

Новые возможности CODESYS V3.5 SP16

В статье рассмотрен новый функционал CODESYS V3.5, появившейся в версиях SP15-SP16, который может использоваться при создании проектов для контроллеров OBEH со свежими версиями прошивок.

ОГЛАВЛЕНИЕ

Редакторы программирования и интерфейс IDE

- [1. Автоматический контроль порядка выполнения блоков в CFC](#)
- [2. Библиотека Net Base Services](#)
- [3. Импорт проектов из CoDeSys V2.3](#)
- [4. Встроенный браузер для менеджера библиотек и визуализации](#)
- [5. Поддержка типов данных LDT/LDATE/LTOD](#)
- [6. Счетчики импульсов с типом LWORD](#)
- [7. Поддержка функций асинхронного шифрования](#)
- [8. Опциональные аргументы для функций и методов](#)
- [9. Автоматическое создание стандартных методов](#)
- [10. Операторы POUNAME и POSITION](#)
- [11. Возвращение «классического» автообъявления в ST](#)
- [12. Выбор активного приложения](#)
- [13. Сортировка файлов на вкладке Device – Файлы](#)
- [14. Библиотека CAA File – исправление ошибки в ФБ DirList](#)
- [15. Оптимизации](#)
- [16. Загрузка отсутствующих библиотек через панель сообщений](#)

Визуализация

- [1. Поддержка технологии overlay](#)
- [2. Порядок переключения элементов с помощью клавиши Tab](#)
- [3. Сортировка сообщений в таблице и баннере тревог](#)
- [4. Поддержка кириллицы в легенде тренда](#)
- [5. Дополнительные настройки внешнего вида для трассировки и тренда](#)
- [6. Выбор списка текстов в выпадающем списке с помощью строковой переменной](#)
- [7. Оптимизации](#)

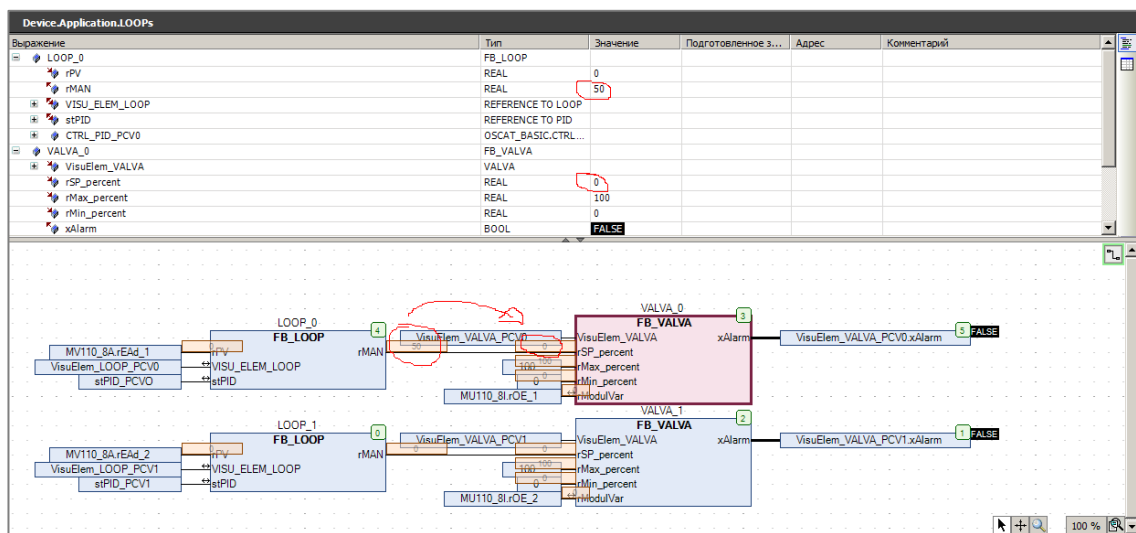
Modbus

- [1. Modbus Slave Device \(Serial/TCP\) – запись в holding-регистры из программы ПЛК](#)
- [2. Modbus Slave Device \(Serial/TCP\) – независимые области памяти для бит](#)
- [3. Modbus TCP Slave Device – возможность привязки к конкретному адаптеру](#)
- [4. Modbus TCP Slave Device – увеличение числа одновременно подключенных клиентов](#)
- [5. Modbus TCP Slave Device – функционал шлюза RTU/TCP \(Serial Gateway\)](#)
- [6. Все компоненты – улучшение возможностей диагностики](#)
- [7. Modbus Slave COM Port – возможность добавление слэйвов с совпадающими Slave ID](#)
- [8. Все компоненты – возможность группового обновления версий компонентов](#)
- [9. Modbus Master \(Serial/TCP\) – оптимизации в драйвере](#)
- [10. Modbus Master \(Serial/TCP\) – улучшения в механизме редактирования каналов](#)
- [11. Modbus Slave Device \(Serial/TCP\) – дополнительные методы и переменные](#)
- [12. Исправление ошибок](#)

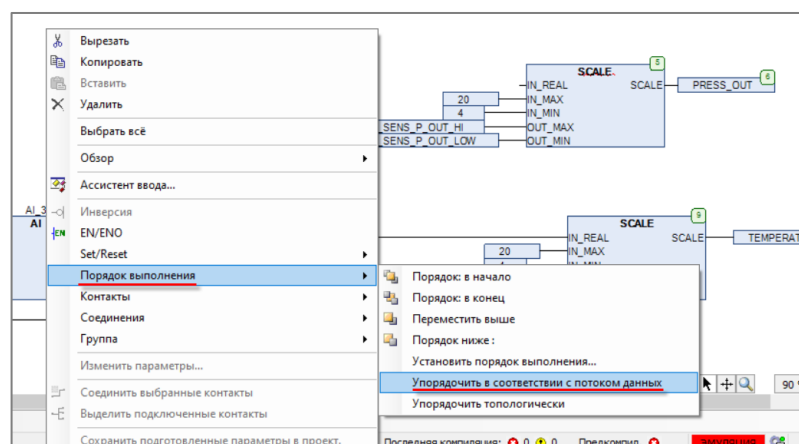
Редакторы программирования и интерфейс IDE

1. Автоматический контроль порядка выполнения блоков в CFC

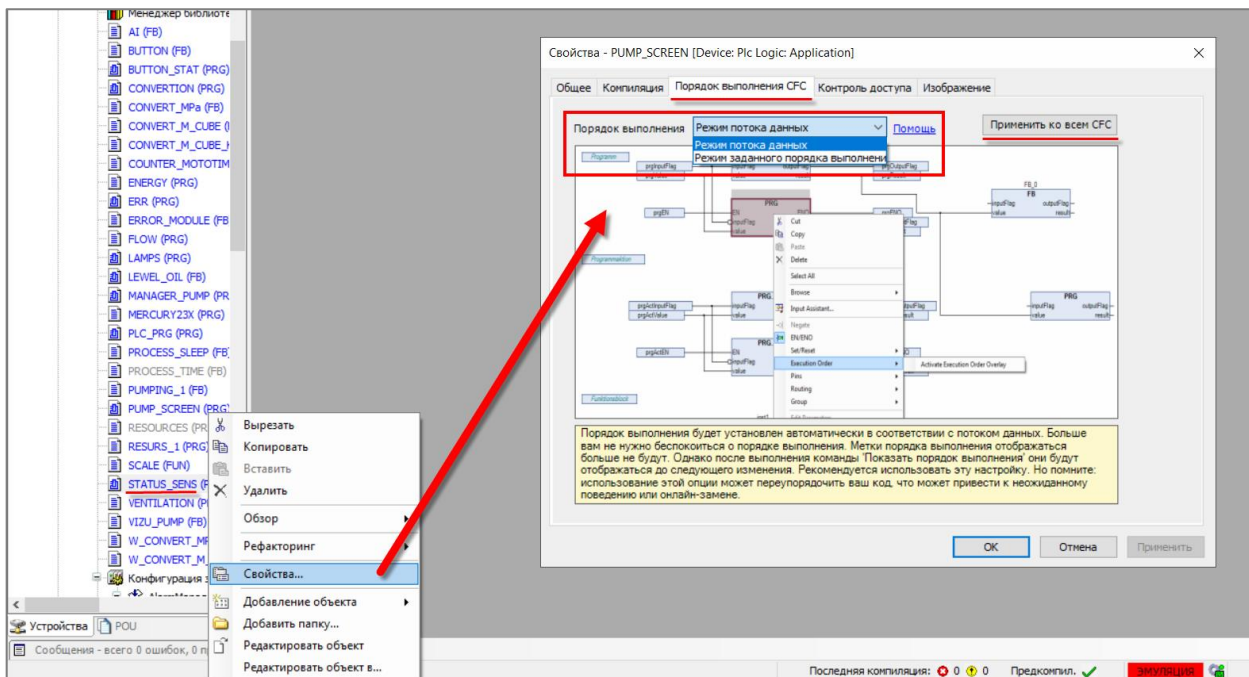
В языке CFC порядок выполнения блоков определяется номерами, отображаемыми в их верхнем правом углу. По умолчанию номер каждого добавляемого на холст блока зависит от его позиции по вертикали по отношению к другим блокам (например, если добавляемый блок по вертикали будет расположен между блоками с номерами 1 и 2, то он получит номер 2, а исходный блок 2 получит номер 3). При перемещении блоков по холсту их номера не меняются, что может привести к следующим ситуациям («значение с выхода одного блока не передается на вход другого»):



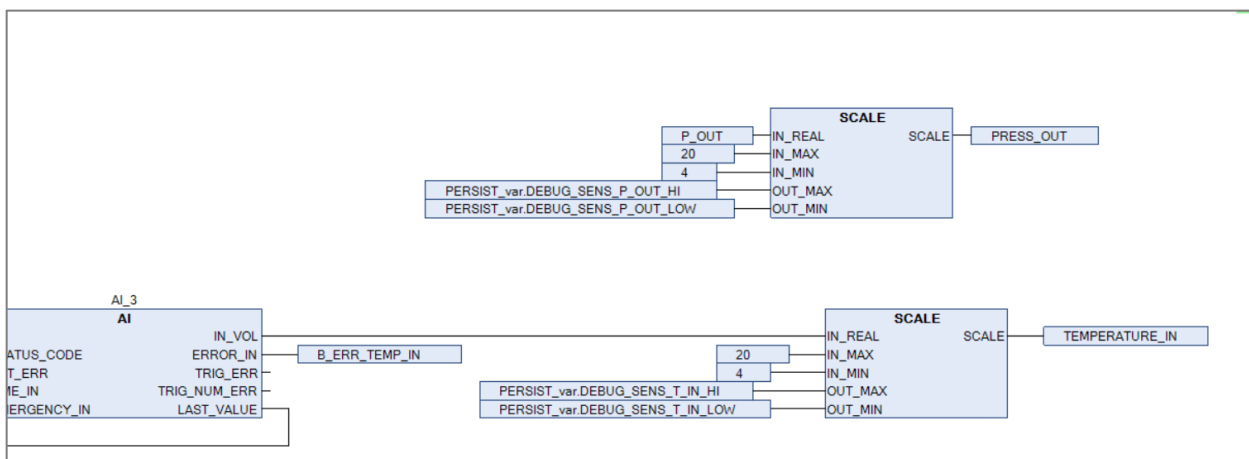
Раньше пользователям приходилось тщательно контролировать номера блоков и при необходимости менять их через контекстное меню – обычно с помощью команды **Упорядочить в соответствии с потоком данных**, которая перераспределяла номера блоков по принципу «слева направо, сверху вниз».



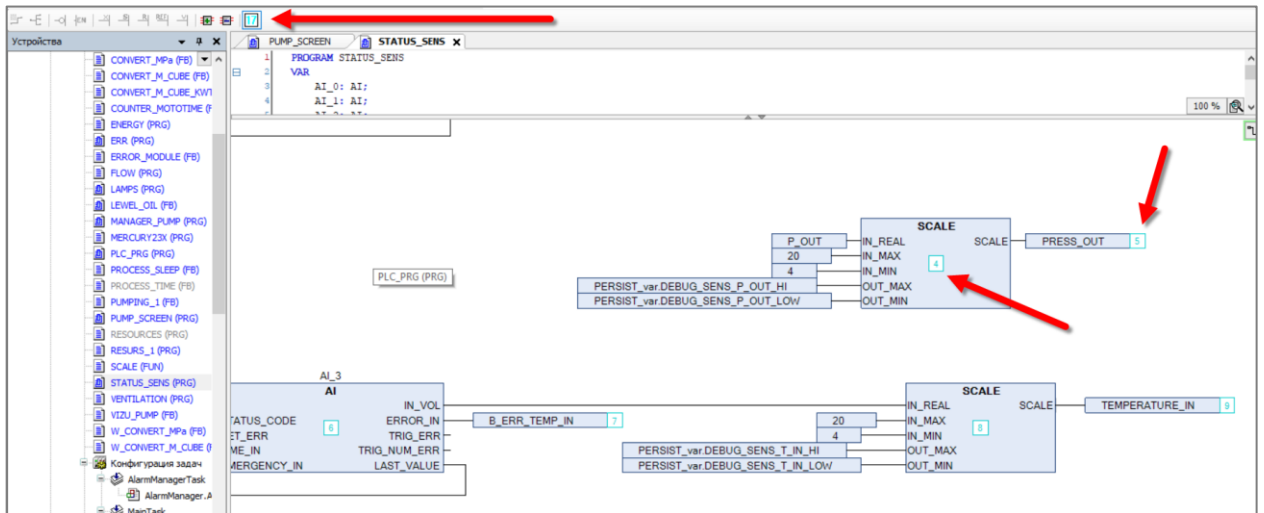
В свежих версиях CODESYS можно включить *автоматический режим упорядочивания номеров*. Для этого надо нажать ПКМ на любой POU с языком CFC, выбрать пункт **Свойства**, в открывшемся окне выбрать вкладку **Порядок выполнения CFC** и задать порядок выполнения **Режим потока данных** (**Режим заданного порядка выполнения** соответствует стандартному поведению из предыдущих версий, описанных выше). С помощью кнопки **Применить ко всем CFC** можно автоматически изменить эту настройку во всех POU проекта.



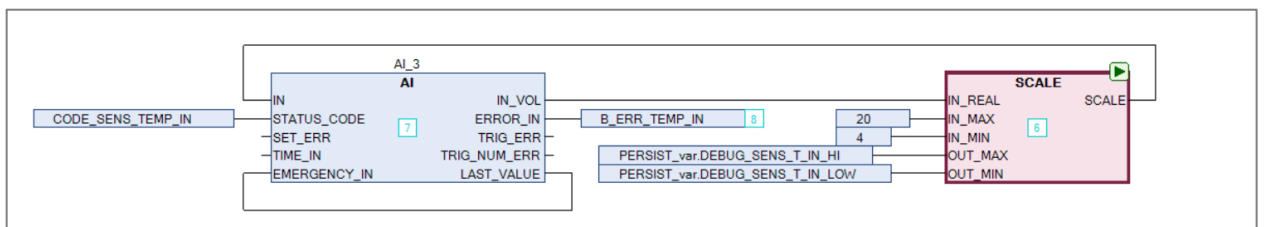
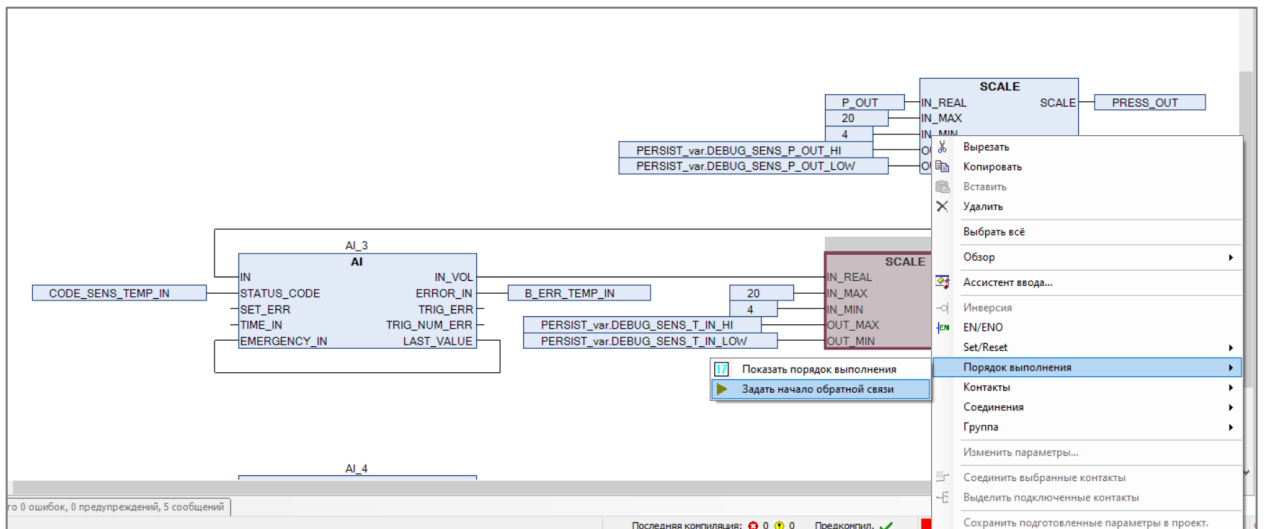
В результате номера блоков на холсте перестанут отображаться – они будут распределяться автоматически в соответствии с потоком данных и меняться «на лету» при перетаскивании блоков по холсту.



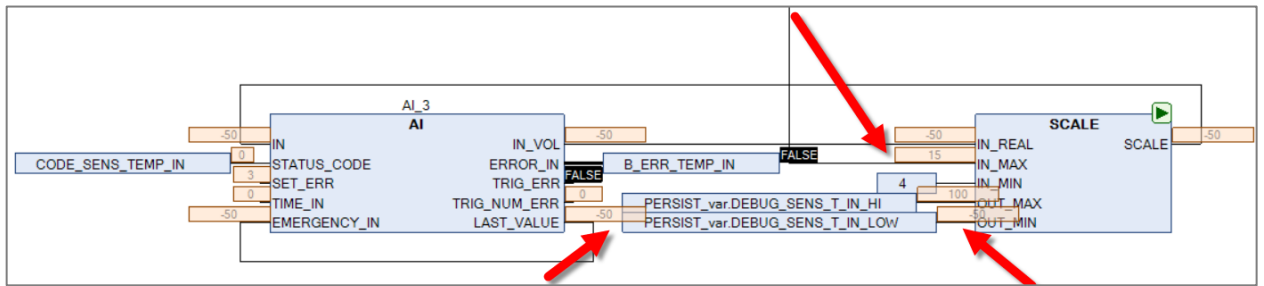
Если пользователь хочет все-таки удостовериться в порядковых номерах блоков – то это можно сделать с помощью специальной кнопки на панели инструментов. Номера будут подсвечиваться лазерным цветом и исчезнут после первого нажатия на холст.



В некоторых случаях хотелось бы использовать автоматическое распределение номеров, но одновременно иметь возможность задавать их вручную – в частности, в тех блоках, которые охвачены обратной связью. Для этого используется специальная команда контекстного меню:

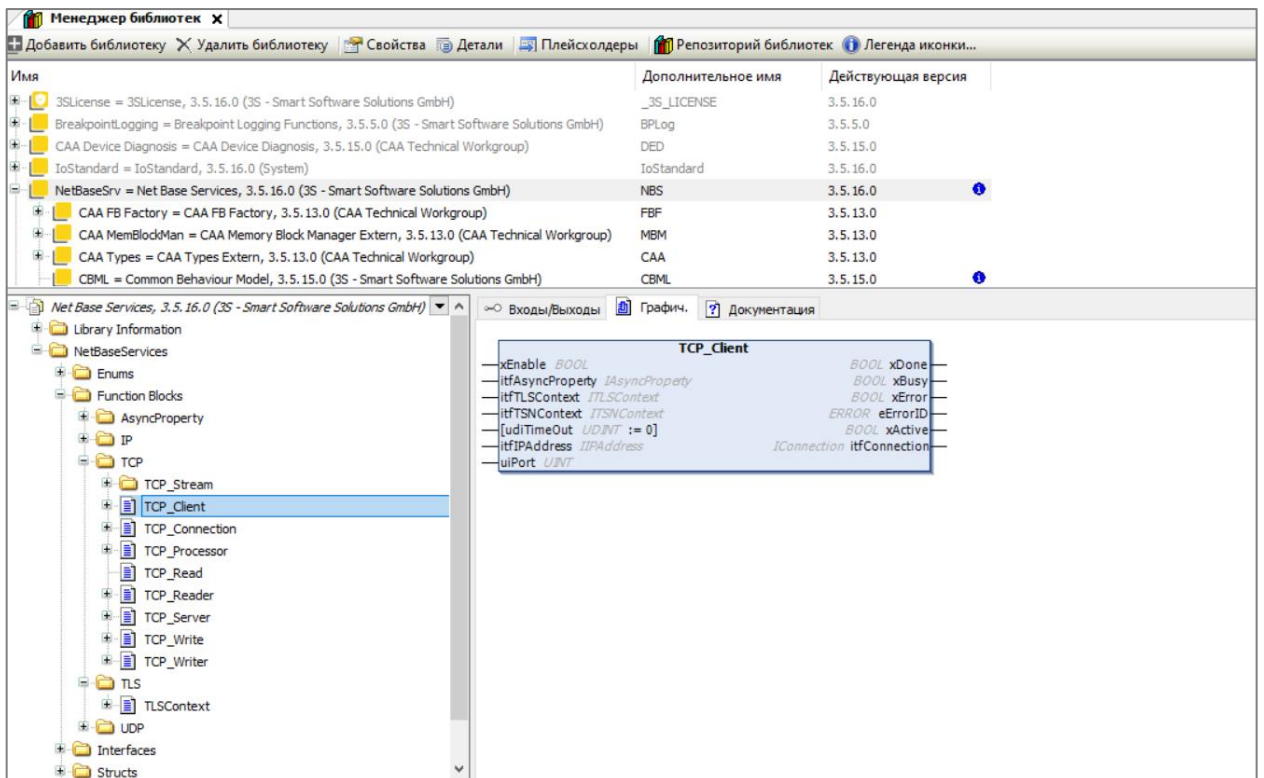


И еще одно мелкое улучшение в CFC – в режиме отладки значение теперь отображается не только в начале соединительной линии, но и на ее конце. Это удобно на больших схемах с длинными линиями связи.



2. Библиотека Net Base Services

Новая библиотека **Net Base Services** является более продвинутой версией библиотеки **CAA Net Base Services** (используемой для обмена по протоколам TCP и UDP) с поддержкой криптографического протокола **TLS**. В будущем разработчики планируют опубликовать пример использования библиотеки в [CODESYS Store](#).

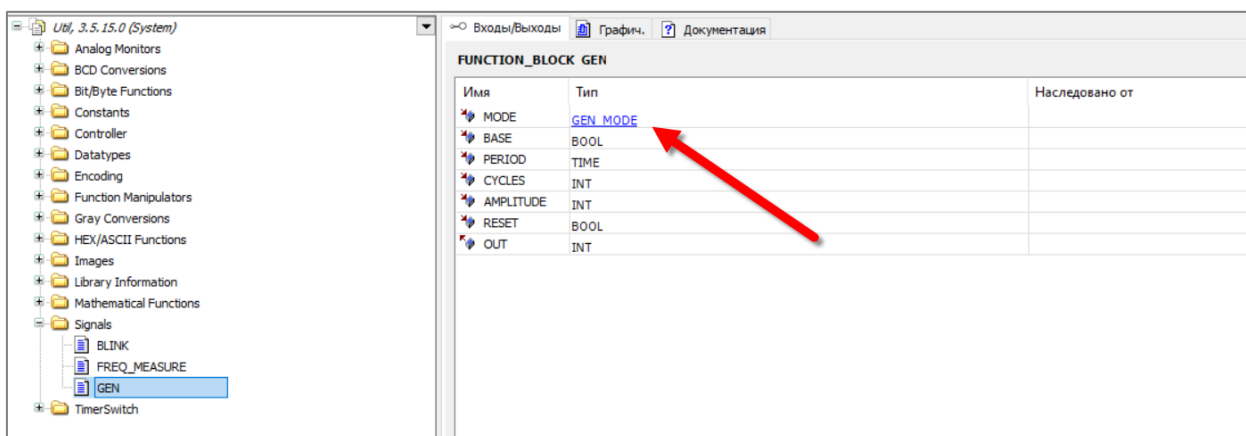


3. Импорт проектов из CoDeSys V2.3

Плагин для импорта проектов из CoDeSys V2.3 теперь не входит в дистрибутив CODESYS, но может быть [загружен из CODEYS Store](#). Обратите внимание, что плагин доступен только для 32-битной версии среды разработки.

4. Встроенный браузер для менеджера библиотек и визуализации

Теперь документация к библиотекам и визуализация в среде программирования отображаются с помощью встроенного браузера на движке Chromium. Это позволяет в документации использовать гиперссылки для просмотра составных типов данных (структуры, перечисления и т.д.) и тестировать все возможности визуализации прямо в среде разработки (при онлайн-подключении) без каких-либо ограничений (раньше, например, в ней не поддерживался функционал [технологии overlay](#)).



5. Поддержка типов данных LDT/LDATE/LTOD

Типы данных **LDT/LDATE/LTOD** являются 64-битными версиями типов **DT/DATE/TOD** с расширенным диапазоном и поддержкой точности до наносекунд.

```
ldtDateAndTime:    LDT    := LDT#2021-05-14-09:15:10.123456789;  
ltodTimeOfDay:    LTOD    := LTOD#12:34:56.123456789;  
ldateDate:        LDATE   := LDATE#3000-05-12;
```

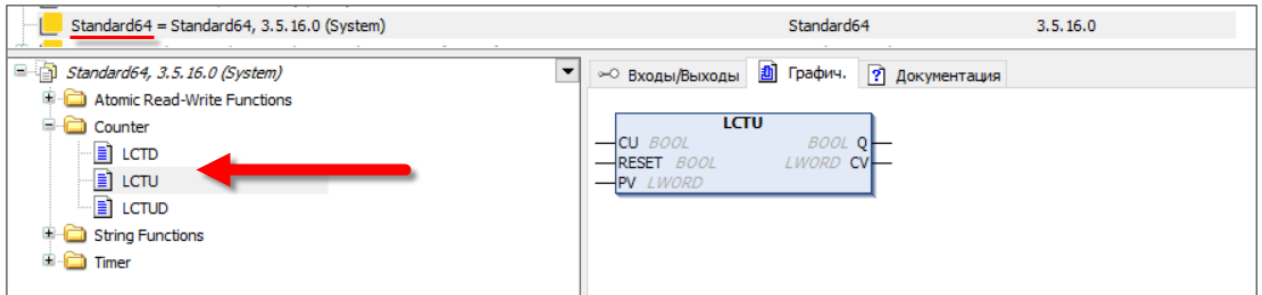
Интересный момент – в библиотеке OSCAT Basic обозначения «LDT», «LDATE» и «LTOD» используются в качестве имен переменных структуры **CALENDAR**. С введением в CODESYS одноименных типов данных эти названия стали зарезервированными, поэтому попытка объявить данную структуру (например, она используется в ФБ **CALENDAR_CALC**) приведет к ошибкам компиляции. Но так как библиотека доступна в исходных кодах – вы можете открыть ее, изменить имена этих переменных и переустановить в репозиторий библиотек для решения этой проблемы. Кроме того, [исправленную версию](#) уже выложили в CODESYS Store.

```
CALENDAR [H3 BASIC] x  
1  TYPE CALENDAR :  
2  STRUCT  
3  UTC : DT;           (* world time UTC *)  
4  LDT : DT;           (* local time *)  
5  LDATE : DATE;       (* local date *)  
6  LTOD : TOD;         (* local time of day *)
```

6. Счетчики импульсов с типом LWORD

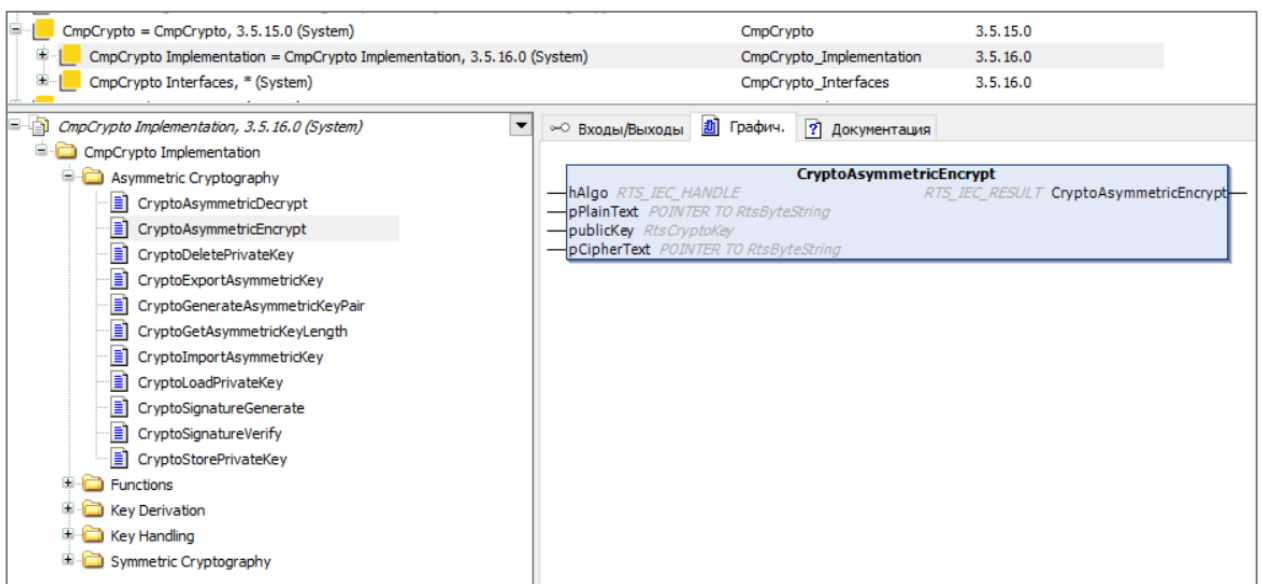
В библиотеке **Standard** есть ФБ СТU/СТD/СТUD, которые используются для подсчета импульсов. Тип переменной, в которой хранится результат подсчета – **WORD**, т.е. счетчики способны подсчитать только **65535** импульсов, после чего произойдет переполнение.

В свежих версиях CODESYS в библиотеку **Standard64** добавлены ФБ LCTU/LCTD/LCTUD – это версии стандартных ФБ счетчиков с типом данных **LWORD**. Они способны подсчитать до $(2^{64}-1)$ импульсов – думаем, для большинства задач этого хватит.



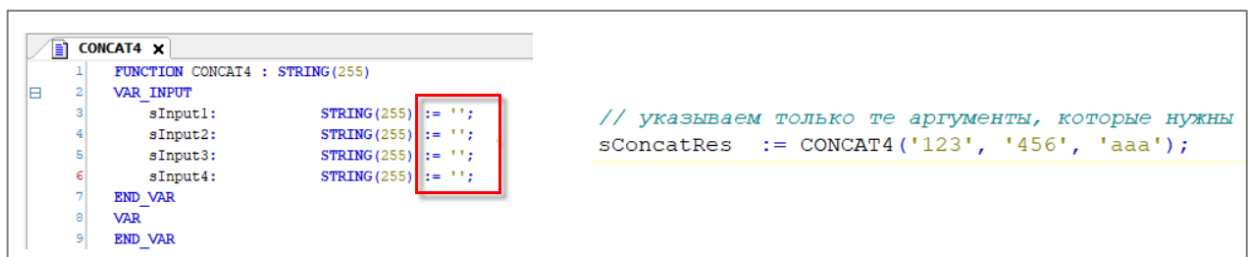
7. Поддержка функций асинхронного шифрования

В библиотеку **CmpCrypto** добавлены функции для асинхронного шифрования.



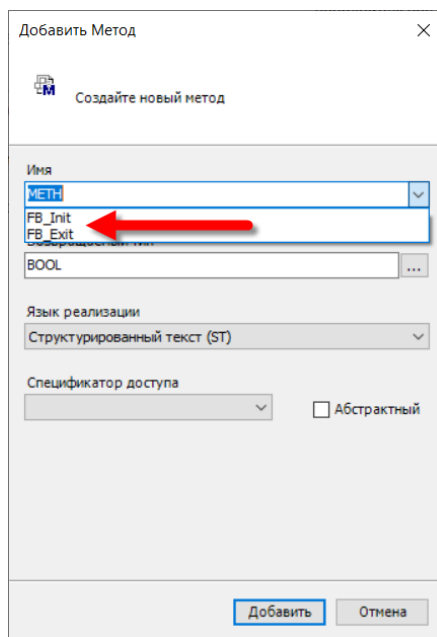
8. Опциональные аргументы для функций и методов

Теперь если входы функций и методов имеют начальные значения – то их можно не указывать при вызове (ранее требовалось явное присвоение всех переменных входам функции/метода при вызове).



9. Автоматическое создание стандартных методов

При добавлении методов теперь можно выбрать шаблон одного из стандартных методов (**FB_Init** или **FB_Exit**).



Добавить Метод

Создайте новый метод

Имя
METH
FB_Init
FB_Exit

BOOL

Язык реализации
Структурированный текст (ST)

Спецификатор доступа
Абстрактный

Добавить Отмена

10. Операторы __POUNAME и __POSITION

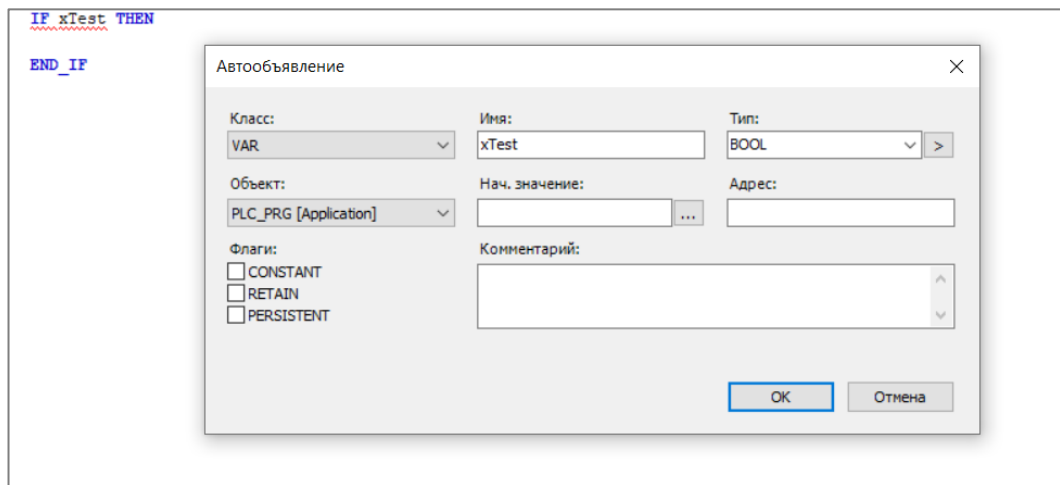
Новые системные операторы **__POUNAME** и **__POSITION** позволяют получить имя POU (включая все пространства имен) и номер его строки. Это может пригодиться, например, при логировании ошибок программы.

Выражение	Тип	Значение
xEvent	BOOL	TRUE
sPouName	STRING	'PLC_PRG'
sLineNumber	STRING	'Строка 7, Столбец 2 (Реализ.)'

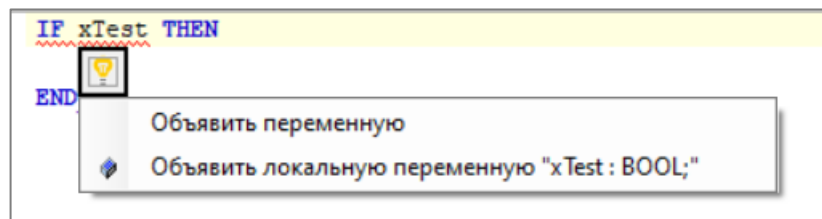
```
1 // ..
2 // ..
3 // ..
4
5 IF xEvent TRUE THEN
6     sPouName 'PLC_PRG' := __POUNAME ();
7     sLineNumber 'Строка 7, ' := __POSITION ();
8 END_IF RETURN
```


11. Возвращение «классического» автообъявления в ST

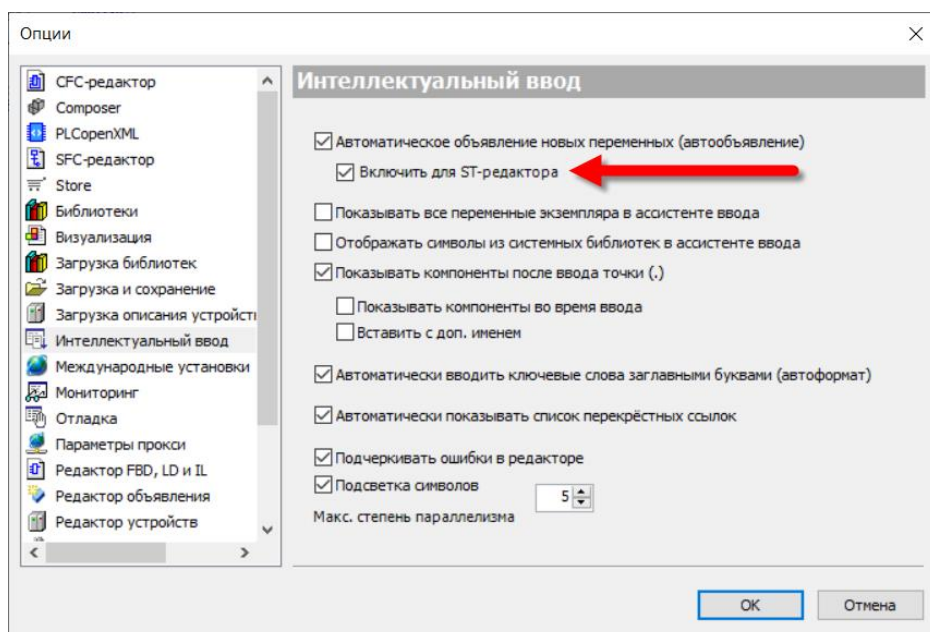
В прошлых версиях CODESYS после ввода в редакторе ST имени несуществующей переменной и нажатия на Enter появлялось окно автообъявления:



Начиная с версии **CODESYS V3.5 SP14** поведение изменилось: после ввода несуществующей переменной теперь нужно было разместить рядом с ней курсор, дождаться появления «лампочки», нажать на нее (или нажать Ctrl+.), выбрать нужную команду – и только после этого увидеть окно автообъявления.

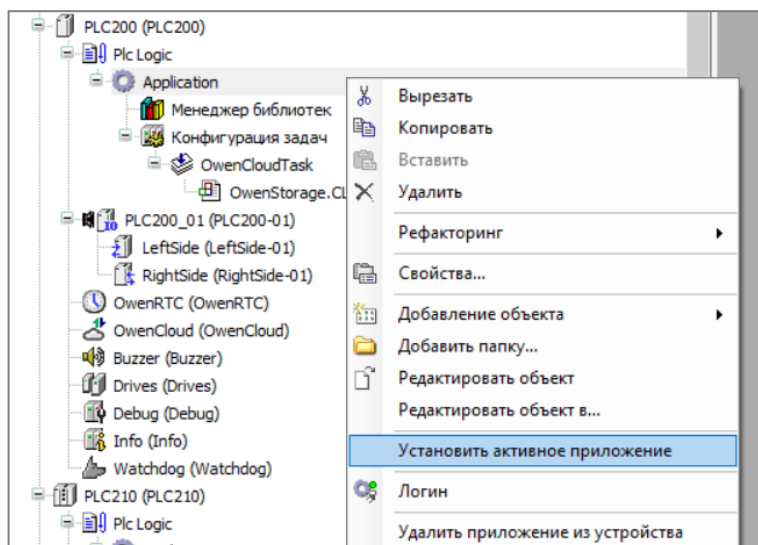


К счастью многих пользователей, в следующих версиях появилась возможность включить старое поведение с помощью галочки в меню **Инструменты – Опции – Интеллектуальный ввод**.

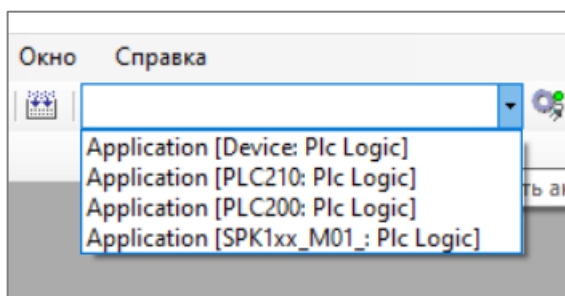


12. Выбор активного приложения

Если в проекте добавлено несколько контроллеров – то требуется переключаться между их приложениями для подключения к конкретному ПЛК. Раньше для этого нужно было нажать ПКМ на приложение и использовать команду **Установить активное приложение**.

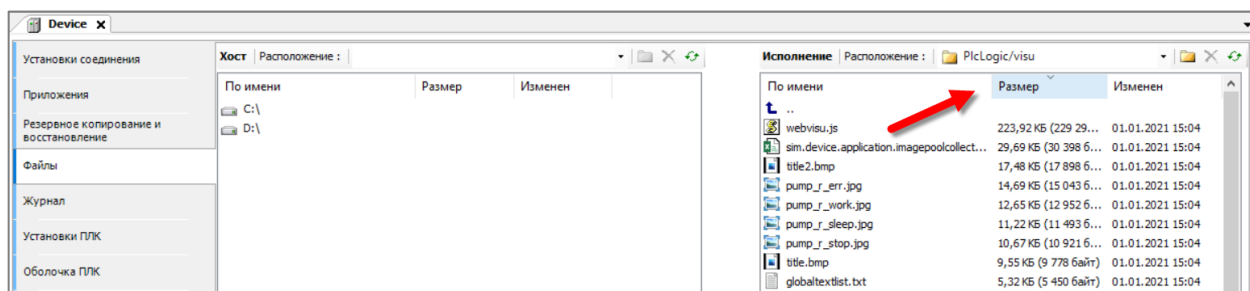


Теперь же это можно сделать с помощью выпадающего списка на панели инструментов.



13. Сортировка файлов на вкладке Device – Файлы

Теперь на вкладке **Device – Файлы** можно сортировать файлы по имени, размеру или дате изменения. Для этого нужно нажать на заголовок соответствующего столбца.



14. Библиотека CAA File – исправление ошибки в ФБ DirList

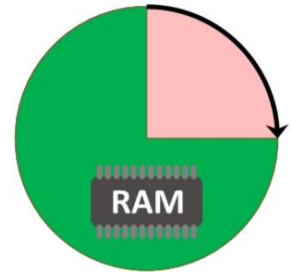
Теперь ФБ **DirList** возвращает дату последнего изменения не только файлов, но и директорий (ранее для директорий всегда возвращалось DT#1970-01-01-00:00:00).

15. Оптимизации

- Уменьшено время, затрачиваемое на установку пакетов (.package);
- Уменьшено время, затрачиваемое на открытие проекта;
- Среда программирования теперь использует на ~25% меньше оперативной памяти ПК.

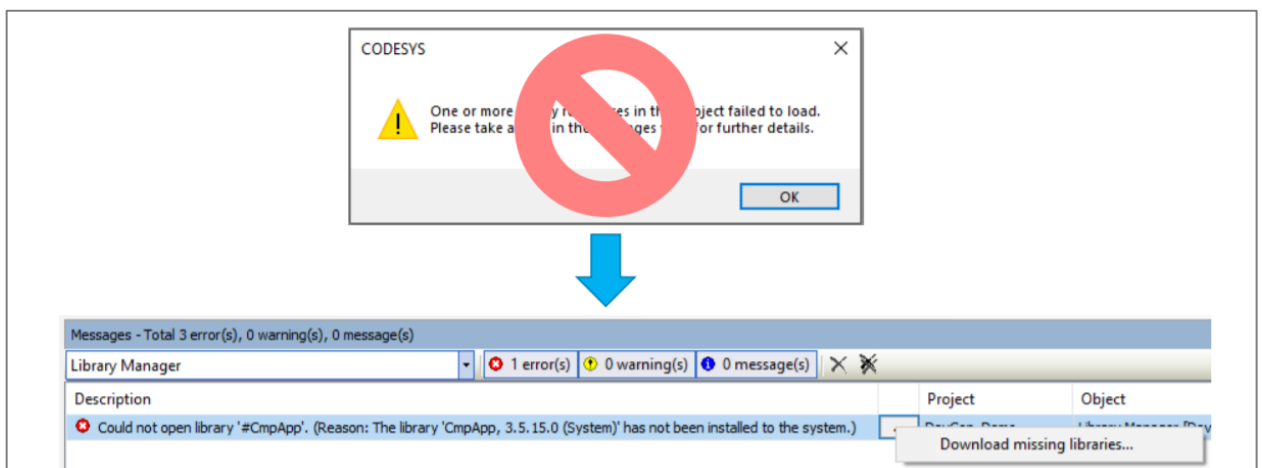
Memory consumption in CODESYS

- **Reduction of memory consumption for the compiler:**
 - In total about 50% less RAM needed
 - 25% less RAM for the CODESYS Development System
 - May save up to hundreds of megabytes of RAM
- **Further improvements of compile / generate code: Pending (SP17)**



16. Загрузка отсутствующих библиотек через панель сообщений

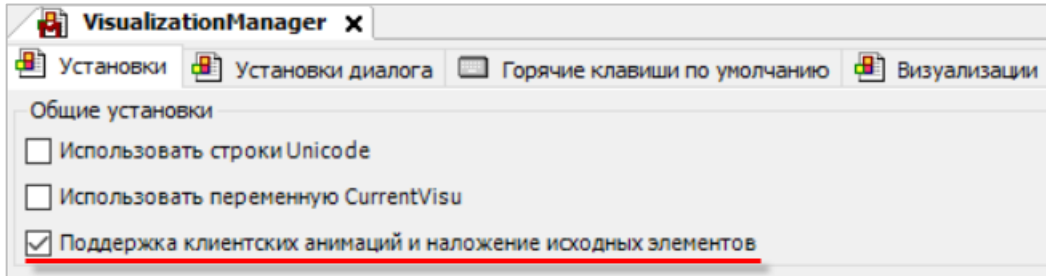
Теперь отсутствующие библиотеки можно загрузить не только через Менеджер библиотек, но и просто по нажатию на соответствующую ошибку на панели сообщений.



Визуализация

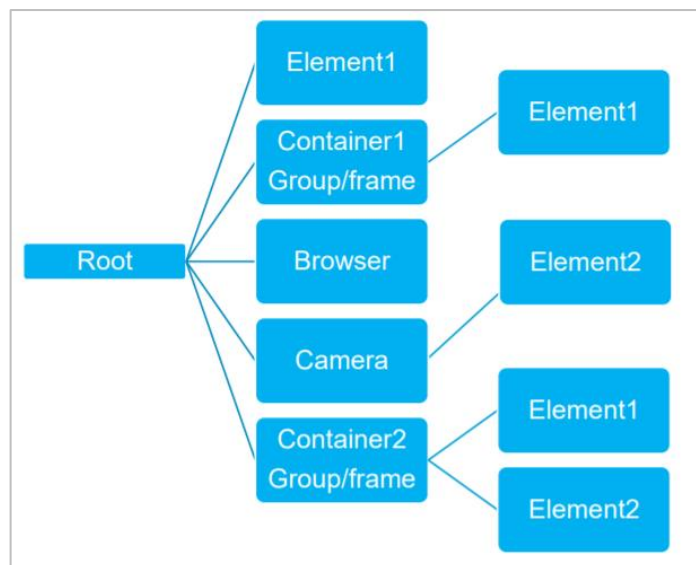
1. Поддержка технологии overlay

Начиная с V3.5 SP14 в CODESYS была добавлена preview-версия этой технологии, а полноценная ее поддержка появилась в V3.5 SP16. Для ее активации следует установить галочку в **Менеджере визуализации**:



Пока что технология поддерживается только для web-визуализации, и если в проекте есть таргет-визуализация – то галочка не будет отображаться. Поддержка технологии overlay для таргет-визуализации ожидается в следующих версиях CODESYS.

Технология меняет логику отрисовки визуализации, представляя каждый элемент как отдельный объект. Это, например, позволяет организовать наложение элементов и переключение их между слоями.



Ниже описываются возможности, которые дает overlay-технология, но для более наглядного знакомства с ними можно посмотреть [это видео](#).

После активации галочки у каждого элемента визуализации появляются параметры **Длительность анимации** и **Переместить на передний план**.

+	Отображение переменные	
+	Переменные состояний	
	Длительность анимации	1000
	Переместить на передний план	PLC_PRG.xSwitchToUpperLayer

Первый параметр имеет тип **INT** и определяет длительность анимации в миллисекундах.

Анимация включается при:

- перемещении элемента (с помощью параметров вкладки **Абсолютное перемещение**);
- открытии/закрытии диалога с помощью данного элемента;
- изменения видимости элемента;
- переключения экрана в фрейме (если параметр задан для фрейма).

Т.е. если раньше при изменении параметров вкладки **Абсолютное перемещение/Перемещение** элемент просто перерисовывался по новым координатам, то теперь он будет плавно перемещаться от старых координат к новым за заданное пользователем время. Этот же эффект «плавности» будет появляться при открытии/закрытии диалогов и т.д. Надо отметить, что обработка анимации происходит на стороне клиента и никак не связана с временем цикла задачи **VISU_TASK**.

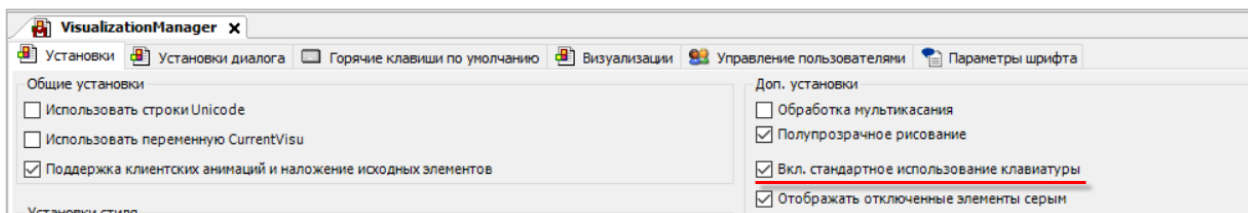
Параметр **Переместить на передний план** имеет тип **BOOL**. Если он принимает значение **TRUE**, то элемент перерисовывается в самом верхнем слое экрана (это будет заметно для элементов, наложенных друг на друга).

Также при активации галочки вкладки **Абсолютное перемещение** появляется у всех элементов визуализации – в том числе у графиков, таблиц и т.д.

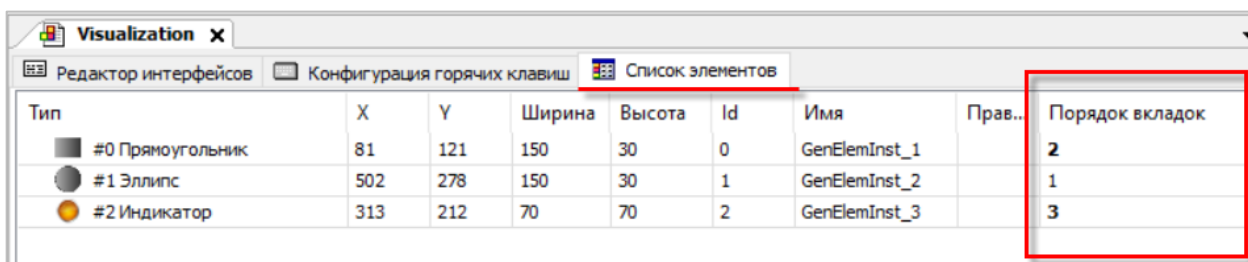
И последний функционал, который предоставляет overlay-технологии – возможность отображения в визуализации анимированных изображений в формате .gif и .svg.

2. Порядок переключения элементов с помощью клавиши Tab

Если в **Менеджере визуализации** установить галочку **Вкл. стандартное использование клавиатуры**, то можно будет переключать выбранный для взаимодействия элемент визуализации с помощью клавиши Tab.

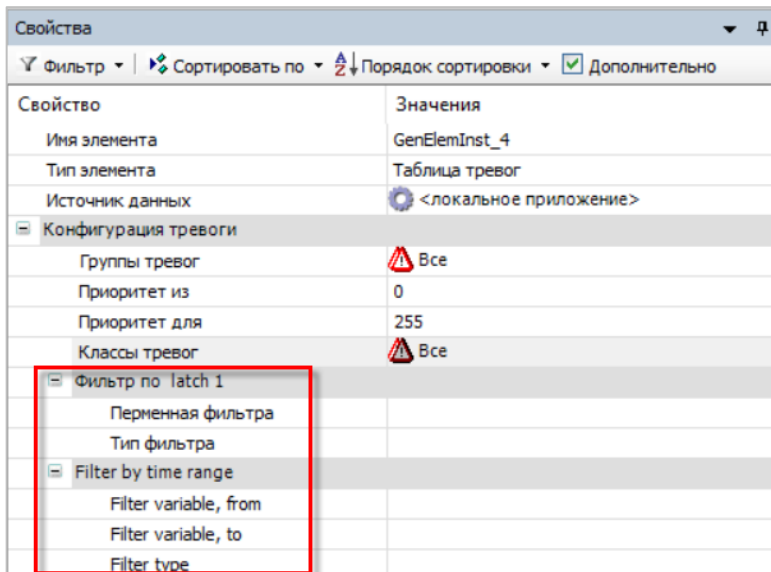


Эта возможность появилась уже давно, но раньше порядок переключения элементов зависел от их внутренних идентификаторов и, в сущности, был произвольным. В свежих версиях CODESYS появилась возможность задать порядок переключения элементов в редакторе экрана визуализации на вкладке **Список элементов** в столбце **Порядок вкладок**.



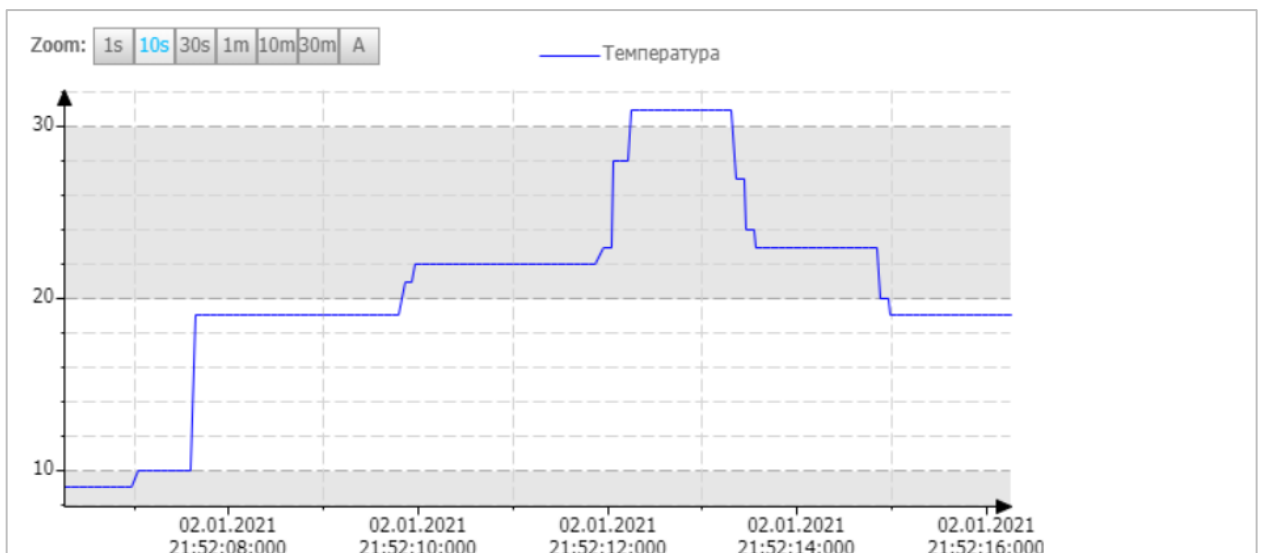
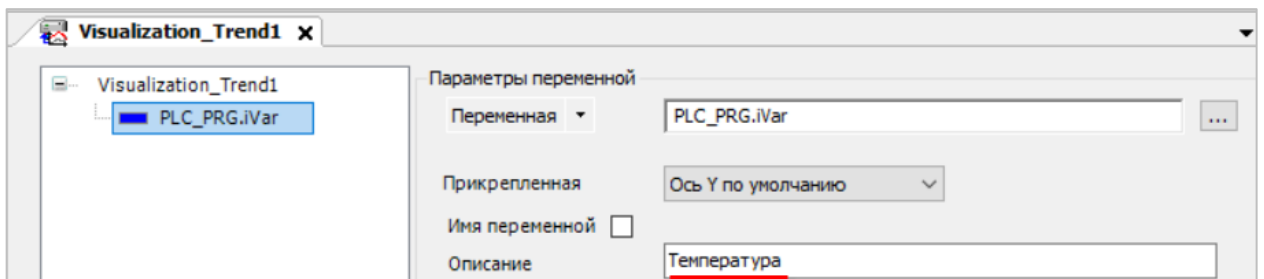
3. Сортировка сообщений в таблице и баннере тревог

С помощью новых параметров таблицы и баннера тревог можно сортировать сообщения по времени появления или значению первой latch-переменной. Более подробная информация приведена в [справке](#) и [примере](#).



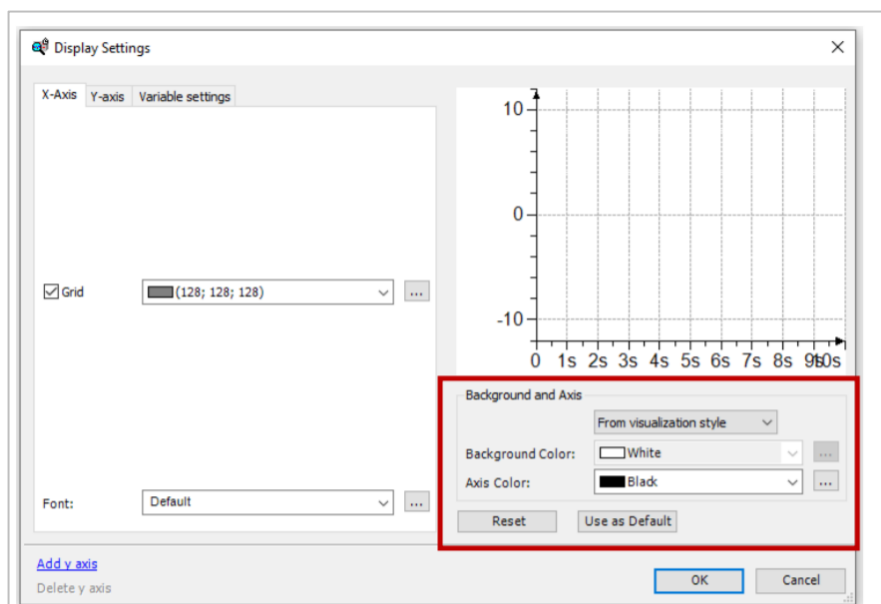
4. Поддержка кириллицы в легенде тренда

Теперь в легенде тренда можно использовать кириллицу (ранее из-за ошибки CODESYS вместо кириллицы отображались «крокозябры»).



5. Дополнительные настройки внешнего вида для трассировки и тренда

Для элементов Трассировка и Тренд появилась возможность настроить цвет фона и осей.



6. Выбор списка текстов в выпадающем списке с помощью строковой переменной

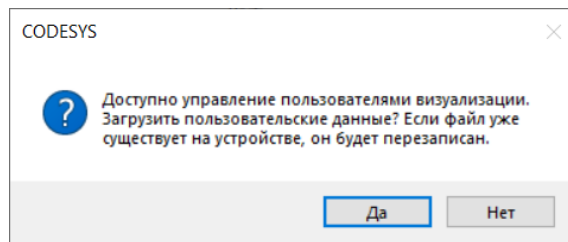
Теперь можно привязать к элементу Выпадающий список (Комбинированное окно, Combobox) не конкретный список текстов, а переменную типа **STRING**, которая будет содержать его название – таким образом, можно переключать списки текстов элемента в процессе работы проекта.

Свойство	Значения
Имя элемента	GenElemInst_8
Тип элемента	Комбинированное окно - Целочисленный
Позиция	
X	58
Y	29
Ширина	169
Высота	40
Переменная	PLC_PRG.iVar
<u>Список текстов</u>	<u>PLC_PRG.sTextListName</u>
Пул изображений	

7. Оптимизации

- уменьшено время переключения экранов визуализации в проектах с нагруженными визуализациями;
- повышена отзывчивость диалогов ввода (Numrad, Keypad) в тех случаях, когда они открываются из другого диалога;
- уменьшен объем памяти, занимаемый визуализацией;
- уменьшено потребление оперативной памяти элементом Тренд.

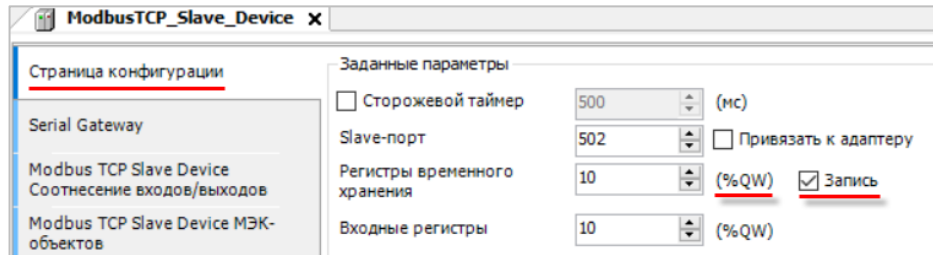
Теперь предложение обновить файлы управления пользователями появляется только в том случае, если они были изменены.



Modbus

1. Modbus Slave Device (Serial/TCP) – запись значений в holding-регистры из программы ПЛК

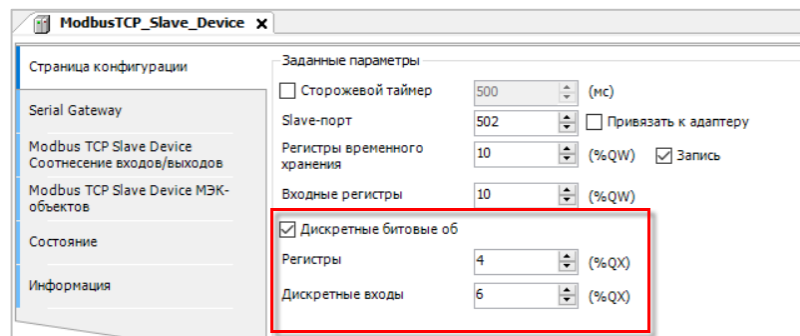
Исторически в CODESYS V3.5 в режиме Modbus Slave нельзя было менять значения holding-регистров (и coil'ов) из программы ПЛК – только со стороны мастер-устройства. В свежих версиях эта возможность, наконец, появилась – достаточно в настройках компонента поставить галочку **Запись**.



Необходимо отметить, что при активации этой галочки значения holding-регистров нельзя присваивать в каналах компонента с помощью команды **Отладка – Запись значения** – то есть в режиме онлайн-отладки вам потребуется изменить значения привязанной к каналу переменной (в POU, где объявлена эта переменная, или через вкладку **Вид – Просмотр**).

2. Modbus Slave Device (Serial/TCP) – независимые области памяти для бит

Исторически в CODESYS V3.5 в режиме Modbus Slave область памяти Coils накладывалась на область Holding Registers, а область Discrete Inputs – на область Input Registers. Теперь при активации галочки **Дискретные битовые области** эти области становятся независимыми и пользователь может указать их размер в битах. То есть теперь coil с адресом 0 не будет соответствовать нулевому биту holding-регистра с адресом 0 – он будет размещен в своей области памяти.



Переменная	Соотнесение	Канал	Адрес	Тип	Описание
		Регистры временного хранения	%QW0	ARRAY [0..9] OF WORD	
		Входные регистры	%QW10	ARRAY [0..9] OF WORD	
		Регистры	%QB40	ARRAY [0..0] OF BYTE	Coils
		Регистры[0]	%QX40	BYTE	
		Bit0	%QX40.0	BOOL	
		Bit1	%QX40.1	BOOL	
		Bit2	%QX40.2	BOOL	
		Bit3	%QX40.3	BOOL	
		Дискретные входы	%QB41	ARRAY [0..0] OF BYTE	Discrete Inputs
		Дискретные входы[0]	%QB41	BYTE	
		Bit0	%QX41.0	BOOL	
		Bit1	%QX41.1	BOOL	
		Bit2	%QX41.2	BOOL	
		Bit3	%QX41.3	BOOL	
		Bit4	%QX41.4	BOOL	
		Bit5	%QX41.5	BOOL	

Тут надо упомянуть о двух багах, появившихся в новых версиях компонента Modbus TCP Slave Device и связанных с битами.

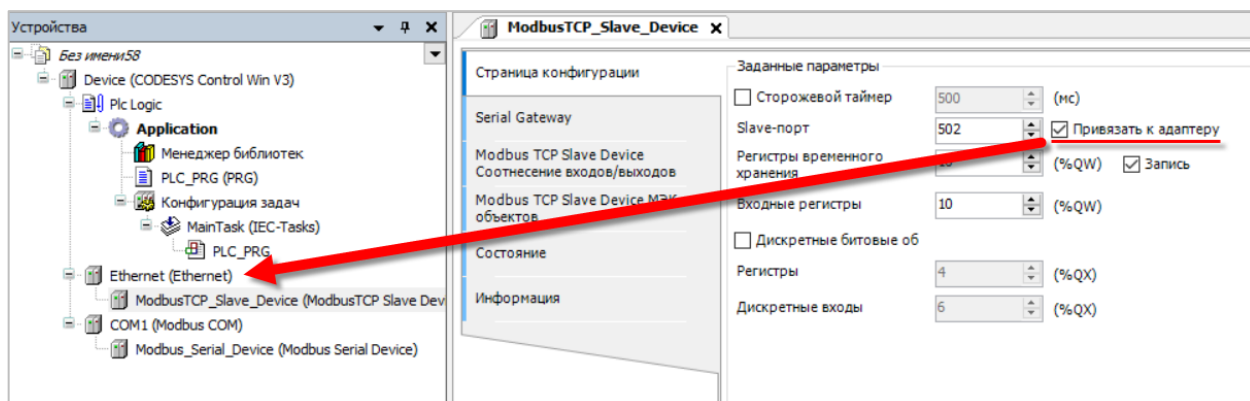
Первый баг связан с тем, что изменился порядок байт для хранения битов (независимо от состояния галочки **Дискретные битовые области**). То есть если мастер-устройство запрашивает состояние 0-го бита – то фактически будет возвращено состояние 8-го бита (и наоборот).

Второй баг заключается в том, что ПЛК теперь корректно обрабатывает запросы на чтение бит только в том случае, если начальный адрес в запросе равен адресу начального бита в байте (т.е. 0, 7, 15 и т.д.). При других начальных адресах слэйв возвращает условно-произвольный набор бит.

Исправление багов ожидается в следующих версиях CODESYS.

3. Modbus TCP Slave Device – возможность привязки к конкретному адаптеру

Ранее компонент Modbus TCP Slave Device всегда автоматически привязывался (bind) к интерфейсу 0.0.0.0 (INADDR_ANY) – то есть слэйв был доступен по любому из интерфейсов ПЛК. Теперь в настройках компонента появилась галочка **Привязать к адаптеру**. При ее активации слэйв привязывается к интерфейсу, который выбран в родительском для слэйва компоненте Ethernet.



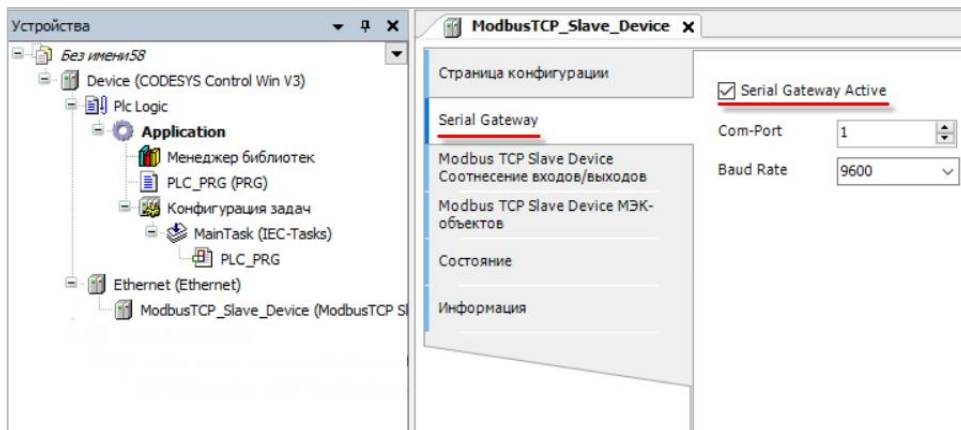
4. Modbus TCP Slave Device – увеличение допустимого количества одновременно подключенных клиентов

Ранее у слэйва было ограничение на 32 одновременно подключенных клиента, в свежих версиях их число увеличено до 64.

5. Modbus TCP Slave Device – функционал шлюза RTU/TCP (Serial Gateway)

Теперь компонент Modbus TCP Slave Device содержит вкладку **Serial Gateway**, которая позволяет настроить ПЛК как шлюз Modbus RTU/Modbus TCP – запросы, присланные Modbus TCP Master'ом, будут преобразованы в запросы Modbus RTU и отправлены в выбранный на вкладке COM-порт (соответственно, этот COM-порт не должен использоваться в других компонентах проекта). Ответы от Modbus RTU Slave'ов будут преобразованы в Modbus TCP и отправлены обратно мастер-устройству.

Обратите внимание, что если галочка **Serial Gateway Active** не установлена, то Modbus TCP Slave Device отвечает только на запросы с Unit ID = **255** и **0**.



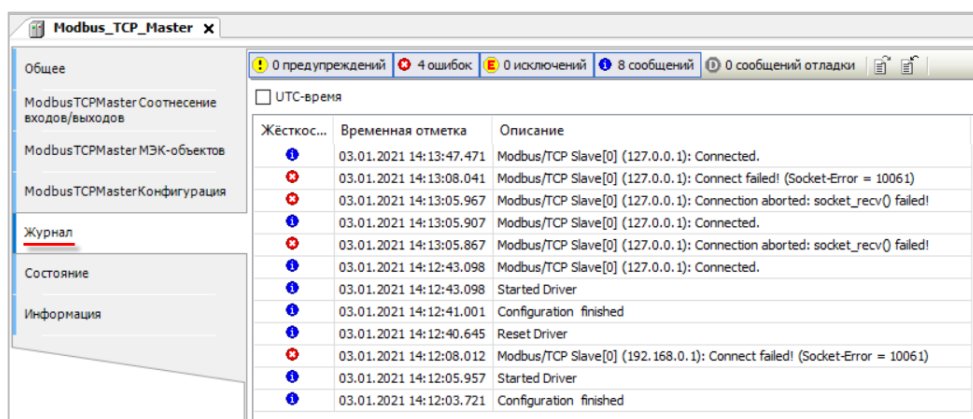
6. Все компоненты – улучшение возможностей диагностики

Серьезно расширен функционал диагностики всех modbus-компонентов.

Теперь если в каком-то из свернутых дочерних компонентов произошла ошибка – то у иконки родительского элемента в правом нижнем углу будет отображаться соответствующая пиктограмма.



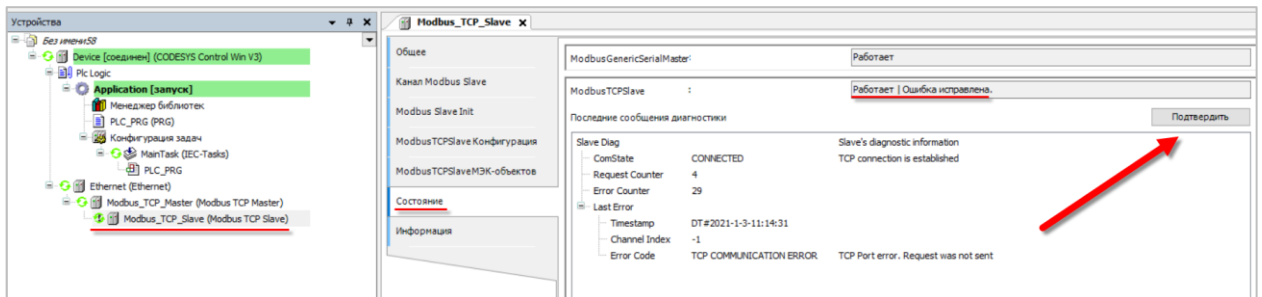
У компонента Modbus TCP Master появилась вкладка **Журнал**, на которой отображается лог подключений/потерь связи со slave-устройствами.



У компонентов Modbus TCP Slave и Modbus Slave COM Port на вкладке **Состояние** теперь отображается информация диагностики:

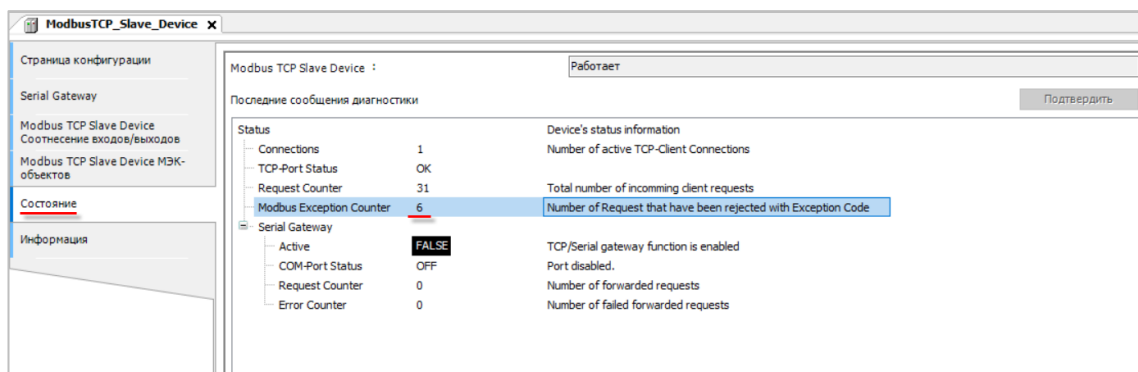
- состояние соединения (для Modbus TCP Slave);
- счетчик запросов, отправленных слэйву;
- счетчик ответов с кодом ошибки Modbus, полученных от слэйва;
- информация о последней ошибке – метка времени, индекс канала запроса (если ошибка не связана с каналом, то отображается -1) и код ошибки (например, ILLEGAL FUNCTION).

Кроме того, теперь при возникновении и последующем исчезновении ошибки на иконке элемента продолжает отображаться бледный восклицательный знак. Для его исчезновения надо нажать кнопку **Подтвердить**.

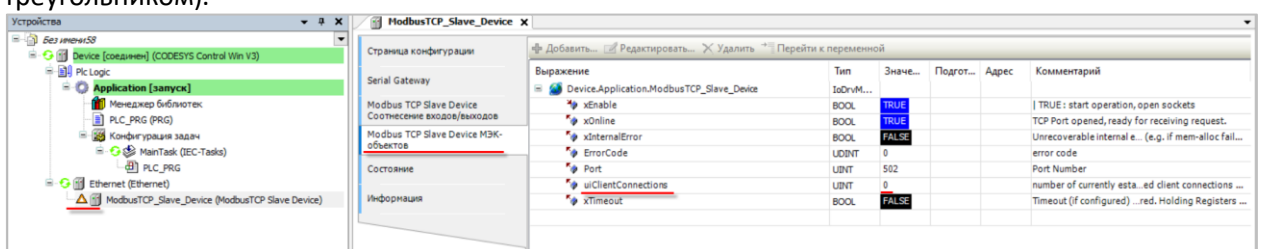


В компоненте Modbus TCP Slave Device на вкладке **Состояние** тоже отображается информация диагностики:

- число подключенных клиентов;
- статус TCP-порта;
- счетчик запросов, полученных компонентом;
- счетчик ответов с кодом ошибки Modbus, которые компонент отправил мастеру (к сожалению, без конкретизации кода ошибки);
- статус, счетчик запросов и счетчик ошибок для [Serial Gateway](#).

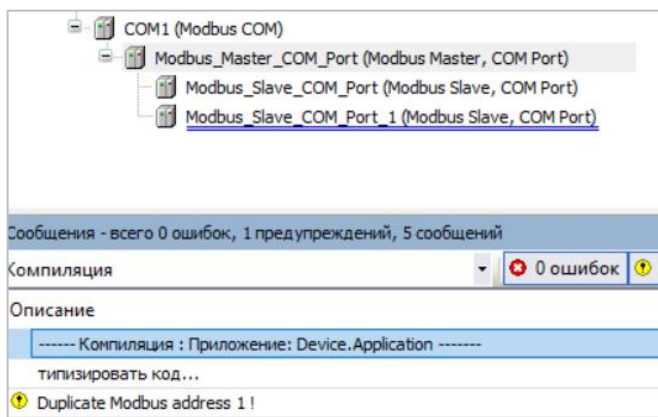


Стоит отметить, что в свежих версиях CODESYS если к Modbus TCP Slave Device не подключено ни одного клиента (Connections = 0), то это индицируется иконкой элемента (оранжевым треугольником).



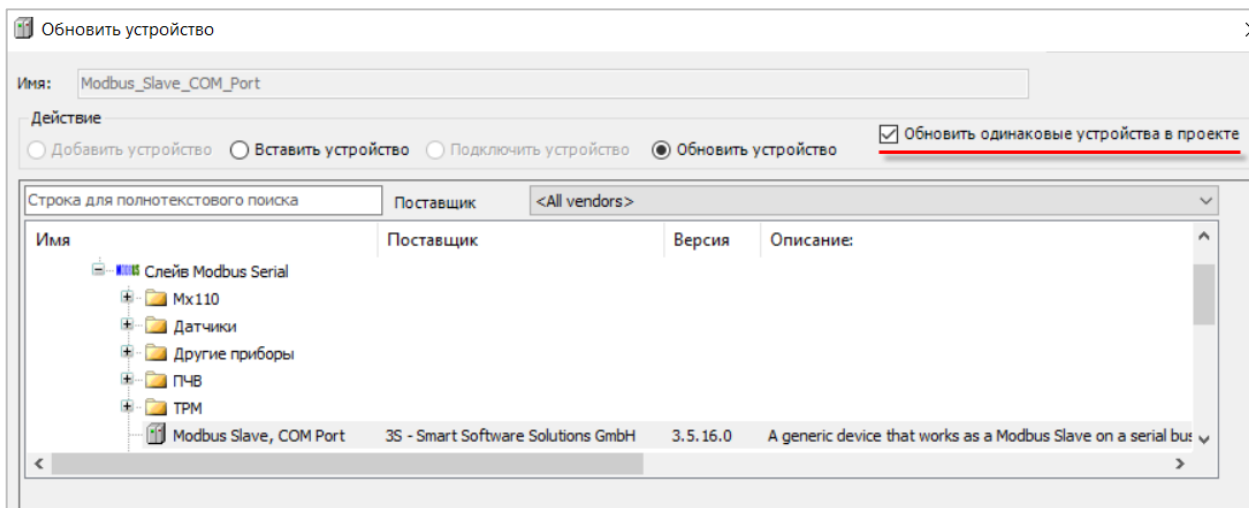
7. Modbus Slave COM Port – возможность добавление слэйвов с совпадающими Slave ID

Ранее добавление нескольких компонентов Modbus Slave COM Port с совпадающими Slave ID приводило к ошибке компиляции, теперь – только к предупреждению. Это может оказаться полезным в тех случаях, когда требуется создать для слэйва более 100 каналов опроса (такое ограничение установлено на один компонент) или, например, при использовании шаблонов опроса, в которых не хватает нескольких нужных параметров – теперь можно добавить их в рамках отдельного компонента.



8. Все компоненты – возможность группового обновления версий компонентов

Теперь при обновлении версии компонента (команда контекстного меню **Обновить устройство**) можно обновить сразу все компоненты проекта до выбранной версии с помощью галочки **Обновить одинаковые устройства в проекте**.

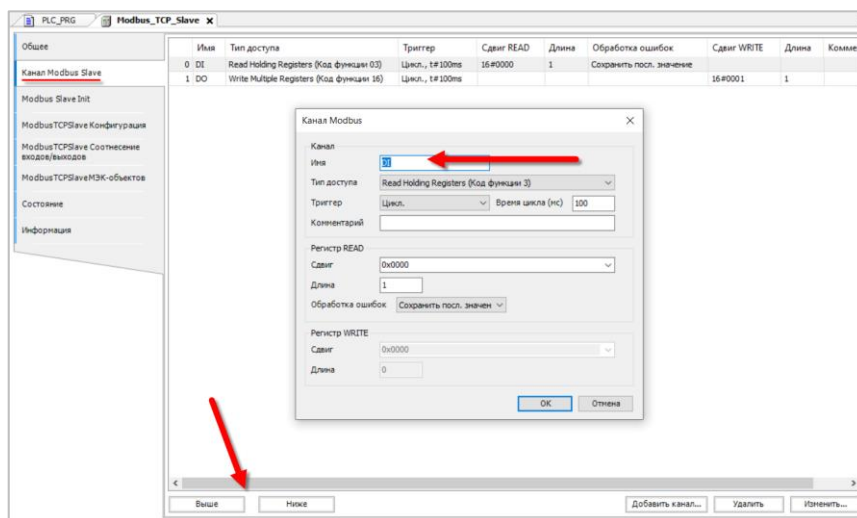


9. Modbus Master (Serial/TCP) – оптимизации в драйвере

Произведена оптимизация драйверов Modbus Master, которая позволила ускорить копирование данных между каналами опроса и привязанными к ним переменными.

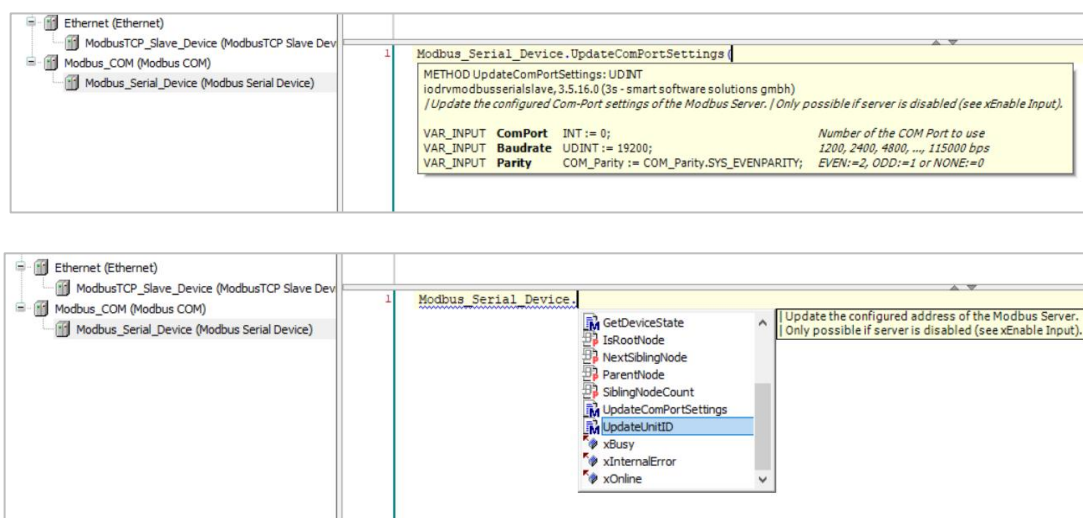
10. Modbus Master (Serial/TCP) – улучшения в механизме редактирования каналов

Теперь после создания канала опроса можно отредактировать его имя и менять порядок каналов в компоненте с помощью кнопок **Выше/Ниже**.



11. Modbus Slave Device (Serial/TCP) – дополнительные методы и переменные

У компонента Modbus Slave Device добавлены методы **UpdateComPortSettings** и **UpdateUnitID**, которые позволяют изменить настройки COM-порта и адрес слэйва в процессе работы ПЛК. Перед вызовом методов необходимо приостановить работу компонента с помощью присвоения его входной переменной **xEnable** значения **FALSE**. После вызова методов компонент можно снова запустить в работу, присвоив переменной значение **TRUE**. Обратите внимание, что при перезагрузке контроллера настройки компонента снова будут взяты из дерева проекта – это нужно учесть в логике программы, организовав в коде перенастройку компонента после загрузки контроллера.



У обоих компонентов добавлены новые выходные переменные:

- **xOnline (BOOL)** – флаг активности компонента (получения запросов от мастера);
- **ErrorCode (UDINT)** – код ошибки компонента;
- **Port (UINT)**; только для компонента Modbus TCP Slave Device) – номер TCP-порта компонента;
- **xTimeout (BOOL)**; только для компонента Modbus TCP Slave Device) – флаг срабатывания таймаута отсутствия запросов от клиента (при установке в настройках компонента соответствующей галочки), в этот момент происходит обнуление holding-регистров и coil'ов.

12. Исправление ошибок

- в компоненте Modbus COM опять можно выбрать скорость обмена 38400 (в нескольких прошлых сервис-паках она отсутствовала);
- в компонентах Modbus Serial Device и Modbus TCP Slave Device исправлена ошибка в обработке функции 05 (Write Single Coil);
- в компоненте Modbus TCP Slave исправлена ошибка обработки выходной переменной компонента **uiChannelIndex** – ранее при ошибках (xError = TRUE) она всегда принимала значение -1 (а теперь, как и должно быть – в нее записывается номер канала, в котором произошла ошибка).