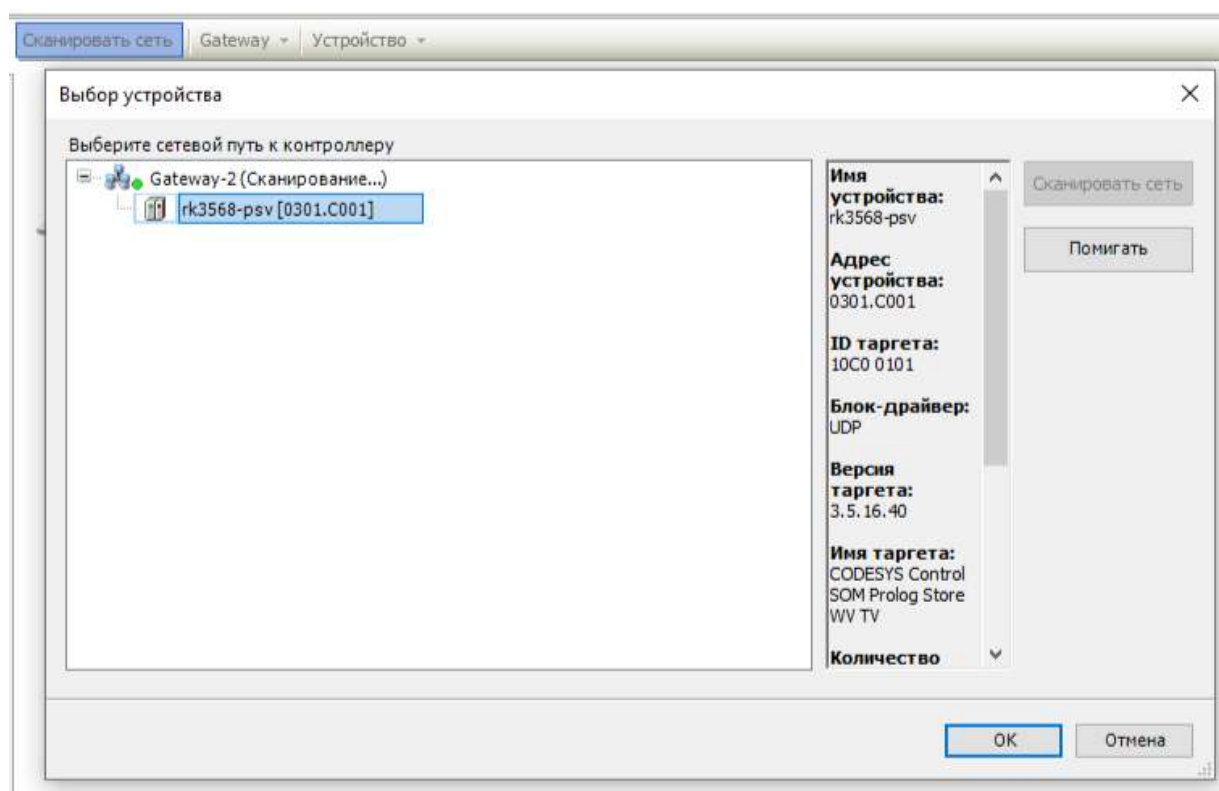


Рисунок 3.7 – Результат сканирования сети: найден прибор ПЛК СТАБУР с IP-адресом

4. Выберите нужный прибор в списке и нажмите **ОК**.

**Если прибор не найден при сканировании:** - Убедитесь, что ping до прибора проходит. - Проверьте, что служба **CODESYS Gateway** запущена (иконка в области уведомлений Windows; если отсутствует – запустите через «Пуск» или «Службы»). - Временно отключите брандмауэр Windows для проверки. - Если ПК имеет несколько сетевых адаптеров, убедитесь, что Gateway работает на нужном интерфейсе.

### 3.5 Компиляция, загрузка и запуск

Теперь загрузим в прибор первую программу.

1. Откройте программу **PLC\_PRG** (двойной щелчок в дереве устройств). В редакторе ST добавьте простую тестовую логику, например:

```
PROGRAM PLC_PRG
```

```
VAR
```

```
    bTestLamp : BOOL;
```

```
    iCounter : INT;
```

```
END_VAR
```

```
bTestLamp := NOT bTestLamp; // Инвертируем каждый цикл
```

```
iCounter := iCounter + 1; // Наращиваем счётчик
```

2. Выполните компиляцию: **Сборка** → **Компилировать** (Build → Build, или клавиша **F11**). В панели сообщений внизу должно быть 0 ошибок, 0 предупреждений.

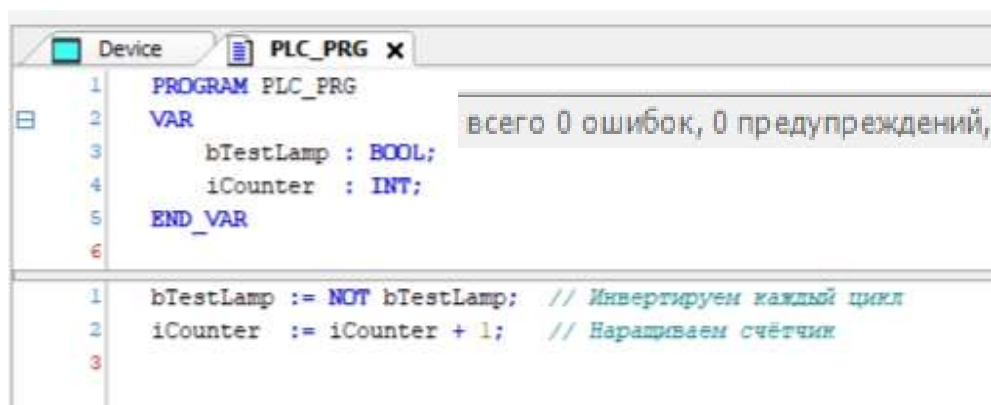


Рисунок 3.8 – Результат компиляции: 0 ошибок

3. Подключитесь к прибору: **Логин** (Online → Login, или **Alt+F8**). При первом подключении CODESYS предложит загрузить приложение – подтвердите.

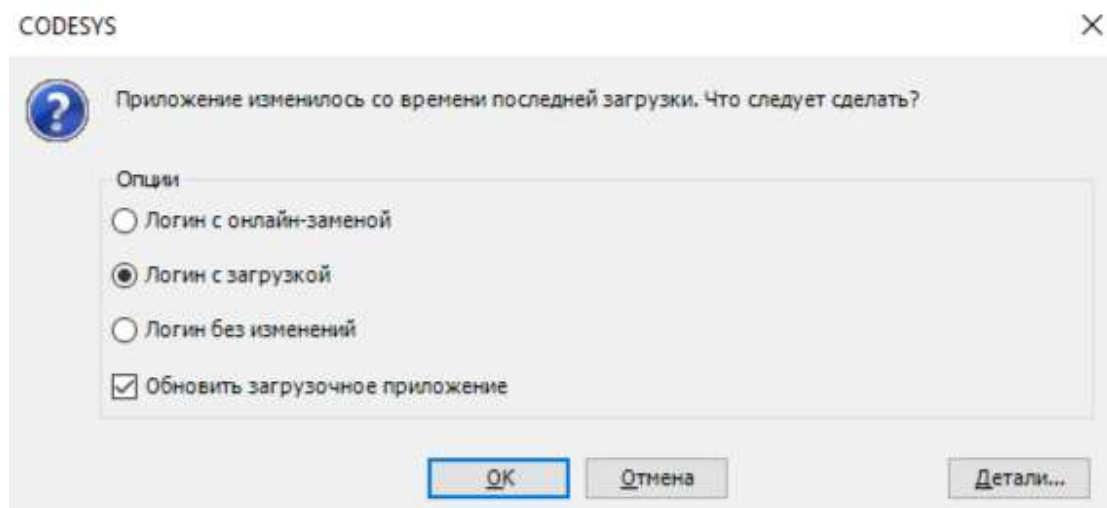


Рисунок 3.9 – Диалог подтверждения загрузки приложения при Login

4. Запустите приложение: **Отладка** → **Старт** (Online → Start, или **F5**). В дереве устройств у узла Application появится зелёный индикатор **Run**.

5. Проверьте работу: переменные bTestLamp и iCounter должны изменяться в редакторе в режиме онлайн. Значение iCounter будет расти с каждым циклом ПЛК.

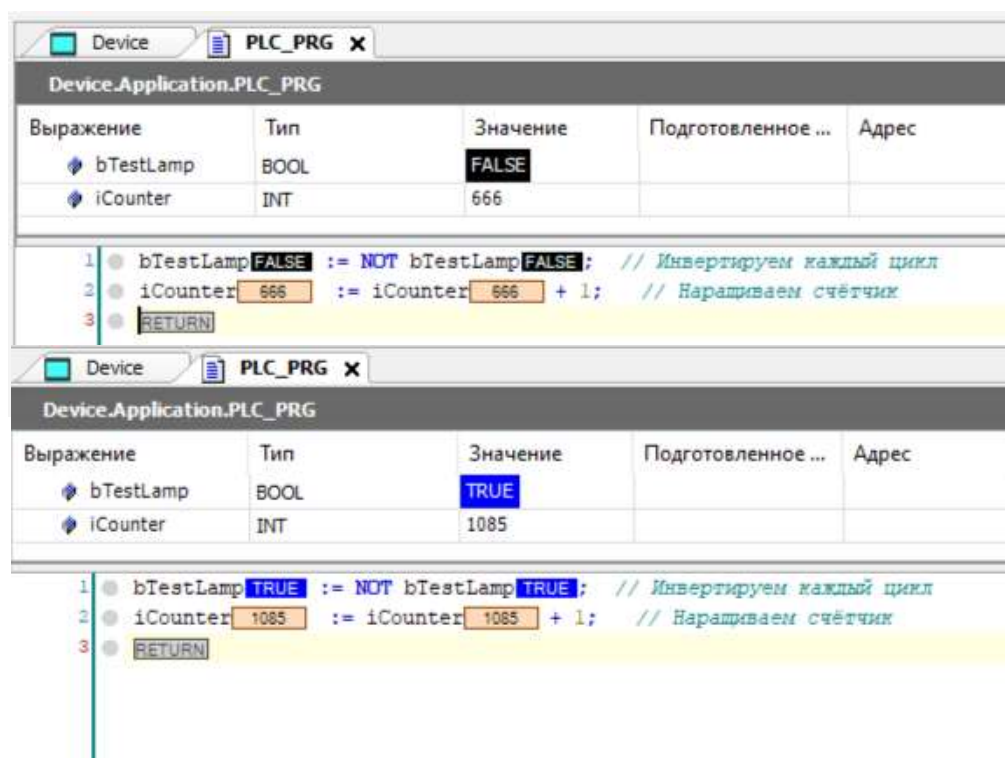


Рисунок 3.10 – Редактор PLC\_PRG в режиме онлайн: значения переменных обновляются

**Рекомендация.** Для ПЛК СТАБУР не рекомендуется использовать прямую АТ-адресацию (%IW, %QW). Работайте с переменными и привязывайте их к каналам устройств через I/O-маппинг (вкладка каналов узла модуля в дереве устройств). Это соответствует концепции CODESYS V3 и обеспечивает переносимость проекта.

## 4 Настройки устройства и компонентов в дереве проекта

При создании проекта CODESYS автоматически формирует дерево устройств. Каждый узел дерева – это аппаратный компонент прибора или программный объект. При двойном щелчке по узлу открывается его редактор с набором вкладок, зависящим от типа узла.

Ниже рассмотрены основные узлы, которые присутствуют в типовом проекте для ПЛК СТАБУР.

### 4.1 Узел контроллера (Device)

Корневой узел дерева – это сам контроллер. При двойном щелчке в вашей конфигурации отображаются следующие вкладки:

Таблица 4.1 – Вкладки узла контроллера (Device)

| Вкладка                                | Назначение                                                                              |
|----------------------------------------|-----------------------------------------------------------------------------------------|
| Установки соединения                   | Настройка связи CODESYS с прибором: сканирование сети, выбор активного пути подключения |
| Приложения                             | Список приложений runtime (если функционал поддержан текущей конфигурацией)             |
| Резервное копирование и восстановление | Экран резервирования/восстановления (доступность зависит от прав и режима)              |
| Файлы                                  | Просмотр файловых областей устройства (если доступ разрешён)                            |
| Журнал                                 | Журнал событий и сообщений runtime                                                      |
| Установки ПЛК                          | Параметры контроллера и обмена                                                          |
| Оболочка ПЛК                           | Параметры оболочки устройства (если используются в проекте)                             |
| Пользователи и группы                  | Настройка пользователей и групп (если включена модель пользователей)                    |
| Права доступа                          | Настройка прав доступа                                                                  |
| Символьные права                       | Права к символам/переменным                                                             |
| Размещение задачи                      | Параметры выполнения задач runtime                                                      |
| Состояние                              | Состояние устройства и runtime                                                          |
| Информация                             | Справочная информация по устройству и интерфейсам                                       |

**! Важно.** Не все вкладки обязательно активны и применимы в каждом таргете/режиме подключения. Для «первого старта» используйте прежде всего три вкладки: **Установки соединения, Установки ПЛК, Информация.**

Практически значимые действия в этом руководстве:

- **Установки соединения** – наладка связи с прибором (поиск устройства, выбор активного подключения).
- **Установки ПЛК** – для обмена рекомендуется параметр «**Всегда обновлять переменные**» = «**Включено 2 (всегда в задаче цикла шины)**».
- **Информация** – просмотр номеров COM-портов (RS-485, RS-232, GSM) и путей накопителей (eMMC, SD, USB).

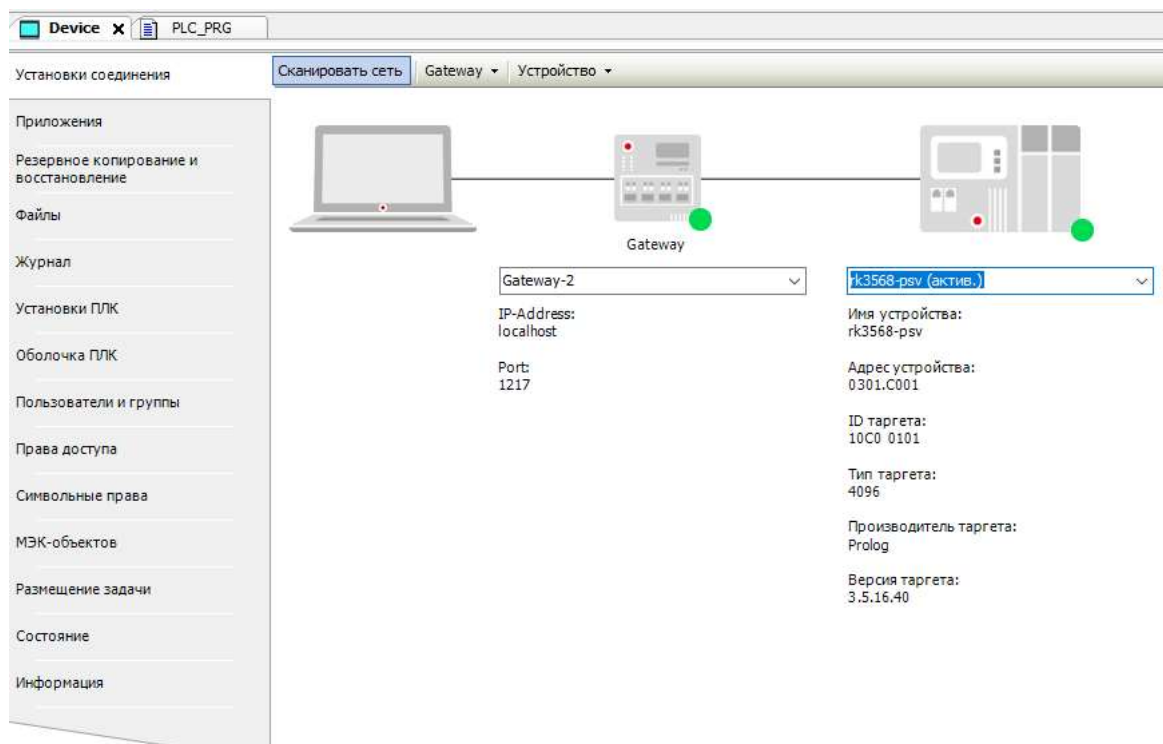


Рисунок 4.1 – Узел Device: перечень вкладок (Установки соединения, Приложения, Резервное копирование и восстановление, Файлы, Журнал, Установки ПЛК, Оболочка ПЛК, Пользователи и группы, Права доступа, Символьные права, Размещение задачи, Состояние, Информация)

## 4.2 Энергонезависимая память (Retain)

Узел **Retain** автоматически добавляется в дерево при создании проекта. Он отвечает за сохранение Retain-переменных при отключении питания.

По нашей сборке у узла Retain доступны следующие вкладки:

- **Конфигурация** – рабочие параметры сохранения Retain.
- **Соотнесение входов/выходов** – служебные каналы (Счетчик записей, Код состояния, Запись).
- **Информация** – общие сведения об объекте.

### 4.2.1 Конфигурация Retain

На вкладке **Конфигурация** задаются параметры хранения и периодичность записи Retain-данных:

Таблица 4.2.1 – Параметры хранения энергонезависимых данных Retain

| Место хранения | Когда использовать                                                              | Ограничения                                                                                                      |
|----------------|---------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------|
| <b>eMMC</b>    | Большое количество переменных, которые редко меняются (уставки, конфигурация)   | Минимальный период сохранения – <b>5 секунд</b> .<br>Данные хранятся в файловой системе на встроенном накопителе |
| <b>FRAM</b>    | Небольшое количество переменных, которые меняются часто (счётчики, интеграторы) | Максимальный объём – <b>8 КБ</b> .<br>Минимальный период – <b>100 мс</b>                                         |

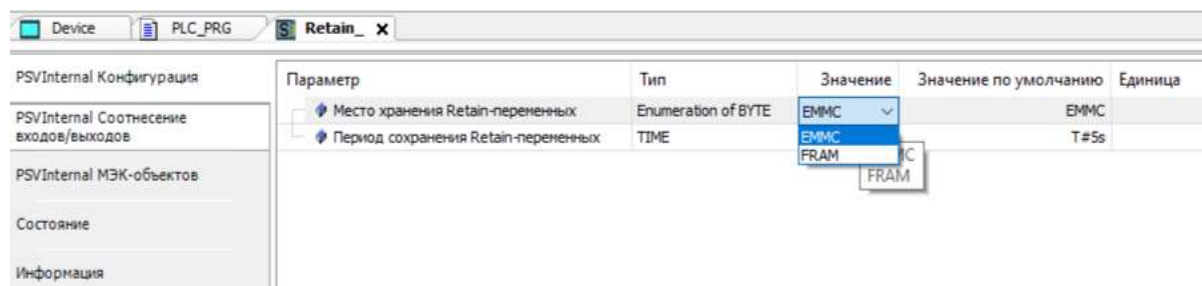


Рисунок 4.2 – Узел Retain, вкладка «Конфигурация»: выбор параметров хранения Retain и режима/периода записи

### 4.2.2 Каналы Retain (вкладка «Соотнесение входов/выходов»)

На вкладке **Соотнесение входов/выходов** доступны диагностические каналы:

Таблица 4.2.2 – Диагностические каналы узла Retain

| Канал           | Тип   | Описание                                                                            |
|-----------------|-------|-------------------------------------------------------------------------------------|
| Счётчик записей | DWORD | Количество выполненных сохранений                                                   |
| Код состояния   | UDINT | 0 – ОК; 1 – область памяти не определена; 36 – файл повреждён; 50 – файл недоступен |
| Запись          | BOOL  | TRUE – принудительная запись без ожидания периода                                   |

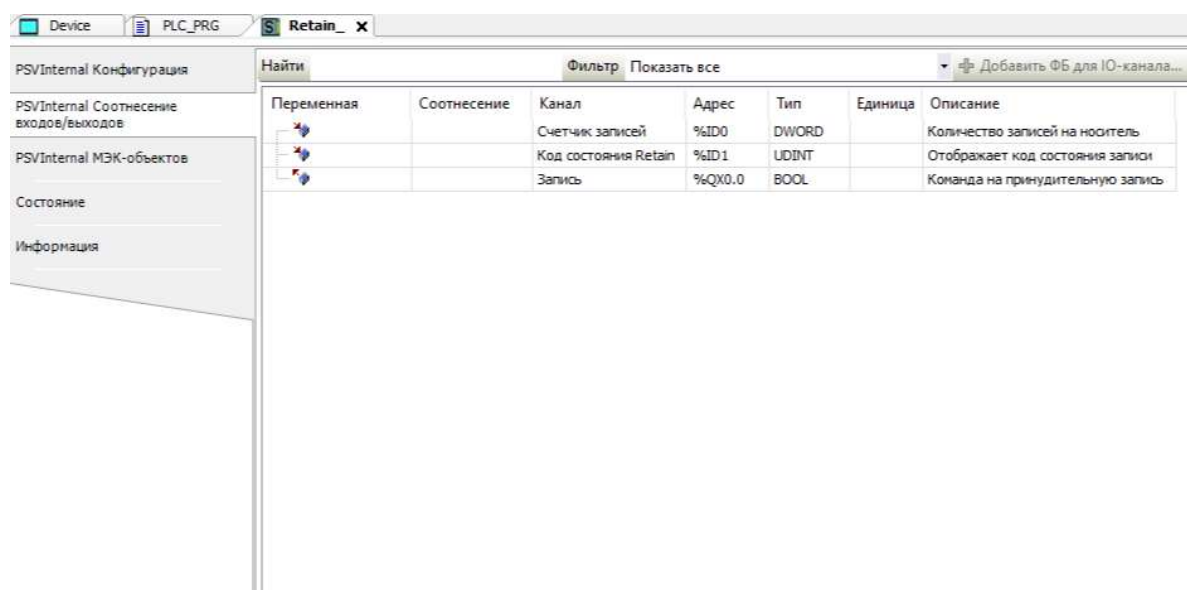


Рисунок 4.3 – Узел Retain, вкладка «Соотнесение входов/выходов»: каналы «Счетчик записей», «Код состояния», «Запись»

### 4.2.3 Служебные вкладки Retain

В узле Retain также присутствуют служебные вкладки (например, «МЭК-объекты» и «Информация»). Их используют по задаче.

## 4.3 Управление индикаторами (LEDS)

Узел **LEDS** также добавляется автоматически. Через него из программы можно управлять светодиодами на лицевой панели, яркостью экрана и зуммером.

На вкладке «Соотнесение входов/выходов» доступны:

Таблица 4.3 – Диагностические каналы узла LEDS

| Канал               | Тип  | Описание                                                   |
|---------------------|------|------------------------------------------------------------|
| Светодиод А Зелёный | BOOL | FALSE – выключен, TRUE – включен                           |
| Светодиод А Красный | BOOL | FALSE – выключен, TRUE – включен                           |
| Светодиод В Зелёный | BOOL | Аналогично                                                 |
| Светодиод В Красный | BOOL | Аналогично                                                 |
| Светодиод С Зелёный | BOOL | Аналогично                                                 |
| Светодиод С Красный | BOOL | Аналогично                                                 |
| Яркость экрана      | BYTE | 0 – подсветка выключена; 1 – минимальная; 8 – максимальная |
| Звуковой сигнал     | BOOL | FALSE – выключен, TRUE –                                   |

| Канал | Тип | Описание |
|-------|-----|----------|
|       |     | включен  |

**Примечание.** При одновременном включении зелёной и красной составляющей одного индикатора результирующий цвет – **оранжевый**.

| Переменная | Соотнесение | Канал             | Адрес  | Тип   | Единица | Описание                      |
|------------|-------------|-------------------|--------|-------|---------|-------------------------------|
|            |             | Leds              | %QB1   |       |         | Leds on front plc panel       |
|            |             | LED_A Green       | %QX1.0 | BOOL  |         | [0: Off, 1: On]               |
|            |             | LED_A Red         | %QX1.1 | BOOL  |         | [0: Off, 1: On]               |
|            |             | LED_B Green       | %QX1.2 | BOOL  |         | [0: Off, 1: On]               |
|            |             | LED_B Red         | %QX1.3 | BOOL  |         | [0: Off, 1: On]               |
|            |             | LED_C Green       | %QX1.4 | BOOL  |         | [0: Off, 1: On]               |
|            |             | LED_C Red         | %QX1.5 | BOOL  |         | [0: Off, 1: On]               |
|            |             | Screen brightness | %QB2   | USINT |         | Level [0:Off, 1: Min, 8: Max] |
|            |             | Bit0              | %QX2.0 | BOOL  |         | Level [0:Off, 1: Min, 8: Max] |
|            |             | Bit1              | %QX2.1 | BOOL  |         | Level [0:Off, 1: Min, 8: Max] |
|            |             | Bit2              | %QX2.2 | BOOL  |         | Level [0:Off, 1: Min, 8: Max] |
|            |             | Bit3              | %QX2.3 | BOOL  |         | Level [0:Off, 1: Min, 8: Max] |
|            |             | Bit4              | %QX2.4 | BOOL  |         | Level [0:Off, 1: Min, 8: Max] |
|            |             | Bit5              | %QX2.5 | BOOL  |         | Level [0:Off, 1: Min, 8: Max] |
|            |             | Bit6              | %QX2.6 | BOOL  |         | Level [0:Off, 1: Min, 8: Max] |
|            |             | Bit7              | %QX2.7 | BOOL  |         | Level [0:Off, 1: Min, 8: Max] |
|            |             | Beeper            | %QX3.0 | BOOL  |         | [0: Выкл, 1: Вкл]             |

Рисунок 4.4 – Узел LEDS, вкладка «Соотнесение входов/выходов»: светодиоды, яркость экрана, зуммер

## 4.4 Корзина модулей и добавление модулей ввода-вывода

Если ваш прибор оснащён модулями ввода-вывода (аналоговыми, дискретными, температурными), необходимо добавить в дерево устройств **корзину модулей** и затем – сами модули.

### Добавление корзины:

1. В дереве устройств нажмите правой кнопкой мыши на узел **Device** и выберите **Добавить устройство** (Add Device).
2. В каталоге выберите **Modules Basket** (Корзина модулей ввода-вывода) и нажмите **Добавить устройство**.

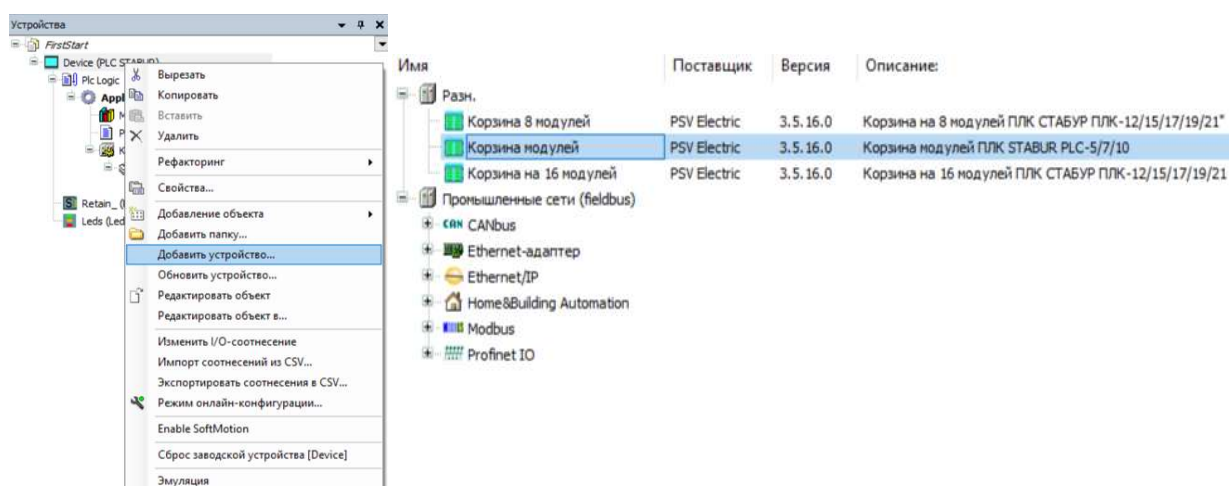


Рисунок 4.5 – Добавление устройства «Modules Basket» в дерево проекта

После добавления корзины в дерево появятся слоты 1–8 (для одной корзины).

### Добавление модуля в слот:

1. Выберите нужный слот в дереве (номер слота должен соответствовать физическому расположению модуля в приборе).
2. Правый клик → **Подключить устройство**.
3. Выберите тип модуля – он должен точно соответствовать модулю, физически установленному в этот слот прибора.

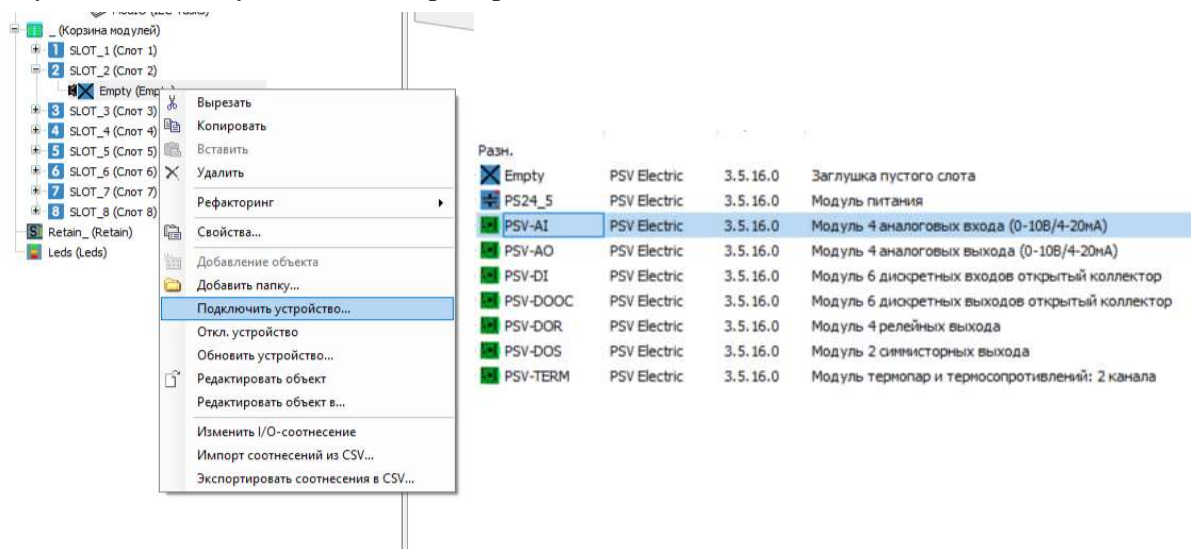


Рисунок 4.6 – Добавление модуля PSV-AI в слот 2 корзины

Доступные модули в таргет-файле:

Таблица 4.4 – Модули ввода-вывода СТАБУР в CODESYS

| Обозначение   | Назначение                  | Каналов |
|---------------|-----------------------------|---------|
| <b>PSV-AI</b> | Аналоговые входы (ток 4–20) | 4       |

| Обозначение     | Назначение                                                        | Каналов |
|-----------------|-------------------------------------------------------------------|---------|
|                 | мА / напряжение 0–10 В)                                           |         |
| <b>PSV-AO</b>   | Аналоговые выходы (ток 4–20 мА / напряжение 0–10 В)               | 2       |
| <b>PSV-TERM</b> | Измерение температуры (термопары, термосопротивления)             | 2       |
| <b>PSV-DI</b>   | Дискретные входы (сухой контакт); каналы 1–2 – счётные до 200 кГц | 6       |
| <b>PSV-DOOC</b> | Дискретные выходы, тип «открытый коллектор» (24 В, до 200 мА)     | 6       |
| <b>PSV-DOR</b>  | Дискретные выходы, тип «реле» (~220 В, до 5 А)                    | 4       |
| <b>PSV-DOS</b>  | Дискретные выходы, тип «симистор» (~220 В, до 2 А)                | 2       |

**! Внимание.** Несоответствие типа модуля в проекте и физически установленного модуля приведёт к диагностической ошибке **ERR\_ID\_MISMATCH** (код 33). Всегда проверяйте соответствие дерева устройств фактической конфигурации прибора.

Каждый модуль после добавления имеет одинаковый набор из 5 вкладок:

- **Конфигурация** – параметры модуля (номер слота, тип входа/выхода, фильтры);
- **Соотнесение входов/выходов** – привязка каналов модуля к переменным программы;
- **МЭК-объектов** – служебная вкладка объекта;
- **Состояние** – статус модуля и диагностическая информация;
- **Информация** – сведения о модуле и его параметрах.

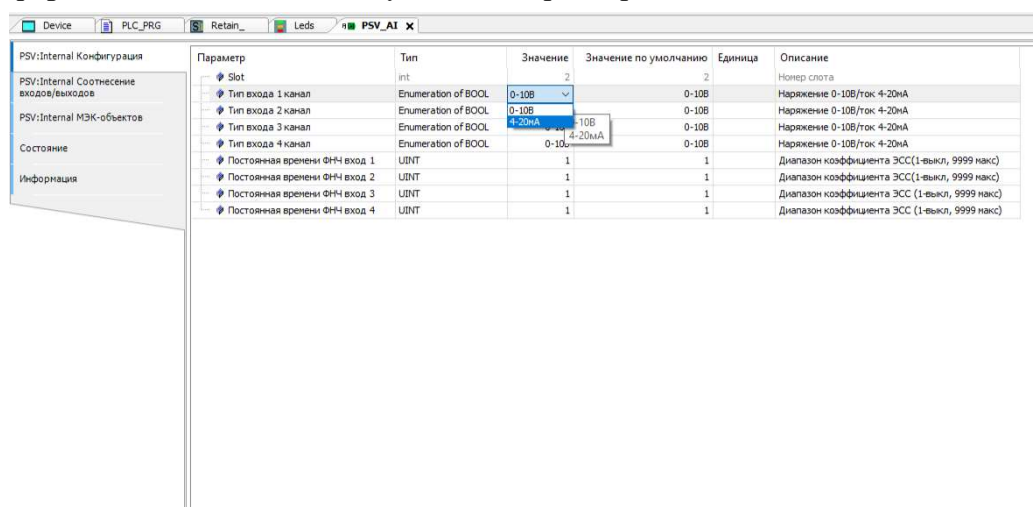


Рисунок 4.7 – Конфигурация модуля PSV-AI: выбор типа входа (ток/напряжение) для каждого канала

| Переменная | Соотнесение | Канал                | Адрес  | Тип   | Единица | Описание                                   |
|------------|-------------|----------------------|--------|-------|---------|--------------------------------------------|
|            |             | Диагностика          | %ID0   |       |         |                                            |
|            |             | Состояние ошибки     | %IX0.0 | BOOL  |         | TRUE: Ошибка модуля, FALSE - Ошибка нет    |
|            |             | Результат выполнения | %ID1   | DWORD |         | ERR_OK(0) - успешно                        |
|            |             | Число запросов       | %ID2   | DWORD |         | Число запросов к модулю                    |
|            |             | Число ответов        | %ID3   | DWORD |         | Число ответов от модуля                    |
|            |             | Число неотвеченных   | %ID4   | DWORD |         | Число неотвеченных от модуля               |
|            |             | Код типа модуля      | %IB20  | BYTE  |         | [1-DOR/2-DI/3-AI/4-DOOC/5-DOS/6-AQ/7-TERM] |
|            |             | Версия ПО            | %ID6   | REAL  |         | Формат Ver.Rev                             |
|            |             | Аппаратная версия    | %ID7   | REAL  |         | Формат Ver.Rev                             |
|            |             | 1 канал              | %ID8   | REAL  |         | Значение 1 аналогового входа               |
|            |             | 2 канал              | %ID9   | REAL  |         | Значение 2 аналогового входа               |
|            |             | 3 канал              | %ID10  | REAL  |         | Значение 3 аналогового входа               |
|            |             | 4 канал              | %ID11  | REAL  |         | Значение 4 аналогового входа               |

Рисунок 4.8 – Каналы I/O mapping модуля PSV-AI: привязка переменных к аналоговым входам

## 4.5 Узел Application и задачи

Узел **Application** содержит всю прикладную логику проекта:

- **PLC\_PRG** – главная программа, создаваемая при инициализации проекта;
- **Task Configuration** – конфигурация задач (циклов вызова программ);
- дополнительные программы, функциональные блоки, GVL, визуализации – добавляются пользователем по мере развития проекта.

**Конфигурация задач** определяет, с каким периодом и приоритетом вызывается каждая программа. По умолчанию создаётся одна задача **MainTask**, которая вызывает PLC\_PRG с интервалом 20 мс.

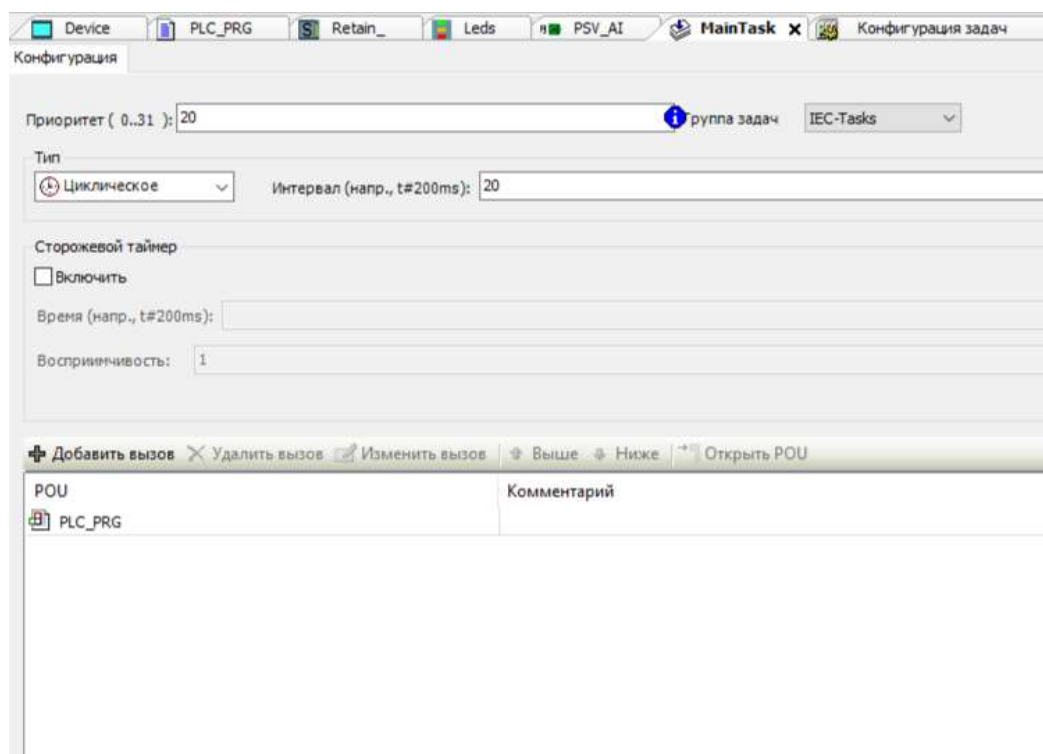


Рисунок 4.9 – Task Configuration: MainTask с интервалом 20 мс и вызовом PLC\_PRG

При необходимости можно добавить дополнительные задачи – например, с более длинным циклом для «медленной» обработки (архивация, вычисление трендов) или с коротким циклом для быстрого регулирования. Приоритет задачи определяет, какая из них выполнится первой при конкуренции за ресурсы процессора.

**Совет.** Не устанавливайте интервал задачи слишком малым. Если время выполнения программы превышает интервал цикла, runtime выдаст предупреждение о превышении времени цикла (watchdog). Для типовых задач автоматизации рекомендуемый диапазон – от 10 до 50 мс.

## 5 Интерфейс CODESYS

В этом разделе рассмотрена структура главного окна CODESYS и основные элементы, с которыми работает программист.

### 5.1 Главное окно среды

Рабочее пространство CODESYS состоит из нескольких областей:

- **Строка меню** (сверху): Файл, Правка, Вид, Проект, Сборка, Отладка, Инструменты, Окно, Справка.
- **Панель инструментов** (под меню): кнопки быстрого доступа к компиляции, подключению, поиску.
- **Дерево устройств / Обозреватель проекта** (слева): иерархическая структура проекта.
- **Центральная область**: вкладки открытых редакторов (программы, визуализации, конфигурации устройств).
- **Панель сообщений** (внизу): результаты компиляции, ошибки, предупреждения, поиск.

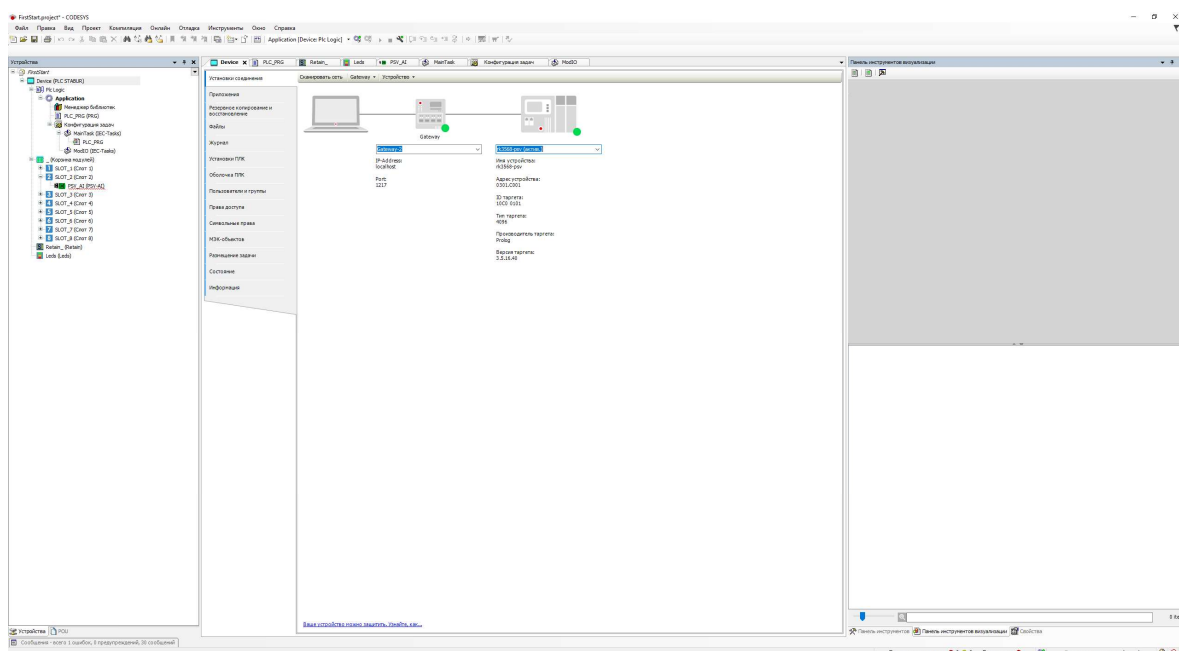


Рисунок 5.1 – Главное окно CODESYS: меню, панель инструментов, дерево устройств, редактор PLC\_PRG, панель сообщений

### 5.2 Дерево устройств

Дерево устройств – центральный навигационный элемент. Оно отображает:

- **Device** (корневой контроллер) и вложенные аппаратные компоненты: Retain, LEDS, корзина модулей, Ethernet, Modbus и т.д.;
- **Application** – программная часть проекта: POU (программы, функциональные блоки, функции), GVL (глобальные списки переменных), DUT (пользовательские типы данных), визуализации, задачи.

Контекстное меню (правый клик) позволяет добавлять устройства, программы, визуализации, подключаться к прибору.

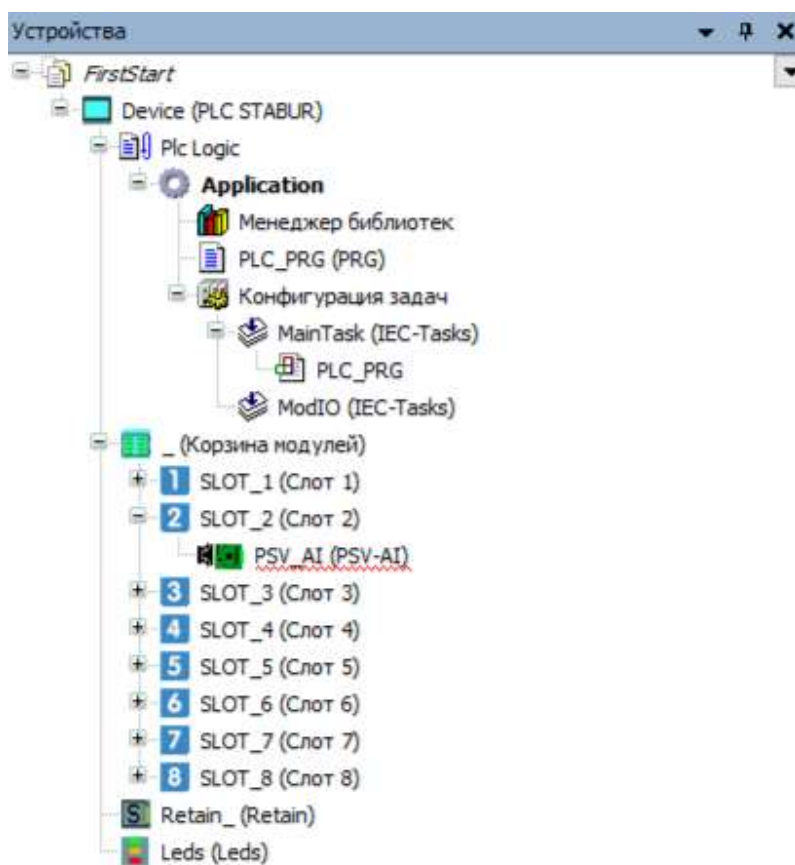


Рисунок 5.2 – Развёрнутое дерево устройств типового проекта: Device → Retain, LEDS, Modules Basket (со слотами), Application → PLC\_PRG, Task Configuration

### 5.3 Редакторы программ

При двойном щелчке по программному объекту (POU) в дереве открывается редактор соответствующего языка:

- **ST** – текстовый редактор с подсветкой синтаксиса, автодополнением, нумерацией строк;
- **LD** – графический редактор релейных схем;
- **FBD / CFC** – графический редактор блочных диаграмм;
- **SFC** – редактор последовательных функциональных диаграмм.

Верхняя часть редактора (для текстовых языков) содержит область **объявления переменных** (VAR ... END\_VAR), нижняя – тело программы.

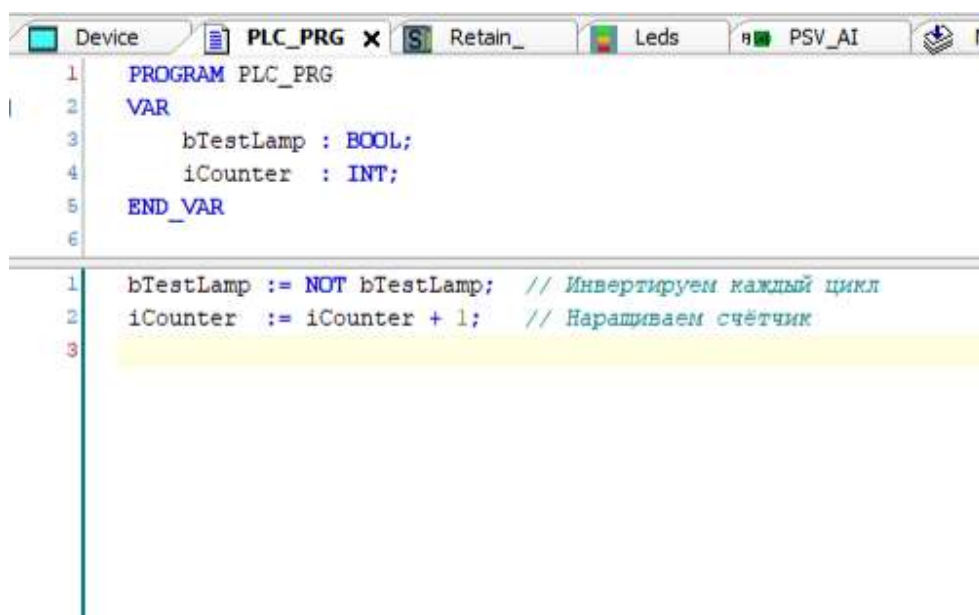


Рисунок 5.3 – Редактор ST: область объявления переменных (сверху), тело программы (снизу)

## 5.4 Панель сообщений

Панель сообщений (внизу окна) отображает:

- **Результаты компиляции:** ошибки (красные), предупреждения (жёлтые), информационные сообщения. Двойной щелчок по ошибке переводит курсор к месту проблемы в коде.
- **Вывод сборки:** ход компиляции и линковки.
- **Поиск:** результаты глобального поиска по проекту.

**Совет.** Используйте глобальный поиск (**Правка → Найти и заменить** или **Ctrl+N**) для переименования переменных и проверки использования символов по всему проекту.

## 5.5 Меню и горячие клавиши

Основные операции с клавиатуры:

Таблица 5.5 – Горячие клавиши CODESYS для первого запуска

| Действие   | Клавиши        | Меню                    |
|------------|----------------|-------------------------|
| Компиляция | <b>F11</b>     | Компиляция → Компиляция |
| Логин      | <b>Alt+F8</b>  | Онлайн → Логин          |
| Отключение | <b>Ctrl+F8</b> | Онлайн → Отключение     |

| Действие             | Клавиши               | Меню                           |
|----------------------|-----------------------|--------------------------------|
| Старт                | <b>F5</b>             | Отладка → Старт                |
| Стоп                 | <b>Shift+F8</b>       | Отладка → Стоп                 |
| Один цикл            | <b>Ctrl+F5</b>        | Отладка → Один цикл            |
| Записать значения    | <b>Ctrl+F7</b>        | Отладка → Записать значения    |
| Фиксировать значения | <b>F7</b>             | Отладка → Фиксировать значения |
| Освободить значения  | <b>Alt+F7</b>         | Отладка → Освободить значения  |
| Сохранить проект     | <b>Ctrl+S</b>         | Файл → Сохранить               |
| Новый проект         | <b>Ctrl+N</b>         | Файл → Новый проект            |
| Открыть проект       | <b>Ctrl+O</b>         | Файл → Открыть проект          |
| Выход                | <b>Alt+F4</b>         | Файл → Выход                   |
| Отменить             | <b>Ctrl+Z</b>         | Правка → Отменить              |
| Повторить            | <b>Ctrl+Y</b>         | Правка → Повторить             |
| Вырезать             | <b>Ctrl+X</b>         | Правка → Вырезать              |
| Копировать           | <b>Ctrl+Ins</b>       | Правка → Копировать            |
| Вставить             | <b>Ctrl+V</b>         | Правка → Вставить              |
| Удалить              | <b>Del</b>            | Правка → Удалить               |
| Выбрать всё          | <b>Ctrl+A</b>         | Правка → Выбрать всё           |
| Ассистент ввода      | <b>F2</b>             | Правка → Ассистент ввода       |
| Автодобавление       | <b>Shift+F2</b>       | Правка → Автодобавление        |
| Следующее сообщение  | <b>F4</b>             | Правка → Следующее сообщение   |
| Предыдущее сообщение | <b>Shift+F4</b>       | Правка → Предыдущее сообщение  |
| Вид «Устройства»     | <b>Alt+0</b>          | Вид → Устройства               |
| Вид «POU»            | <b>Alt+1</b>          | Вид → POU                      |
| Вид «Модули»         | <b>Alt+2</b>          | Вид → Модули                   |
| Вид «Сообщения»      | <b>Alt+3</b>          | Вид → Сообщения                |
| Во весь экран        | <b>Ctrl+Shift+F12</b> | Вид → Во весь экран            |
| Следующий редактор   | <b>Ctrl+F6</b>        | Окно → Следующий редактор      |
| Предыдущий редактор  | <b>Ctrl+Shift+F6</b>  | Окно → Предыдущий редактор     |
| Следующая панель     | <b>F6</b>             | Окно → Следующая панель        |
| Предыдущая панель    | <b>Shift+F6</b>       | Окно → Предыдущая панель       |
| Содержание справки   | <b>Ctrl+Shift+F1</b>  | Справка → Содержание           |
| Указатель справки    | <b>Ctrl+Shift+F2</b>  | Справка → Указатель            |

**Примечание.** При необходимости горячие клавиши можно добавить или изменить вручную: **Инструменты** → **Настройки** → **Клавиатура**.

## 6 Кратко о программировании

### 6.1 Языки МЭК 61131-3

CODESYS поддерживает все языки стандарта МЭК 61131-3. В рамках одного проекта языки можно комбинировать: каждый POU (программа, функциональный блок, функция) создаётся на выбранном языке независимо от остальных.

#### **ST – Structured Text (Структурированный текст)**

Текстовый язык, синтаксически близкий к Pascal. Поддерживает условные конструкции (IF, CASE), циклы (FOR, WHILE, REPEAT), вызовы функций и функциональных блоков. Наиболее универсальный и выразительный язык: подходит для вычислений, обработки данных, реализации сложных алгоритмов. Рекомендуется как основной при изучении CODESYS.

#### **LD – Ladder Diagram (Релейно-контактная схема)**

Графический язык, визуально напоминающий релейные схемы. Слева и справа – «шины питания», между ними – горизонтальные цепочки контактов и катушек. Интуитивно понятен инженерам с опытом работы с релейной автоматикой. Хорошо подходит для дискретной логики: включение/выключение по условиям, блокировки, защиты.

#### **FBD – Function Block Diagram (Диаграмма функциональных блоков)**

Графический язык. Программа строится из блоков (стандартных и пользовательских), соединённых линиями данных. Выход одного блока подаётся на вход другого. Удобен для аналоговых цепочек обработки сигналов, ПИД-регулирования, фильтрации. Схема выполнения – слева направо, сверху вниз.

#### **SFC – Sequential Function Chart (Последовательная функциональная схема)**

Графический язык для описания пошаговых процессов. Программа состоит из шагов (Step) и переходов (Transition): из одного шага в другой переходят при выполнении условия. В каждом шаге можно вызывать действия на любом другом языке. Применяется для рецептурного управления, автоматов состояний, технологических последовательностей (открыть клапан → ждать → закрыть).

#### **CFC – Continuous Function Chart (Непрерывная функциональная схема)**

Расширение FBD: блоки размещаются свободно на листе, соединения проводятся вручную в произвольном порядке. В отличие от FBD, порядок выполнения задаётся вручную (нумерация вызовов). Удобен для построения регуляторов и схем управления, которые наглядно отражают структуру системы.

## IL – Instruction List (Список инструкций)

Низкоуровневый текстовый язык, напоминающий язык ассемблера. В CODESYS V3 считается устаревшим и практически не используется в новых проектах.

В одном проекте языки комбинируются: например, основная логика на ST, управление технологической последовательностью – на SFC, визуальная отладка входов – на LD. Выбор языка производится при создании POU.

## 6.2 Типы данных

Стандарт МЭК 61131-3 определяет набор элементарных и составных типов данных. CODESYS реализует их в полном объёме.

### 6.2.1 Логический тип

Таблица 6.2.1 – Логический тип данных

| Тип  | Размер | Диапазон     | Применение                                |
|------|--------|--------------|-------------------------------------------|
| BOOL | 1 бит  | FALSE / TRUE | Флаги, состояния, дискретные входы/выходы |

Литералы: TRUE, FALSE. Используется для дискретных каналов модулей I/O, условий, флагов состояния.

### 6.2.2 Целые типы

Знаковые типы хранят положительные и отрицательные значения, беззнаковые – только неотрицательные.

Таблица 6.2.2 – Целые типы данных

| Тип          | Размер | Диапазон (знаковый)                        | Беззнаковый аналог | Диапазон (беззнаковый)     |
|--------------|--------|--------------------------------------------|--------------------|----------------------------|
| SINT / USINT | 8 бит  | –128 ... 127                               | USINT              | 0 ... 255                  |
| INT / UINT   | 16 бит | –32 768 ... 32 767                         | UINT               | 0 ... 65 535               |
| DINT / UDINT | 32 бит | –2 147 483 648 ...<br>2 147 483 647        | UDINT              | 0 ... 4 294 967 295        |
| LINT / ULINT | 64 бит | –9·10 <sup>18</sup> ... 9·10 <sup>18</sup> | ULINT              | 0 ... 1,8·10 <sup>19</sup> |

На практике INT и DINT покрывают большинство задач. DINT рекомендуется там, где возможно переполнение INT (счётчики, накопленные значения).

### 6.2.3 Байтовые типы

Таблица 6.2.3 – Байтовые типы данных

| Тип   | Размер | Применение                                |
|-------|--------|-------------------------------------------|
| BYTE  | 8 бит  | Маски, флаги, байт данных                 |
| WORD  | 16 бит | Регистры Modbus, управляющие слова        |
| DWORD | 32 бит | Расширенные регистры, битовые операции    |
| LWORD | 64 бит | Редко используется, для специальных задач |

Байтовые типы не имеют знака и предназначены прежде всего для побитовых операций (AND, OR, XOR, SHL, SHR). Часто применяются при работе с протоколами передачи данных.

### 6.2.4 Вещественные типы

Таблица 6.2.4 – Вещественные типы данных

| Тип   | Размер | Точность          | Применение                                |
|-------|--------|-------------------|-------------------------------------------|
| REAL  | 32 бит | ~7 значимых цифр  | Аналоговые сигналы, температура, давление |
| LREAL | 64 бит | ~15 значимых цифр | Высокоточные вычисления                   |

REAL – стандартный тип для аналоговых каналов модулей I/O. Литерал: 3.14, 1.0E-3. При делении двух целых чисел результат необходимо явно привести к REAL, иначе дробная часть будет отброшена.

### 6.2.5 Строки

Таблица 6.2.5 – Строковые типы данных

| Тип     | Описание                                                                                |
|---------|-----------------------------------------------------------------------------------------|
| STRING  | Строка символов ASCII. По умолчанию – до 80 символов. Размер задаётся явно: STRING(255) |
| WSTRING | Строка в кодировке Unicode (двухбайтовые символы)                                       |

Строки используются в визуализации, формировании сообщений, работе с файловой системой. Длина STRING влияет на объём занимаемой памяти.

## 6.2.6 Временные типы

Таблица 6.2.6 – Временные типы данных

| Тип                | Формат литерала                | Применение                                |
|--------------------|--------------------------------|-------------------------------------------|
| TIME               | T#1h30m, T#500ms,<br>T#1s200ms | Задержки, таймеры,<br>интервалы           |
| LTIME              | LTIME#1h                       | Расширенный TIME (64 бит,<br>наносекунды) |
| DATE               | D#2024-01-15                   | Дата (без времени)                        |
| TIME_OF_DAY (TOD)  | TOD#08:30:00                   | Время суток                               |
| DATE_AND_TIME (DT) | DT#2024-01-15-08:30:00         | Дата и время вместе                       |

Тип TIME широко используется совместно со стандартными функциональными блоками таймеров: TON (задержка включения), TOF (задержка отключения), TP (импульс).

## 6.2.7 Составные типы

**ARRAY** – массив элементов одного типа. Индексация с произвольного числа:

```
VAR
  arData : ARRAY[0..9] OF REAL;
END_VAR
```

**STRUCT** – структура из полей разных типов. Объявляется в отдельном DUT (Data Unit Type):

```
TYPE ST_Sensor :
STRUCT
  rValue : REAL;
  bValid : BOOL;
  sName : STRING;
END_STRUCT
END_TYPE
```

**ENUM** – перечисление именованных констант. Делает код читаемым: вместо числа 2 используется, например, eModeAuto.

**UNION** – объединение: несколько полей занимают одну и ту же область памяти. Применяется для разбора байтовых структур (например, чтения отдельных байт из DWORD).

**Совет.** Для аналоговых каналов модулей I/O используйте тип REAL, для дискретных – BOOL. Тип привязываемой переменной должен точно совпадать с типом канала в разделе «Соотнесение входов/выходов», иначе компилятор выдаст ошибку несоответствия типов.

## 6.3 Переменные и области видимости

Любая переменная в CODESYS должна быть объявлена до использования. Объявление состоит из имени, типа и необязательного начального значения:

```
имя_переменной : ТИП := начальное_значение;
```

Переменные объявляются в специальных блоках в начале каждого POU (программы, функционального блока, функции). Блок определяет область видимости – откуда переменная доступна.

### 6.3.1 Блоки объявления переменных

**VAR ... END\_VAR** – локальные переменные

Доступны только внутри того POU, в котором объявлены. Значения сохраняются между вызовами программы или функционального блока (статические). Это основной блок для рабочих переменных алгоритма.

```
VAR
  bRunning : BOOL := FALSE;
  rSetpoint : REAL := 50.0;
  iCounter : INT;
END_VAR
```

**VAR\_INPUT ... END\_VAR** – входные параметры

Используется только в функциональных блоках (FUNCTION\_BLOCK) и функциях (FUNCTION). Значения передаются извне при каждом вызове. Внутри блока их изменить нельзя.

```
VAR_INPUT
  rActualValue : REAL;
  bEnable : BOOL;
END_VAR
```

**VAR\_OUTPUT ... END\_VAR** – выходные параметры

Результаты работы функционального блока, доступные вызывающей программе после выполнения. Внутри блока записываются, снаружи – только читаются.

```
VAR_OUTPUT
  bAlarm : BOOL;
  rResult : REAL;
END_VAR
```

**VAR\_IN\_OUT ... END\_VAR** – двунаправленные параметры

Передаются по ссылке: изменение внутри блока изменяет оригинальную переменную вызывающего кода. Применяется, когда функциональный блок должен модифицировать переданные данные (например, обработать массив на месте).

**VAR\_TEMP ... END\_VAR** – временные переменные

Существуют только во время одного вызова. Значение не сохраняется между итерациями задачи. Используются как промежуточные буферы для вычислений внутри одного цикла.

**VAR\_GLOBAL ... END\_VAR** – глобальные переменные

Объявляются не в POU, а в отдельном объекте **GVL (Global Variable List)**: правый клик по **Application** → **Добавить объект** → **Список глобальных переменных**. Доступны из любого POU в рамках приложения. При обращении к глобальной переменной используется префикс имени GVL:

```
GVL.bSystemReady := TRUE;
```

**Примечание.** GVL – стандартный и удобный способ хранения переменных, доступных из нескольких программ или из визуализации. В практике CODESYS-разработки активное использование GVL считается нормой.

### 6.3.2 Модификаторы переменных

Модификаторы указываются перед типом переменной и изменяют её поведение.

#### CONSTANT

Значение задаётся при объявлении и не может быть изменено в процессе выполнения программы. Используется для физических констант, коэффициентов, фиксированных порогов:

```
VAR CONSTANT
  rPi    : REAL := 3.14159;
  iMaxItems : INT := 100;
END_VAR
```

#### RETAIN

Значение сохраняется в энергонезависимой памяти при потере питания и восстанавливается при следующем запуске. Работает только при правильно настроенном узле Retain в дереве проекта (см. раздел 4.2).

При загрузке нового проекта в контроллер RETAIN-переменные **сбрасываются** в начальные значения. Это удобно в проектах, где структура переменных часто меняется в процессе разработки.

Типичное применение: счётчики наработки, текущие уставки оператора, положение исполнительных механизмов на момент остановки.

```
VAR RETAIN
    iProductionCount : DINT;
    rLastSetpoint    : REAL;
END_VAR
```

## PERSISTENT

Значение сохраняется как при потере питания, так и при загрузке нового проекта в контроллер – при условии, что структура переменных (имена, типы) не изменилась. Если структура изменилась, значения также будут сброшены.

Применяется для параметров, которые должны сохраняться при доработках проекта: калибровочные коэффициенты, накопленные данные, долгосрочные пользовательские настройки.

Сравнение двух модификаторов:

Таблица 6.3.2 – Сравнение модификаторов RETAIN и PERSISTENT

| Характеристика                         | RETAIN | PERSISTENT                        |
|----------------------------------------|--------|-----------------------------------|
| Сохранение при отключении питания      | Да     | Да                                |
| Сохранение при загрузке нового проекта | Нет    | Да (если структура не изменилась) |
| Удобство при частых изменениях проекта | Выше   | Ниже                              |

## 6.3.3 Именованние переменных

В CODESYS принято использовать венгерскую нотацию: префикс указывает на тип переменной, что делает код читаемым без необходимости смотреть объявление.

Таблица 6.3.3 – Префиксы именования переменных

| Префикс | Тип    | Пример       |
|---------|--------|--------------|
| b       | BOOL   | bPumpRunning |
| i       | INT    | iCounter     |
| di      | DINT   | diTotalCount |
| r       | REAL   | rTemperature |
| s       | STRING | sDeviceName  |
| t       | TIME   | tDelay       |

| Префикс | Тип                           | Пример   |
|---------|-------------------------------|----------|
| ar      | ARRAY                         | arBuffer |
| st      | STRUCT                        | stSensor |
| e       | ENUM                          | eMode    |
| fb      | FUNCTION_BLOCK<br>(экземпляр) | fbTimer  |

Соглашение не является обязательным, однако его соблюдение значительно упрощает работу с проектом в команде и при его последующем сопровождении.

## 6.4 Программные объекты (POU)

POU (Program Organization Unit, компонент организации программы) – основная структурная единица кода в CODESYS. Каждый POU имеет раздел объявления переменных и раздел тела программы. При создании нового POU выбирается его тип и язык реализации.

Для добавления нового POU: правый клик по **Application** → **Добавить объект** → **POU** → задать имя, выбрать тип и язык реализации → **Добавить**.

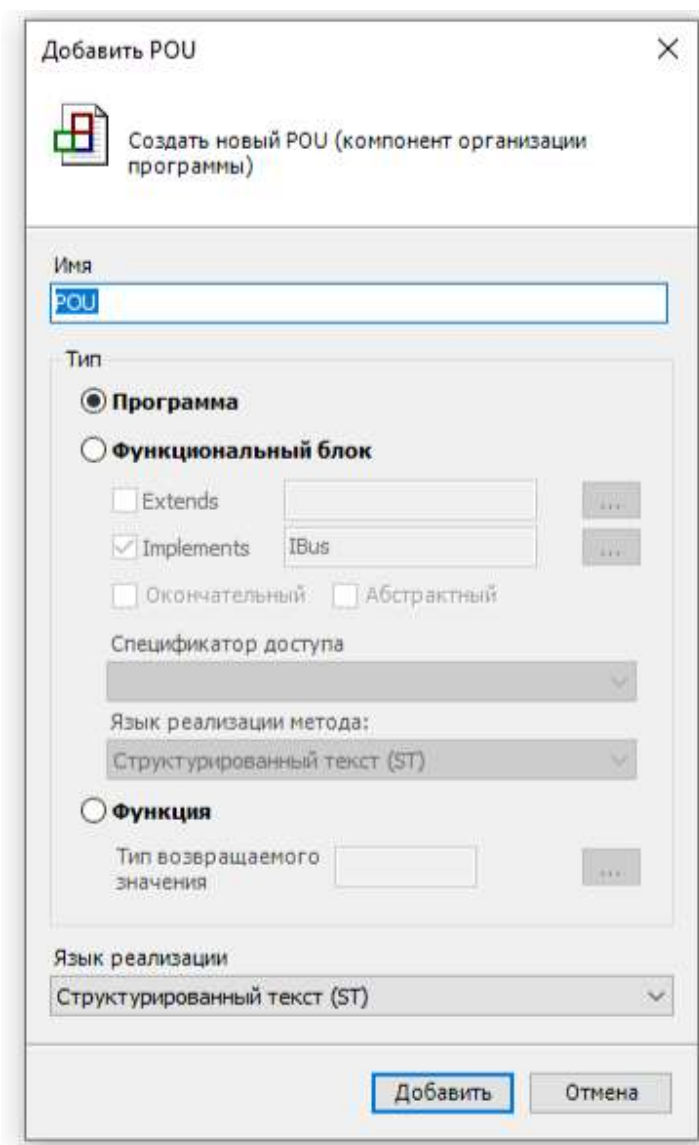


Рисунок 6.1 – Диалог добавления POU: выбор типа (Программа / Функциональный блок / Функция) и языка реализации

#### 6.4.1 PROGRAM – Программа

Программа – верхний уровень логики приложения. Вызывается непосредственно из задачи (Task Configuration) с заданной периодичностью. В большинстве проектов присутствует одна главная программа с именем PLC\_PRG, которую CODESYS создаёт автоматически при создании нового проекта.

Ключевые свойства:

- переменные сохраняют своё значение между циклами (статическое состояние);
- может вызывать функции и экземпляры функциональных блоков;

- существует в единственном экземпляре – создать несколько копий одной программы нельзя;
- не принимает входных параметров и не возвращает значения напрямую – обмен данными ведётся через глобальные переменные (GVL) или через входы/выходы вызываемых FB.

Типичное применение: главная логика управления, диспетчер задач, координация работы отдельных функциональных блоков.

#### 6.4.2 FUNCTION\_BLOCK – Функциональный блок

Функциональный блок (FB) – переиспользуемый компонент с собственным состоянием, входами и выходами. В отличие от программы, FB не вызывается из задачи напрямую: сначала в программе объявляется **экземпляр** FB (переменная типа этого блока), затем экземпляр вызывается в теле программы.

Ключевые свойства:

- каждый экземпляр независим – два экземпляра одного FB хранят разные значения своих внутренних переменных;
- имеет явные входы (VAR\_INPUT) и выходы (VAR\_OUTPUT), значения которых задаются и читаются при вызове;
- сохраняет состояние между вызовами – именно поэтому таймеры, счётчики и регуляторы реализуются как FB.

Пример: стандартный блок TON (таймер с задержкой включения) – это функциональный блок. В одной программе можно объявить несколько его экземпляров, каждый будет отсчитывать своё время независимо.

```
VAR
    fbTimer1 : TON;
    fbTimer2 : TON;
END_VAR
```

Типичное применение: таймеры, счётчики, ПИД-регуляторы, драйверы оборудования, любые алгоритмы, которые нужно использовать несколько раз в разных частях проекта.

#### 6.4.3 FUNCTION – Функция

Функция – RCU, который получает входные параметры и возвращает **одно** результирующее значение. Тип возвращаемого значения указывается при создании. Функция **не имеет состояния**: каждый вызов независим, внутренние переменные не сохраняются между вызовами.

Ключевые свойства:

- вызывается как выражение и возвращает значение непосредственно;

- результат зависит только от входных параметров – одни и те же аргументы всегда дают одинаковый результат;
- не может хранить данные между вызовами (нет статических переменных).

Типичное применение: математические преобразования, масштабирование аналоговых сигналов, пересчёт единиц измерения, форматирование строк.

#### 6.4.4 Сравнение типов POU

Таблица 6.4.4 – Сравнение типов POU

| Характеристика                      | PROGRAM | FUNCTION_BLOCK | FUNCTION           |
|-------------------------------------|---------|----------------|--------------------|
| Сохраняет состояние между вызовами  | Да      | Да             | Нет                |
| Можно создать несколько экземпляров | Нет     | Да             | –                  |
| Принимает входные параметры         | Нет     | Да             | Да                 |
| Возвращает значение                 | Нет     | Через выходы   | Да (одно значение) |
| Вызывается из задачи (Task)         | Да      | Нет            | Нет                |
| Типичный пример                     | PLC_PRG | TON, PID       | Масштабирование    |

#### 6.5 Простой пример на ST

Ниже – учебный пример: программа считывает аналоговый вход, сравнивает с уставкой и управляет дискретным выходом (включение насоса при превышении уровня).

```

PROGRAM PLC_PRG
VAR
    rLevel    : REAL;          // Уровень с модуля AI (привязать через Соотнесение
                               // входов/выходов)
    rSetpoint : REAL := 75.0; // Уставка включения, %
    bPumpOn   : BOOL;         // Команда на насос (привязать к DO через Соотнесение
                               // входов/выходов)
    tonDelay  : TON;          // Задержка включения
END_VAR

// Включение насоса с задержкой 2 секунды после превышения уставки
tonDelay(IN := rLevel > rSetpoint, PT := T#2s);
bPumpOn := tonDelay.Q;

```

**Что здесь происходит:**

1. Переменная `rLevel` привязывается к каналу модуля аналоговых входов через вкладку **Соотнесение входов/выходов** – значение датчика уровня поступает в переменную автоматически каждый цикл задачи.
2. Таймер TON запускается, когда условие `rLevel > rSetpoint` становится истинным (`IN = TRUE`). Через 2 секунды непрерывного выполнения условия выход таймера `Q` переходит в `TRUE` – насос включается.
3. Как только уровень опускается ниже уставки, условие `IN` становится `FALSE`, таймер сбрасывается, `Q` переходит в `FALSE` – насос выключается.
4. Переменная `bPumpOn` привязывается к каналу модуля дискретных выходов через вкладку **Соотнесение входов/выходов**.

**Как привязать переменную к каналу модуля:** откройте нужный модуль в дереве устройств → вкладка **Соотнесение входов/выходов**. В столбце **Переменная** нажмите ... и выберите переменную из дерева проекта, либо введите имя вручную (например, `PLC_PRG.rLevel`).

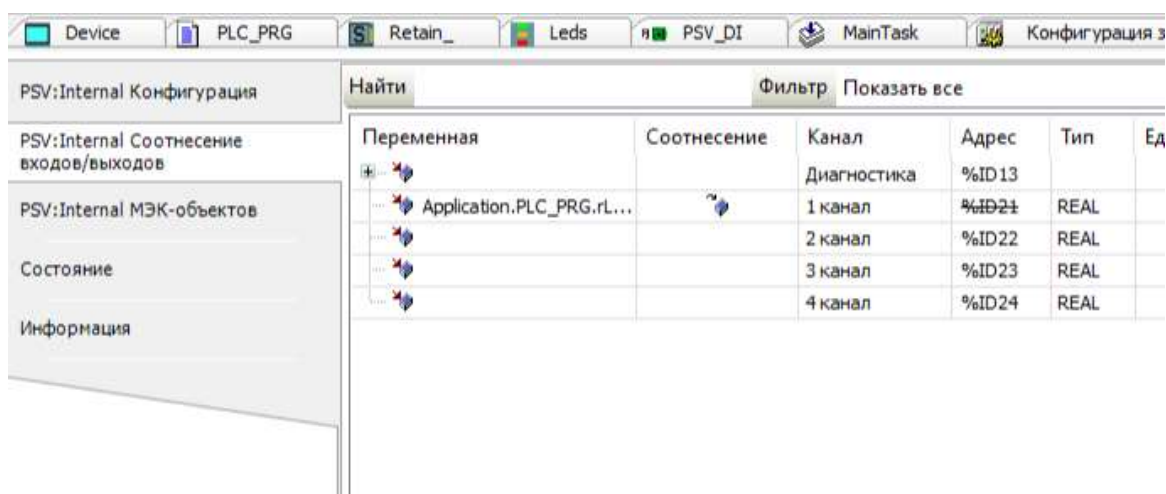


Рисунок 6.2 – Соотнесение входов/выходов модуля аналоговых входов: привязка переменной `PLC_PRG.rLevel` к каналу

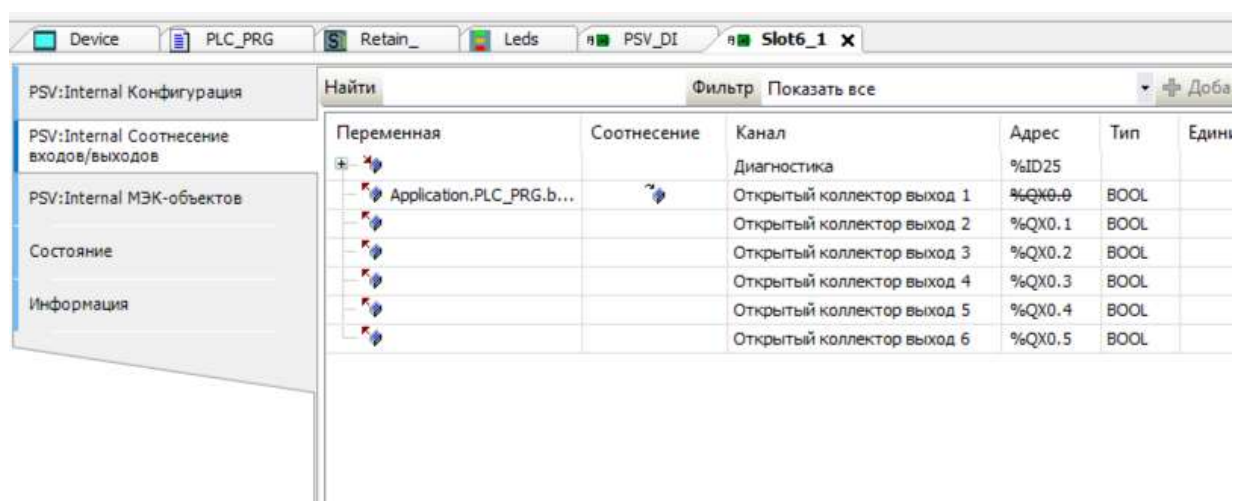


Рисунок 6.3 – Соотнесение входов/выходов модуля дискретных выходов: привязка переменной PLC\_PRG.bPumpOn к каналу

**Примечание.** Если уровень сигнала нестабилен и «дрожит» около значения уставки, насос будет часто включаться и выключаться. В реальных проектах для защиты оборудования применяют гистерезис: уставка включения и уставка выключения задаются разными значениями (например, включить при 75%, выключить при 65%).

## 7 Создание визуализации

Все приборы серии СТАБУР оснащены сенсорным экраном. CODESYS позволяет создавать экраны визуализации (HMI) непосредственно в проекте – без сторонних SCADA-систем.

### 7.1 TargetVisu и WebVisu

CODESYS поддерживает два типа визуализации, которые могут работать одновременно:

Таблица 7.1 – Сравнение TargetVisu и WebVisu

| Тип               | Где отображается                                   | Когда использовать                                                         |
|-------------------|----------------------------------------------------|----------------------------------------------------------------------------|
| <b>TargetVisu</b> | На встроенном сенсорном экране прибора СТАБУР      | Основной режим: оператор работает непосредственно с прибором               |
| <b>WebVisu</b>    | В веб-браузере на любом ПК, планшете или смартфоне | Удалённый мониторинг и управление по сети без установки дополнительного ПО |

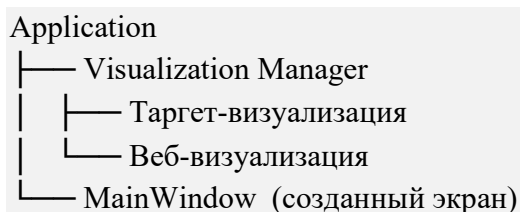
Оба типа строятся на одних и тех же экранах (объектах Visualization), созданных в проекте CODESYS. Отдельно разрабатывать интерфейс для каждого типа не нужно – один и тот же экран отображается и на приборе, и в браузере.

#### 7.1.1 Visualization Manager

При добавлении первого объекта визуализации CODESYS автоматически создаёт узел **Visualization Manager** в дереве под **Application**. Вручную его добавлять не нужно.

Чтобы добавить первую визуализацию: правый клик по **Application** → **Добавить объект** → **Визуализация** → задать имя → **Добавить**.

После этого в дереве проекта появится следующая структура:



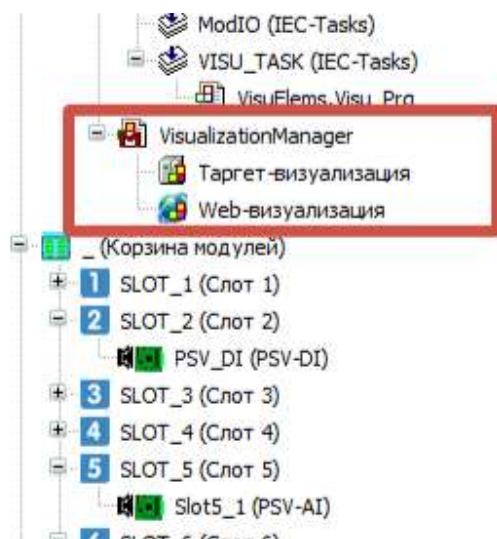


Рисунок 7.1 – Дерево проекта: узел Visualization Manager с дочерними объектами Таргет-визуализация и Веб-визуализация

### 7.1.2 Настройки Visualization Manager

Двойной клик по узлу **Visualization Manager** открывает его собственные настройки. Основные параметры на вкладке **Установки**:

- **Использовать строки Unicode** – необходимо включить, если в визуализации используется кириллица. Без этой опции русскоязычные надписи будут отображаться некорректно (нечитаемые символы).
- **Выбранный стиль** – визуальная тема элементов управления (кнопки, ползунки, индикаторы).
- **Выбранный язык** – язык интерфейса визуализации.
- **Объём памяти для визуализации** и **Размер буфера рисования** – параметры производительности, менять без необходимости не рекомендуется.
- **Режим передачи файлов** – по умолчанию «Передавать файлы визуализации в ПЛК», что необходимо для корректной работы TargetVisu.

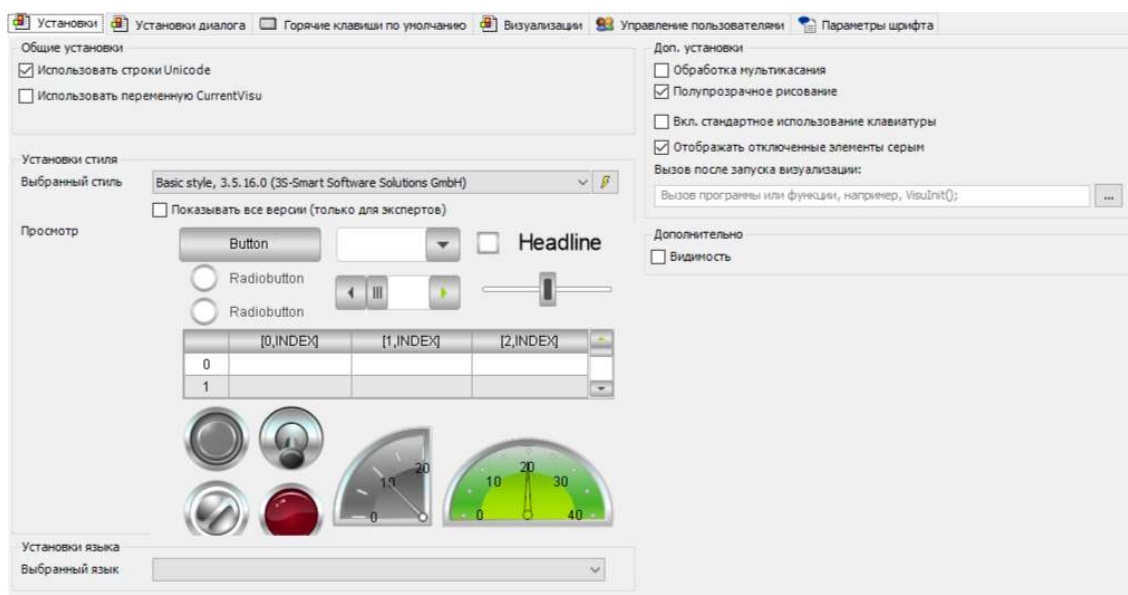


Рисунок 7.2 – Настройки Visualization Manager, вкладка «Установки»: опция «Использовать строки Unicode», выбор стиля и языка

### 7.1.3 Настройки Таргет-визуализации

Двойной клик по узлу **Таргет-визуализация** открывает параметры отображения на экране прибора:

- **Стартовая визуализация** – имя экрана, который открывается при запуске приложения (например, MainWindow).
- **Частота обновления (мс)** – период обновления экрана в миллисекундах. По умолчанию 200 мс (5 раз в секунду).
- **Опции масштабирования** – при режиме Фикс. поля ширины и высоты недоступны: размер экрана берётся из свойств самого объекта визуализации (см. п. 7.1.5). Режимы Изотропная и Анизотропная масштабируют содержимое под размер дисплея с сохранением или без сохранения пропорций соответственно.
- **Ввод с** – тип устройства ввода. Для приборов СТАБУР со встроенным сенсорным экраном выбрать Сенсорный экран.

Стартовая визуализация:  ...

Частота обновления (мс):

[Показать используемые визуализации](#)

Опции масштабирования

☒ Фикс. ☐ Изотропная ☐ Анизотропная

☒ Использовать опции масштабирования для диалогов

☒ Использовать автоматически определяемый размер

☐ Использовать заданный размер

Ширина клиента:

Высота клиента:

Опции представления

☒ Сглаживание

Ввод текста по умолчанию

Ввод с:

Рисунок 7.3 – Настройки узла «Таргет-визуализация»: стартовый экран, опции масштабирования, тип ввода

#### 7.1.4 Настройки Веб-визуализации

Двойной клик по узлу **Веб-визуализация** открывает аналогичные параметры, дополненные настройками для браузерного доступа:

- **Стартовая визуализация** – имя стартового экрана.
- **Имя .htm-файла** – имя файла веб-страницы (по умолчанию webvisu).
- **Частота обновления (мс)** – период обновления страницы в браузере. По умолчанию 200 мс.
- **Размер буфера соединения** – объём данных, передаваемых за один цикл обновления.

Рисунок 7.4 – Настройки узла «Веб-визуализация»: имя htm-файла, стартовый экран, буфер соединения

**Доступ к WebVisu в браузере.** После загрузки приложения в контроллер откройте в браузере: `http://<IP-адрес прибора>:9090/webvisu.htm`. Дополнительное программное обеспечение не требуется. Если страница не открывается – убедитесь, что приложение запущено (режим Run) и в настройках Visualization Manager выбрано «Передавать файлы визуализации в ПЛК».

**Порт веб-сервера.** Порт задаётся не в настройках проекта CODESYS, а в системном конфигурационном файле устройства `/opt/codesys/CODESYSControl.cfg` в секции `[CmpWebServer]`: `[CmpWebServer] WebServerPortNr=9090`. На приборах СТАБУР стандартное значение – `**9090**` (в стандартной поставке CODESYS по умолчанию используется 8080). Для изменения порта отредактируйте этот файл и перезапустите сервис командой `systemctl restart codesys`. Доступ к файлу возможен через терминал (SSH/PuTTY) с логином root.

### 7.1.5 Задание размера экрана визуализации

Размер каждого экрана (ширина и высота в пикселях) задаётся в свойствах самого объекта визуализации, а не в настройках Таргет- или Веб-визуализации:

1. В дереве проекта – правый клик по нужному экрану (например, MainWindow) → **Свойства**.
2. Перейти на вкладку **Визуализация**.

3. Установить переключатель **«Использовать заданный размер визуализации»**.
4. Ввести **Ширину** и **Высоту** в пикселях в соответствии с разрешением дисплея прибора. Для ПЛК СТАБУР 07 – 800 × 480.
5. Нажать **ОК**.

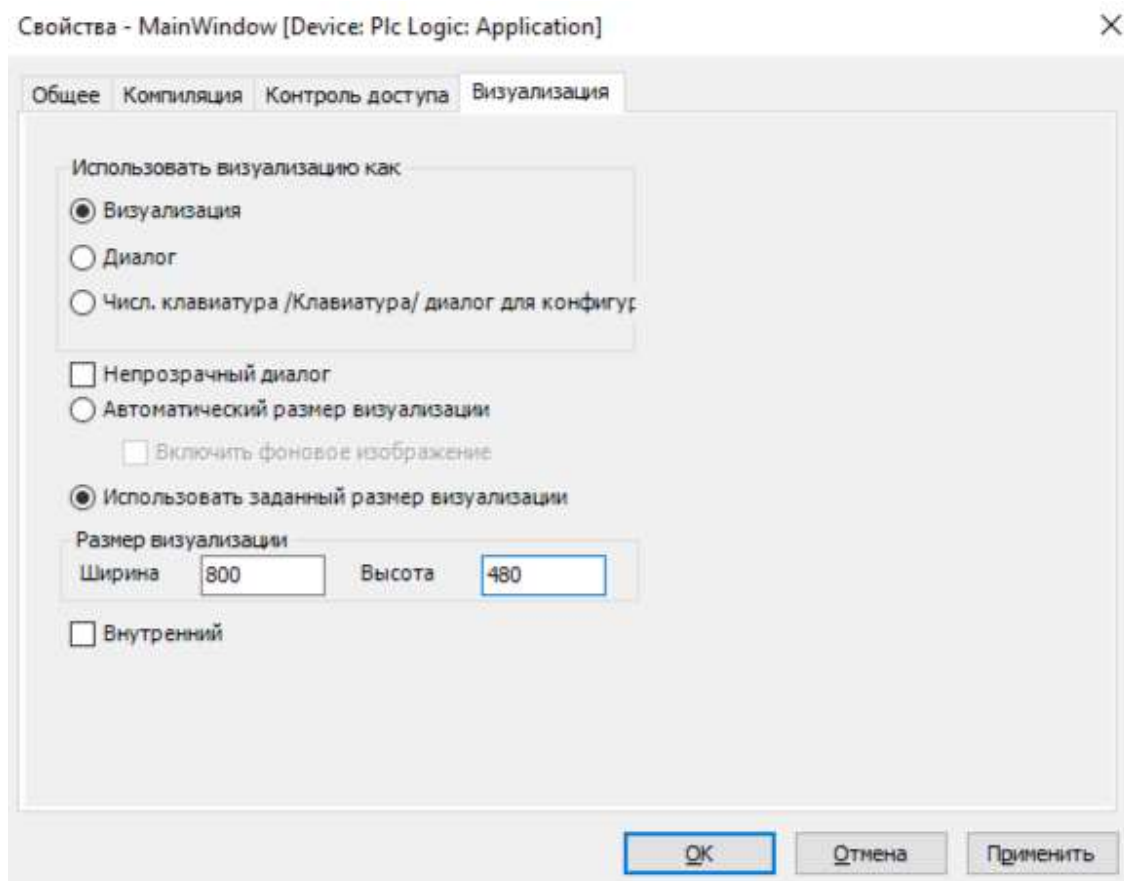


Рисунок 7.5 – Свойства экрана визуализации, вкладка «Визуализация»: опция «Использовать заданный размер», поля ширины 800 и высоты 480

## 7.2 Создание экрана визуализации

1. Правый клик по **Application** → **Добавить объект** → **Визуализация**.
2. Укажите имя экрана, например MainScreen. Нажмите **Добавить**.
3. Откроется графический редактор визуализации – пустой холст заданного размера. Слева или справа (в зависимости от раскладки окон) находится панель инструментов **Панель инструментов визуализации** с доступными элементами.
4. При необходимости создайте дополнительные экраны (например, AlarmScreen, SettingsScreen) аналогичным образом. Переходы между экранами настраиваются через элемент **Frame** или через свойства кнопки: **Конфигурация ввода** → например, **OnClick** → **Изменить показанную визуализацию** → указать имя нужного экрана.

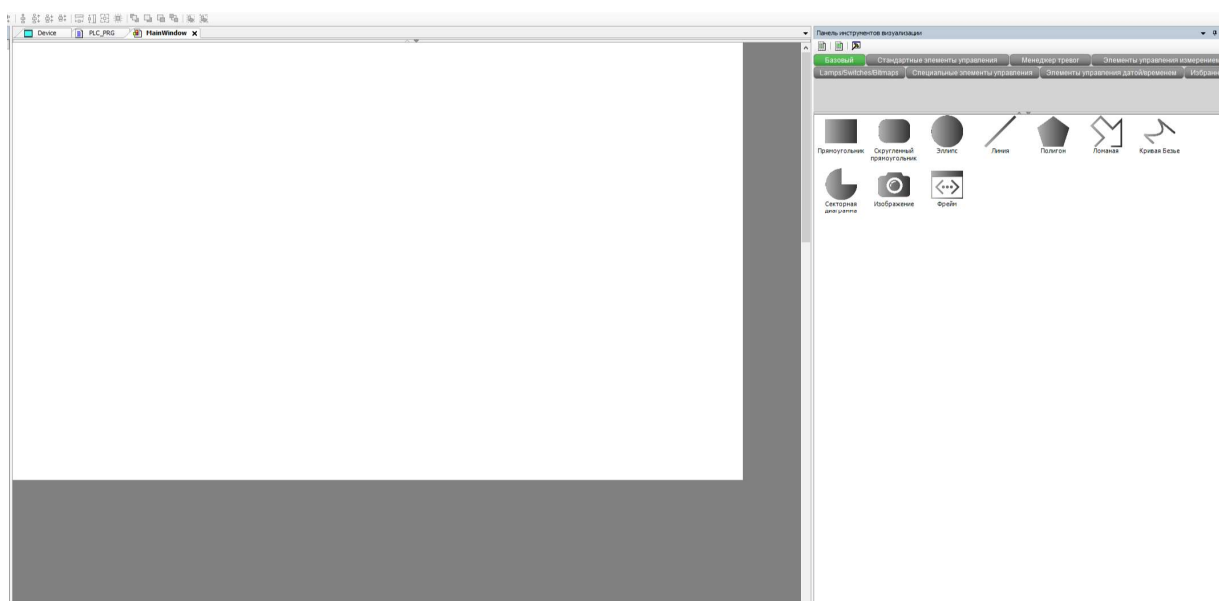


Рисунок 7.6 – Пустой редактор визуализации: холст 800×480, панель инструментов визуализации

## 7.3 Элементы и привязка переменных

Панель инструментов визуализации разделена на категории:

Таблица 7.3 – Элементы панели инструментов визуализации

| Категория                                 | Содержимое                                                                                                         |
|-------------------------------------------|--------------------------------------------------------------------------------------------------------------------|
| <b>Базовый</b>                            | Основные геометрические фигуры, текст, кнопки, поля ввода, изображения – строительный материал большинства экранов |
| <b>Стандартные элементы управления</b>    | Стандартные виджеты: чекбоксы, переключатели, выпадающие списки, ползунки                                          |
| <b>Менеджер тревог</b>                    | Элементы для отображения списка аварийных сообщений и работы с тревогами                                           |
| <b>Элементы управления измерением</b>     | Стрелочные приборы, шкалы, индикаторы уровня и другие аналоговые отображения                                       |
| <b>Lamps / Switches / Bitmaps</b>         | Готовые графические лампочки, переключатели и растровые элементы                                                   |
| <b>Специальные элементы управления</b>    | Таблицы, тренды (графики), элементы рецептов и другие сложные компоненты                                           |
| <b>Элементы управления датой/временем</b> | Поля отображения и ввода даты и времени                                                                            |
| <b>Избранное</b>                          | Пользовательские избранные элементы                                                                                |

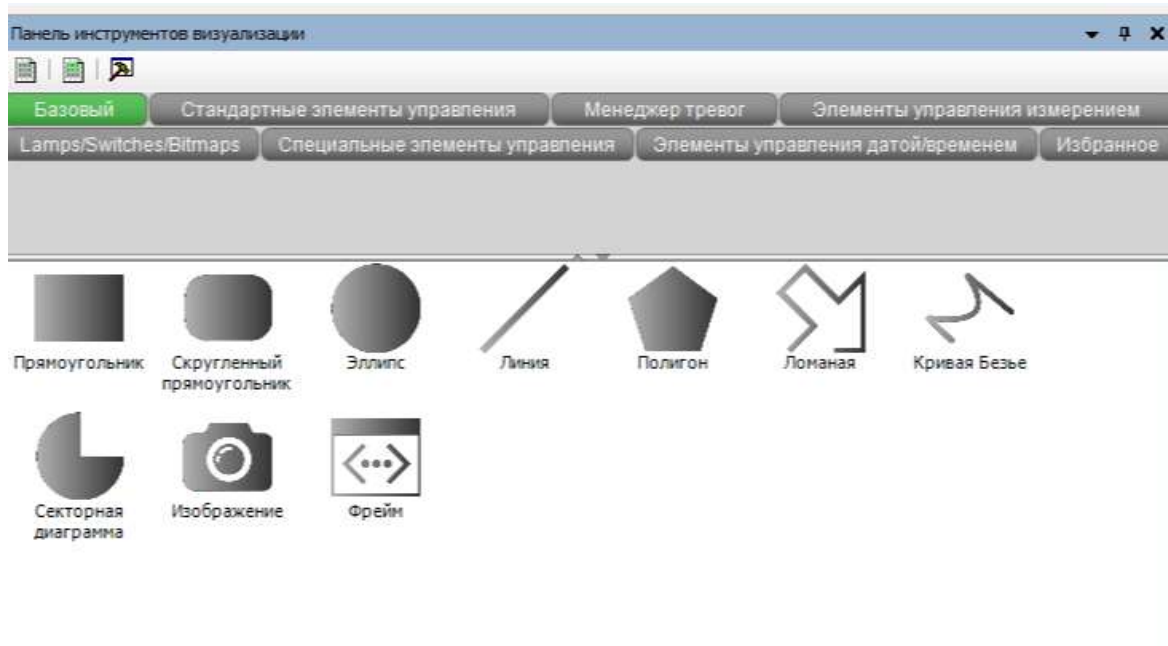


Рисунок 7.7 – Панель инструментов визуализации: категории элементов

### Привязка переменной к элементу:

1. Разместите элемент на холсте (перетащите с панели инструментов визуализации).
2. Откройте **свойства элемента** (Properties – панель справа или двойной щелчок).
3. В нужном поле (например, **Конфигурация ввода** → **Нажатие** для кнопки, **Текстовые переменные** для дисплея) нажмите кнопку ... – откроется **ассистент ввода** (Input Assistant).
4. В ассистенте ввода выберите переменную из дерева проекта.

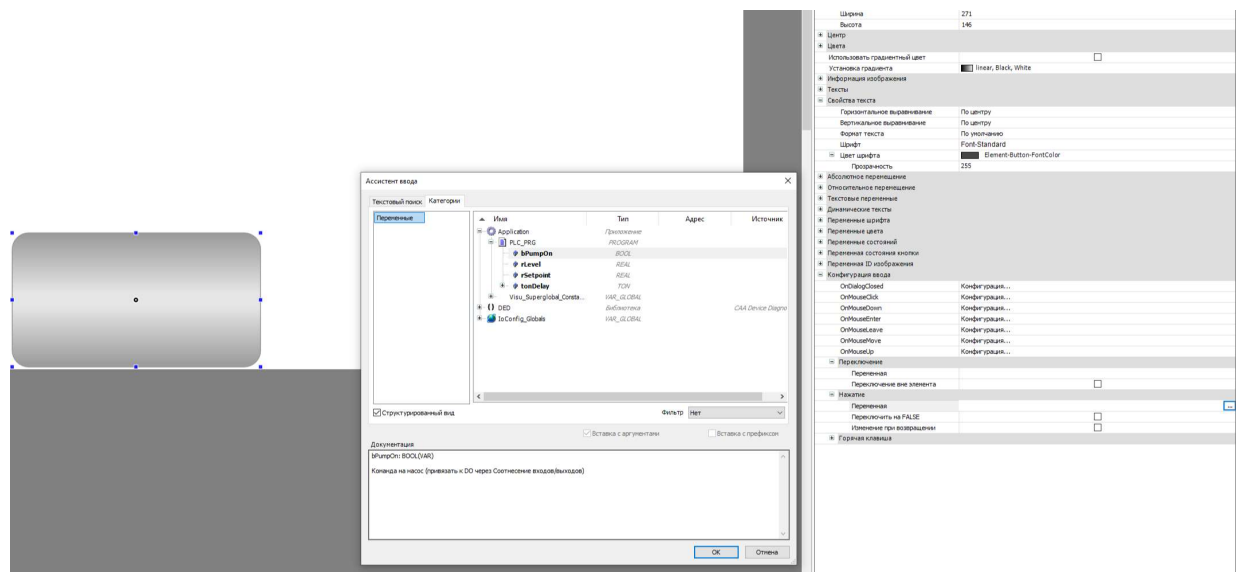


Рисунок 7.8 – Свойства элемента «Кнопка»: поле Нажатие → переменная, ассистент ввода для выбора переменной

## 7.4 Фреймы

**Фрейм** (Frame) – это переиспользуемый фрагмент визуализации. Например: шапка экрана с логотипом и временем, навигационное меню, блок аварийных сообщений.

Порядок использования:

1. Создайте отдельную визуализацию – например, Header – и разместите в ней элементы шапки.
2. На основном экране добавьте элемент **Фрейм** (из панели инструментов визуализации).
3. В свойствах Frame укажите ссылку на визуализацию Header.

Фрейм можно размещать на любом количестве экранов – при изменении Header обновление произойдёт везде.

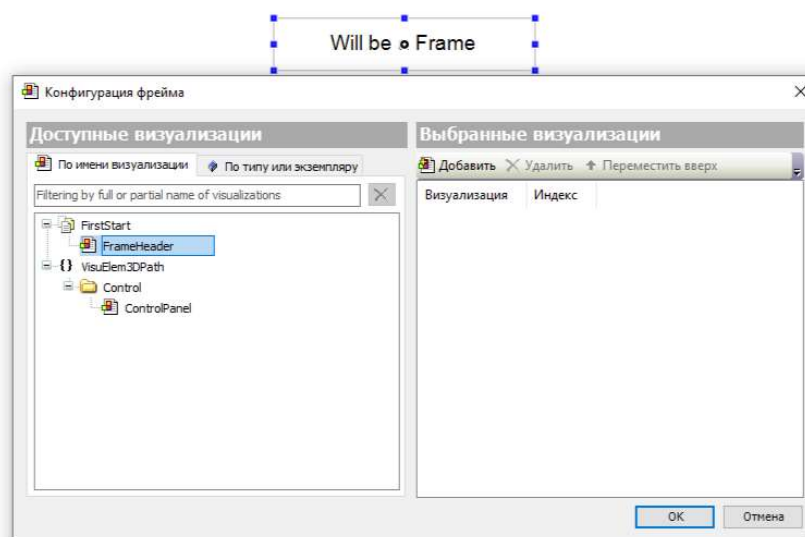


Рисунок 7.10 – Конфигурация фрейма

## 8 Дополнительные материалы и проверка результата

### 8.1 Чек-лист после первой загрузки

После успешной загрузки проекта в прибор выполните следующие проверки:

Таблица 8.1 – Чек-лист проверки после первой загрузки

| # | Проверка                                             | Ожидаемый результат                                                 |
|---|------------------------------------------------------|---------------------------------------------------------------------|
| 1 | Статус Application в дереве устройств                | Зелёный индикатор                                                   |
| 2 | Мониторинг переменных в PLC_PRG                      | Значения обновляются в режиме онлайн                                |
| 3 | Отсутствие ошибок на вкладке «Сообщения» узла Device | Журнал пуст или содержит только информационные сообщения            |
| 4 | Визуализация на экране прибора (если настроена)      | Отображается TargetVisu                                             |
| 5 | Диагностика модулей (если добавлены)                 | Канал «Состояние ошибки» = FALSE (нет ошибки), число ответов растёт |
| 6 | Индикаторы LEDS (если управление включено)           | Светодиоды реагируют на запись в переменные                         |

### 8.2 Диагностика модулей

Каждый модуль ввода-вывода в дереве устройств имеет диагностические каналы. Они доступны на вкладке **Соотнесение входов/выходов** в режиме онлайн в группе **Диагностика**:

Таблица 8.2 – Диагностические каналы модулей в CODESYS

| Канал                | Тип   | Описание                                                |
|----------------------|-------|---------------------------------------------------------|
| Состояние ошибки     | BOOL  | TRUE – ошибка модуля;<br>FALSE – ошибок нет             |
| Результат выполнения | DWORD | Код результата последней операции. ERR_OK (0) – успешно |
| Число запросов       | DWORD | Количество обращений к модулю                           |
| Число ответов        | DWORD | Количество успешных ответов от модуля                   |
| Число неотвечов      | DWORD | Количество обращений, на которые модуль не ответил      |
| Код типа модуля      | BYTE  | 1 – DOR, 2 – DI, 3 – AI, 4 – DOOC, 5 – DOS, 6 – AO, 7 – |

| Канал             | Тип  | Описание                                                            |
|-------------------|------|---------------------------------------------------------------------|
|                   |      | TERM                                                                |
| Версия ПО         | REAL | Версия встроенного программного обеспечения модуля. Формат: Ver.Rev |
| Аппаратная версия | REAL | Версия аппаратной части модуля. Формат: Ver.Rev                     |

Наиболее частая причина ненулевого кода – несоответствие типа модуля в проекте CODESYS и физически установленного модуля в корзине. В этом случае необходимо проверить конфигурацию дерева устройств и фактический состав модулей в приборе.

Данный документ охватывает базовые принципы работы с модулями ввода-вывода. Подробная документация по каждому модулю PSV, а также настройка интерфейсных модулей и коммуникационных протоколов – Modbus Serial Master, Modbus RTU Slave, Modbus TCP, CANOpen Manager и CANOpen Device – описаны в отдельном документе:

**ПЛК СТАБУР. Руководство по программированию CODESYS** (доступно на сайте [psve.ru](http://psve.ru)).

Руководство включает:

- общую методику конфигурации интерфейсных модулей;
- настройку контроллера в режиме Modbus Serial Master;
- настройку контроллера в режиме Modbus RTU Slave;
- настройку контроллера в режиме Modbus TCP Master и Slave;
- настройку контроллера в режиме CANOpen Manager (Master);
- настройку контроллера в режиме CANOpen Device (Slave).

## 8.3 Куда обращаться и полезные ссылки

Таблица 8.3 – Полезные ссылки и контакты

| Ресурс                                       | Адрес                                                                  |
|----------------------------------------------|------------------------------------------------------------------------|
| Сайт производителя                           | <a href="http://psve.ru">psve.ru</a>                                   |
| Скачать дистрибутивы и таргет-файлы          | <a href="http://psve.ru">psve.ru</a> (раздел технической документации) |
| Руководство по эксплуатации (РЭ)             | <a href="http://psve.ru">psve.ru</a> (раздел технической документации) |
| Руководство по программированию CODESYS (РП) | <a href="http://psve.ru">psve.ru</a> (раздел технической документации) |
| Справка CODESYS (встроенная)                 | Клавиша <b>F1</b> в среде CODESYS                                      |
| Техническая поддержка                        | <a href="mailto:help@psvyaz.ru">help@psvyaz.ru</a>                     |
| Тест-драйв оборудования                      | Заявка на сайте <a href="http://psve.ru">psve.ru</a>                   |