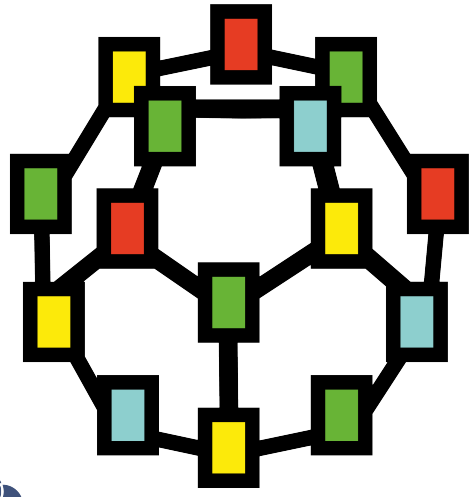
Библиотека paModbus



ПромАвтоматика

Руководство пользователя

06.2024
версия 1.1

Содержание

Используемые термины и сокращения

ПК – персональный компьютер.

ПЛК – программируемый логический контроллер.

ШИМ – широтно-импульсная модуляция.

OPC UA (Open Platform Communications, Unified Architecture) – протокол для обмена данными с ПЛК и для управления ими.

SQL (**Structured Query Language**) – язык программирования для хранения и обработки информации в реляционной базе данных.

Введение

Настоящее руководство описывает настройку обмена данными по протоколам **Modbus RTU** и **Modbus TCP** для контроллеров ОВЕН, программируемых в среде **Полигон**. Подразумевается, что читатель обладает базовыми навыками работы с **Полигон**, поэтому общие вопросы (например, создание и загрузка проектов) в данном документе не рассматриваются – они подробно описаны в документах Руководство по программированию. Библиотека paCore и Быстрый старт.

Настройка обмена по протоколу **Modbus** в среде **Полигон** осуществляется с помощью функциональных блоков из библиотеки ***paModbus***.

Примеры в документе актуальны для среды **Полигон** версии **1912** и для библиотеки **paModbus** версии **107** и выше.

1 Общие сведения

1.1 Основные сведения об интерфейсе RS-485

Интерфейс RS-485 подразумевает использование исключительно топологии типа «шина» (топологии типа «звезда» и «кольцо» не описаны в стандарте).

В сети может присутствовать только одно Master-устройство, которое отсылает запросы и принимает ответы от подчиненных Slave-устройств. Slave-устройства не могут являться инициаторами обмена.

Число Slave-устройств в сегменте сети не должно превышать **32**. Сегменты могут быть соединены повторителями (например, ОВЕН АС5), но следует учитывать, что так как опрос всех устройств происходит последовательно, то время одного полного цикла опроса может значительно увеличиться. Общее ограничение числа Slave-устройств в сети для протокола **Modbus** – **247**.

На первом и последнем устройстве в сети рекомендуется устанавливать согласующий резистор (терминатор) с сопротивлением **120 Ом**.

Для линий связи RS-485 необходимо использовать экранированный кабель с витой парой, предназначенный для промышленного интерфейса RS-485 с волновым сопротивлением **120 Ом** (например, КИПЭВ). Экран кабеля должен быть соединен с функциональной землей **только в одной точке**.

Некоторые устройства имеют встроенные резисторы подтяжки интерфейса RS-485. Информация и рекомендации по их использованию приведены в руководстве по эксплуатации на соответствующие приборы.

1.2 Основные сведения о протоколе Modbus

Modbus – открытый коммуникационный протокол, основанный на архитектуре Master-Slave (ведущий-ведомый). Спецификация протокола доступна на сайте Modbus Organization.

Master (мастер, ведущее устройство) является инициатором обмена и может считывать и записывать данные в Slave-устройства.

Slave (слэйв, подчиненное устройство) отвечает на запросы Master-устройства, но не может самостоятельно инициировать обмен.

Существуют две основные реализации протокола:

- **Modbus Serial** для передачи данных с использованием последовательных интерфейсов RS-232/RS-485
- **Modbus TCP** для передачи данных через сети TCP/IP.

Если для работы используют протокол **Modbus RTU** с интерфейсом **RS-232/RS-485**, то в сети может присутствовать **только одно** Master-устройство и несколько (до **247**) Slave-устройств. Адрес **0** используется для широковещательной рассылки (команд записи, которую получат все Slave-устройства).

У протокола **Modbus TCP** нет явного ограничения на количество Master- и Slave-устройств. Кроме того, устройство может одновременно выполнять функции Master и Slave.

Запрос Master-устройства к Slave-устройству содержит следующие данные:

- **Slave ID** (адрес Slave-устройства);
- **Код функции**, применяемой к Slave-устройству;
- **Данные** – адрес первого регистра и их количество (в случае записи – также записываемые значения);
- **Контрольная сумма**.

Ответ Slave-устройства имеет схожую структуру.

Запрос Master-устройства представляет собой обращение к одной из **областей памяти** Slave-устройства с помощью определенной функции. Область памяти характеризуется типом хранящихся в ней значений (биты/регистры) и типом доступа (только чтение/чтение и запись). Стандарт Modbus определяет четыре области памяти.

Таблица 1.1 – Области данных протокола Modbus

Область данных	Обозначение	Тип данных	Тип доступа
Coils (Регистры флагов)	0x	BOOL	Чтение/запись
Discrete Inputs (Дискретные входы)	1x	BOOL	Только чтение
Input Registers (Регистры ввода)	3x	WORD	Только чтение
Holding Registers (Регистры хранения)	4x	WORD	Чтение/запись

Каждая область памяти состоит из определенного (зависящего от конкретного устройства) количества ячеек. Каждая ячейка имеет уникальный адрес. Для конфигурируемых устройств (таких как ТРМ, ПЧВ и т. д.) производитель предоставляет **карту регистров**, в которой содержится информация об адресах и типах параметров устройства. Для программируемых устройств пользователь формирует такую карту самостоятельно с помощью среды разработки. Существуют устройства, в которых сочетаются оба рассмотренных случая – у их карты регистров есть фиксированная часть, которую пользователь может дополнить в соответствии со своей задачей (но адреса ячеек не должны пересекаться).

ПРИМЕЧАНИЕ
В некоторых устройствах области памяти наложены друг на друга (например, **0x** и **4x**) – т. е. к одним и тем же ячейкам памяти можно обращаться разными функциями.

Функция определяет операцию (чтение/запись) и область памяти, в которой эта операция будет выполняться. Ниже приведен список наиболее часто используемых функций.

Таблица 1.2 – Основные функции протокола Modbus

Код функции	Имя функции	Команда
1 (0x01)	Read Coil Status	Чтение значений из регистров флагов
2 (0x02)	Read Discrete Inputs	Чтение значений из дискретных входов
3 (0x03)	Read Holding Registers	Чтение значений из регистров хранения
4 (0x04)	Read Input Registers	Чтение значений из регистров ввода
5 (0x05)	Write Single Coil	Запись значения в один регистр флага
6 (0x06)	Write Single Register	Запись значения в один регистр хранения
15 (0x0F)	Write Multiple Coils	Запись значений в несколько регистров флагов
16 (0x10)	Write Multiple Registers	Запись значений в несколько регистров хранения

ПРИМЕЧАНИЕ
Нельзя смешивать понятия области памяти и функции. У начинающих пользователей часто возникают проблемы при работе с **Input** и **Holding** регистрами, поскольку область памяти **Holding** регистров имеет обозначение **4x**, а функция чтения **Holding** регистров – **0x03** (может интуитивно показаться, что идентификатор области памяти и код функции должны совпадать – но на практике это не так).

Запрос к Slave-устройству может быть **одиночным** или **групповым**.

При одиночном запросе Master-устройство считывает каждый из параметров Slave-устройства отдельной командой. При групповом опросе Master-устройство считывает одной командой сразу несколько параметров, адреса которых в карте регистров расположены строго последовательно и не имеют разрывов.

Групповой опрос позволяет уменьшить трафик в сети и время, затрачиваемое на опрос устройства, но в некоторых случаях его нельзя применять (или можно, но с ограничениями) из-за индивидуальных особенностей устройства.

2 Функциональные блоки настройки интерфейсов

2.1 Функциональные блоки настройки COM-портов

В данном разделе описаны блоки настройки COM-портов контроллеров ОВЕН ПЛК210 из библиотеки ***paOwenIO***.

2.1.1 Порт RS-485 (210-RS485)

Блок ***210-RS485*** предназначен для работы с портами ПЛК210 стандарта RS-485.

Таблица 2.1 – Назначение входов и выходов 210-RS485

Элемент	Описание
Входы (константные)	
port	Порт
spd	Скорость в бодах – **1200**, **2400**, **4800**, **9600**, **19200**, **38400**, **57600**, **115200**
par	Контроль четности: **0** – нет; **1** – нечетный; **3** – четный
stb	Количество стоповых бит – **7** или **8**
dtb	Количество бит данных – **1** или **2**
term	Включение/выключение терминальных резисторов
Выходы	
cnc	Связь с блоком протокола
stat	Статус: **1** - соединение установлено; **-1** - не удалось открыть указанный интерфейс; **-2** - отсутствует соединение; **-6** - ошибка настройки интерфейса
rcnt	Количество полученных байт
wcnt	Количество отправленных байт
diag	Диагностический – счетчик разности между количеством ошибок и принятыми (не может быть меньше нуля)

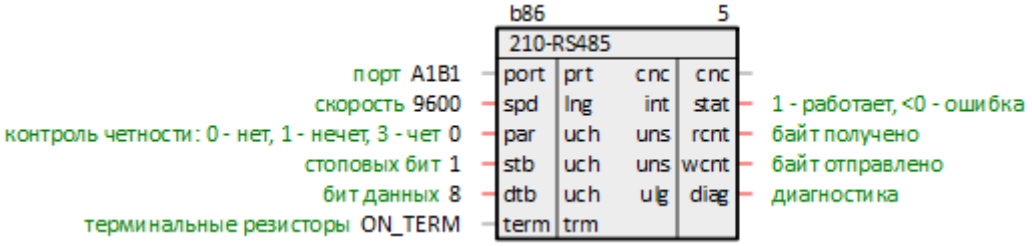


Рисунок 2.1 – Порт RS-485 (210-RS485)

2.1.2 Порт RS-232 (210-RS232)

Блок ***210-RS232*** предназначен для работы с портом ПЛК210 стандарта RS-232.

Таблица 2.2 – Назначение входов и выходов 210-RS232

Элемент	Описание
Входы (константные)	
spd	Скорость в бодах – **1200**, **2400**, **4800**, **9600**, **19200**, **38400**, **57600**, **115200**

Продолжение таблицы 2.2

Элемент	Описание
par	Контроль четности: **0** – нет; **1** – нечетный; **3** – четный
stb	Количество стоповых бит – **7** или **8**
dtb	Количество бит данных – **1** или **2**
Выходы	
cnc	Связь с блоком протокола
stat	Статус: **1** - соединение установлено; **-1** - не удалось открыть указанный интерфейс; **-2** - отсутствует соединение
rcnt	Количество полученных байт
wcnt	Количество отправленных байт
diag	Диагностический – счетчик разности между количеством ошибок и принятыми (не может быть меньше нуля)

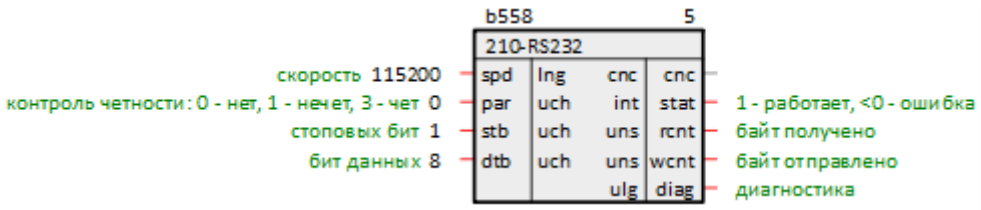


Рисунок 2.2 – Порт RS-232 (210-RS232)

2.2 Функциональные блоки настройки TCP-соединений

В данном разделе описаны блоки настройки TCP-соединений ПЛК210 из библиотеки ***paCore***.

2.2.1 TCP/IP-сервер (TcpIpSrA)

Блок ***TcpIpSrA*** представляет собой TCP/IP-сервер для обеспечения работы протоколов (например, Modbus TCP Slave). Сервер поддерживает одновременно не более **20** подключений.

Таблица 2.3 – Назначение входов и выходов TcpIpSrA

Элемент	Описание
Входы (константные)	
prt	Локальный порт
lip	Локальный IP адрес
sdr	Сетевой стек, для ПЛК ОВЕН **"/"**
wait	Время до закрытия пустого канала, мс. При установке **0** – никогда
Выходы	
cnc	Связь с блоком протокола
stat	Статус: **0** – есть подключения; **>0** – нет подключений

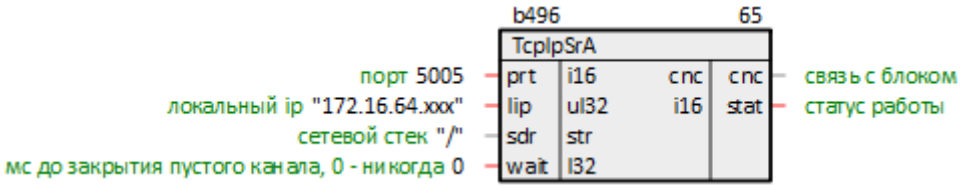


Рисунок 2.3 – TCP/IP-сервер (TcpIpSrA)

При настройке блока ***TcpIpSrA*** удобно использовать технологию SQL-запросов. Это позволяет изменять IP-адрес и порт в одном месте и использовать эти значения в разных частях проекта.

Запрос IP-адреса (prop_ip):
```
"<sql>SELECT value FROM blocks_prop WHERE indx=:module AND type="prop_ip"</sql>"
```

Запрос пользовательского свойства ***Пользовательское свойство 00*** (prop_0):
```
<sql> SELECT value FROM blocks_prop WHERE indx=:module AND type="prop_0"</sql>
```

2.2.2 TCP/IP-клиент (TcpIpCIA)

Блок ***TcpIpCIA*** представляет собой TCP/IP-клиент для обеспечения работы протоколов (например, Modbus TCP Master).

Так как работа блока занимает значительное время, может быть размещен только в ***Фоне***.

Таблица 2.4 – Назначение входов и выходов TcpIpCIA

Элемент	Описание
Входы (константные)	
lprt	Локальный порт
lip	Локальный IP адрес
sdr	Сетевой стек, для ПЛК ОВЕН **"/"**
rprt	Удаленный порт
ip	IP адрес удаленного сервера
Выходы	
cnc	Связь с блоком протокола
stat	Статус: **0** – есть связь с TCP/IP-сервером; **>0** – нет связи

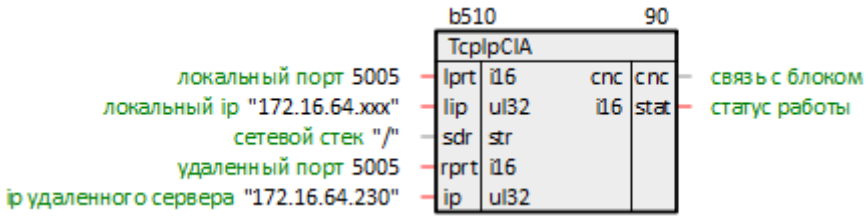


Рисунок 2.4 – TCP/IP-клиент (TcpIpCIA)

При настройке блока ***TcpIpCIA*** удобно использовать технологию SQL-запросов. Это позволяет изменять IP-адреса и порты в одном месте, и использовать эти значения в разных частях проекта.

Запрос IP-адреса (prop_ip):
```
"<sql>SELECT value FROM blocks_prop WHERE indx=:module AND type="prop_ip"</sql>"
```

Запрос пользовательского свойства ***Пользовательское свойство 00*** (prop_0):
```
<sql> SELECT value FROM blocks_prop WHERE indx=:module AND type="prop_0"</sql>
```

3 Библиотека paModbus

Библиотека ***paModbus*** содержит функциональные блоки для обмена по протоколам Modbus – блоки **Modbus Master** и **Modbus Slave**, а также блоки команд чтения и записи.

Для добавления библиотеки ***paModbus*** в проект следует:

1. Перейти в меню ***Окна/Проекты***. В появившемся окне отобразится текущий проект и добавленные библиотеки.

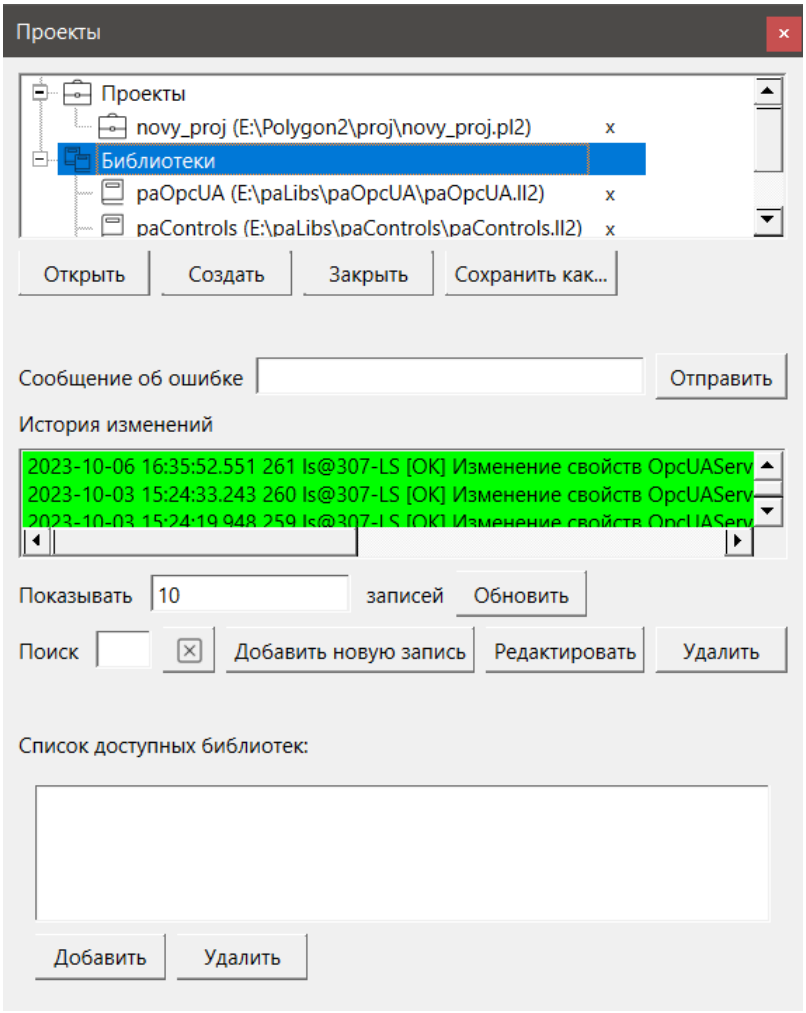


Рисунок 3.1 – Добавление библиотеки paModbus в проект

2. Нажать кнопку ***Открыть*** и перейти в папку с файлами библиотеки, которую необходимо добавить. Затем в выпадающем списке выбрать тип файла ***Библиотека Полигон 2 (*.lI2)***

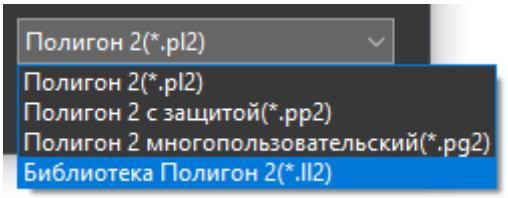


Рисунок 3.2 – Добавление библиотеки paModbus в проект

3. В окне появится файл библиотеки с расширением **.lI2**. Необходимо выбрать его и нажать **Открыть**.

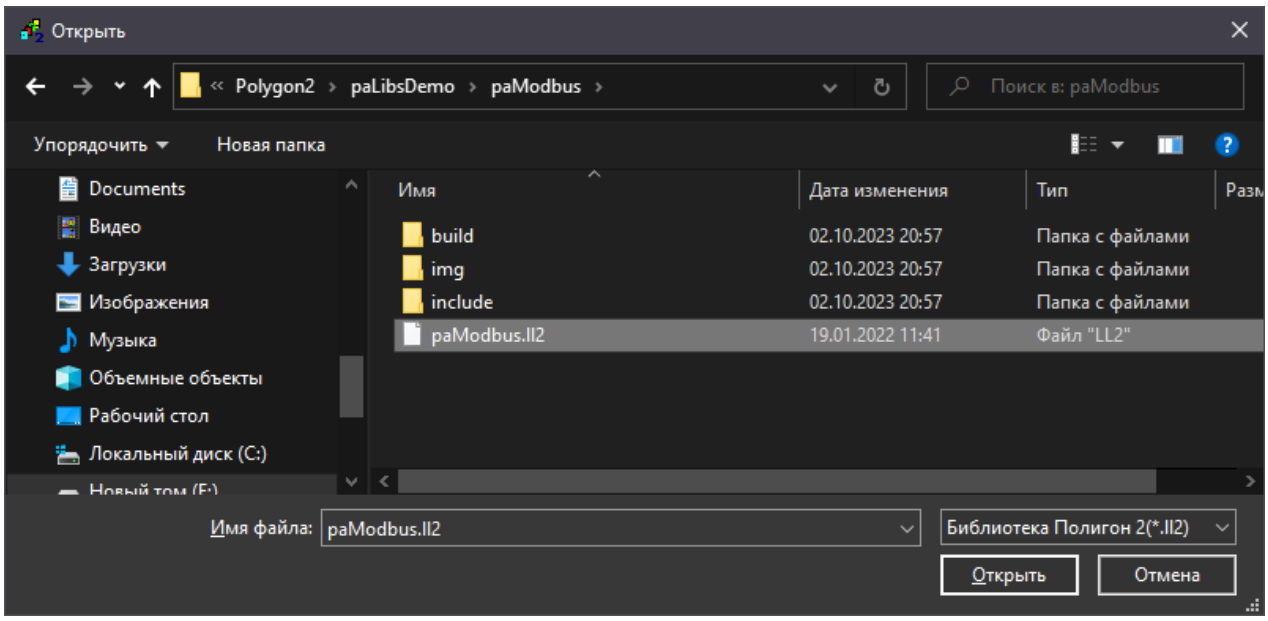


Рисунок 3.3 – Добавление библиотеки paModbus в проект

Добавленная библиотека отобразится в окне ***Проекты***.

3.1 Modbus RTU

3.1.1 Блок Modbus RTU Master

Блок ***Modbus RTU Master*** обеспечивает работу **Master**-устройства по протоколу **Modbus RTU**.

Так как работа блока занимает значительное время, его можно разместить только в **Фоне**.

Таблица 3.1 – Назначение входов и выходов Modbus RTU Master

Элемент	Описание
Входы	
cnc	Связь с блоком COM-порта (210-RS485, 210-RS232)
enbl	Разрешение работы
tmp	Время (в мс) между получением ответа от Slave-устройства и началом опроса следующего (константный)
wait	Время (в мс), в течение которого Master ожидает ответа Slave устройства (константный)
bo	Входы для связи с блоками записи (циклический)
Выходы	
itr	Связь с блоками чтения
sts	Статус работы: **0** – есть связь; **>0** – нет связи или Slave не отвечает (в младшем байте: **1** –Slave не отвечает, **2** –неверная контрольная сумма)

Команды, подключенные к блоку Master, опрашиваются последовательно с периодичностью **tmp**. Выключенные команды (вход **enb** блока команды, равный **0**) пропускаются.

ВНИМАНИЕ
Для правильной инициализации необходимо разместить в программе блок Master после блоков записи (см. свойства блоков ***Порядок***).

Чтение осуществляется циклически, запись – по изменению.

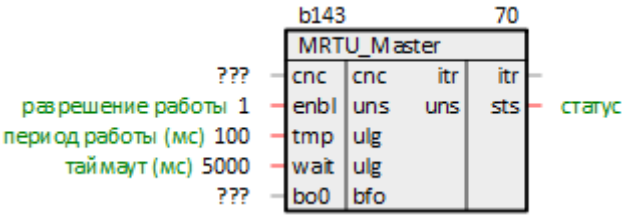


Рисунок 3.4 – Блок Modbus RTU Master

3.1.2 Блок Modbus RTU Slave

Блок ***Modbus RTU Slave*** обеспечивает работу Slave-устройства по протоколу **Modbus RTU**.

Так как работа блока занимает значительное время, его можно разместить только в **Фоне**.

Таблица 3.2 – Назначение входов и выходов Modbus RTU Slave

Элемент	Описание
Входы	
cnc	Связь с блоком COM-порта (210-RS485, 210-RS232)
enbl	Разрешение работы
tmp	Периодичность обработки запросов, мс (константный)
wait	Время ожидания запроса от Master, мс (константный)
bo	Входы для связи с блоками записи (циклический)
Выходы	
itr	Связь с блоками чтения
sts	Не используется

Slave обрабатывает команды, полученные от Master, с периодичностью **tmp**. Если при обработке команды не обнаружен соответствующий блок команды, то Slave посылает мастеру соответствующую ошибку:

- ***0x80 | команда*** – нет блока с соответствующим кодом команды, код команды – в блоке не определен запрашиваемый адрес (например, при опросе Holding Register команда *0x03*, тогда ошибка: *0x80 | 0x03 = 0x83*);
- ***0x23*** – неверная контрольная сумма.

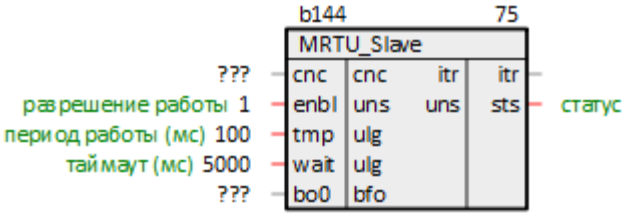


Рисунок 3.5 – Блок Modbus RTU Slave

3.2 Modbus TCP

3.2.1 Блок Modbus TCP Master

Блок ***Modbus TCP Master*** обеспечивает работу Master-устройства по протоколу **Modbus TCP**.

Так как работа блока занимает значительное время, его можно разместить только в **Фоне**.

Таблица 3.3 – Назначение входов и выходов Modbus TCP Master

Элемент	Описание
Входы	
cnc	Связь с блоком TcpIpCIA
enbl	Разрешение работы
tmp	Время (в мс) между получением ответа от Slave-устройства и началом опроса следующего (константный)

Продолжение таблицы 3.3

Элемент	Описание
wait	Время (в мс), в течение которого Master ожидает ответа Slave устройства (константный)
bo	Входы для связи с блоками записи (циклический)
Выходы	
itr	Связь с блоками чтения
sts	Статус работы: **0** – есть связь с сервером TCP/IP **>0** – нет связи или Slave не отвечает

Команды, подключенные к блоку Master, опрашиваются последовательно с периодичностью **tmp**. Выключенные команды (вход **enb** блока команды, равный **0**) пропускаются.

Чтение осуществляется циклически, запись – по изменению.

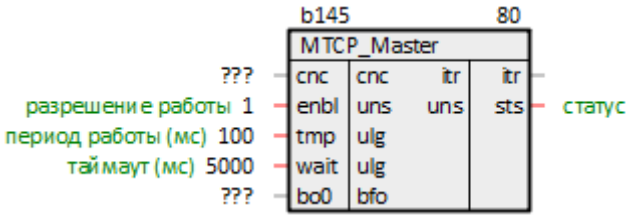


Рисунок 3.6 – Блок Modbus TCP Master

3.2.2 Блок Modbus TCP Slave

Блок ***Modbus TCP Slave*** обеспечивает работу Slave-устройства по протоколу **Modbus TCP**.

Так как работа блока занимает значительное время, его можно разместить только в **Фоне**.

Таблица 3.4 – Назначение входов и выходов Modbus TCP Slave

Элемент	Описание
Входы (константные)	
cnc	Связь с блоком TcpIpSrA
enbl	Разрешение работы
tmp	Периодичность обработки запросов, мс (константный)
wait	Время ожидания запроса от Master, мс (константный)
bo	Входы для связи с блоками записи (циклический)
Выходы	
itr	Связь с блоками чтения
sts	Статус работы: **0** – есть связь с Master; **>0** – нет связи или нет запросов от Master

Slave обрабатывает команды, полученные от Master, с периодичностью **tmp**. Если при обработке команды не обнаружен соответствующий блок команды, то Slave посылает мастеру одну из ошибок:

- **1** – нет блока с соответствующим кодом команды;
- **2** – в блоке не определен запрашиваемый адрес.

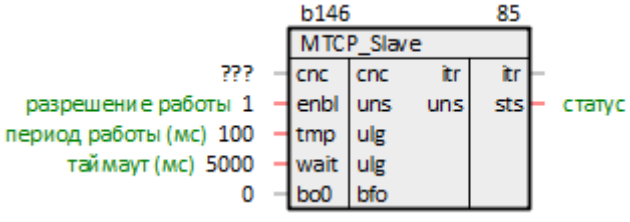


Рисунок 3.7 – Блок Modbus TCP Slave

3.3 Команды чтения

В данном разделе приведено описание блоков библиотеки ***paModbus***, которые реализуют команды протокола **Modbus** на чтение параметров.

Таблица 3.5 – paModbus. Команды чтения

Код функции	Имя функции	Команда	Блоки paModbus	
			Подключение к *Modbus Master*	**Подключение к *Modbus Slave***
1 (0x01)	Read Coil Status	Чтение значений из регистров флагов	ModbusCoilIn	ModbusCoilOut ModbusCoilsOut
2 (0x02)	Read Discrete Inputs	Чтение значений из дискретных входов	ModbusDInputIn	-
3 (0x03)	Read Holding Registers	Чтение значений из регистров хранения	ModbusRegIn ModbusFltIn OwenFltIn	ModbusRegOut ModbusRegsOut ModbusFltOut
4 (0x04)	Read Input Registers	Чтение значений из регистров ввода	ModbusAInputIn ModbusFltIn OwenFltIn	-

3.3.1 Команда чтения флагов 0x01 (ModbusCoilIn)

Блок **ModbusCoilIn** отправляет команду **Modbus** для чтения значений из регистров флагов (**Coil**).

При подключении к блоку **Master** блок **ModbusCoilIn** выполняет команду **0x01**, при подключении к **Slave** – **0x05**.

Таблица 3.6 – Назначение входов и выходов ModbusCoilIn

Элемент	Описание
Входы	
itr	Связь с блоком ***Modbus Master***/***Modbus Slave***
enb	Разрешение работы (при подключении к ***Modbus Master***)
slv	Адрес ведомого устройства, с которого считывают данные
adr0	Адрес, с которого начинается чтение
Выходы	
bi	Указатель на блок, не используется
st	Статус работы (при подключении к ***Modbus Master***): **0** – нет связи; **1** – есть связь, ошибок нет; **>1** – есть связь, код ошибки в старшем байте
o	Результаты чтения (циклический)

Для инициализации следует подключить вход **itr** к выходу блока Master или Slave и задать верный адрес устройства.

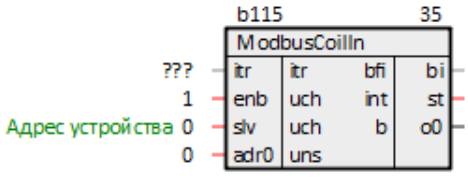


Рисунок 3.8 – Блок ModbusCoilIn

3.3.2 Команда чтения дискретных входов 0x02 (ModbusDInputIn)

Блок ***ModbusDInputIn*** отправляет команды **Modbus** на чтение дискретных входов (**Inputs**).

При подключении к блоку **Master** блок **ModbusDInputIn** выполняет команду **0x02**.

Таблица 3.7 – Назначение входов и выходов ModbusDInputIn

Элемент	Описание
Входы	
itr	Связь с блоком ***Modbus Master***
enb	Разрешение работы
slv	Адрес ведомого устройства, с которого считывают данные
adr0	Адрес, с которого начинается чтение
Выходы	
bi	Указатель на блок, не используется
st	Статус работы: **0** – нет связи; **1** – есть связь, ошибок нет; **>1** – есть связь, код ошибки в старшем байте
o	Результаты чтения (циклический)

Для инициализации следует подключить вход **itr** к выходу блока Master и задать верный адрес устройства.

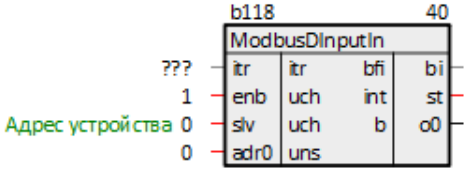


Рисунок 3.9 – Блок ModbusDInputIn

3.3.3 Команда чтения регистров хранения 0x03 (ModbusRegIn)

Блок ***ModbusRegIn*** отправляет команды **Modbus** на чтение регистров хранения (**Holding Registers**).

При подключении к блоку **Master** блок **ModbusRegIn** выполняет команду **0x03**, при подключении к **Slave** – **0x06**.

Таблица 3.8 – Назначение входов и выходов ModbusRegIn

Элемент	Описание
Входы (константные)	
itr	Связь с блоком ***Modbus Master***/***Modbus Slave***
enb	Разрешение работы (при подключении к ***Modbus Master***)
slv	Адрес ведомого устройства, с которого считывают данные
adr0	Адрес, с которого начинается чтение
Выходы	
bi	Указатель на блок, не используется
st	Статус работы (при подключении к ***Modbus Master***): **0** – нет связи; **1** – есть связь, ошибок нет; **>1** – есть связь, код ошибки в старшем байте
o	Результаты чтения (циклический)

Для инициализации следует подключить вход **itr** к выходу блока Master или Slave и задать верный адрес устройства.

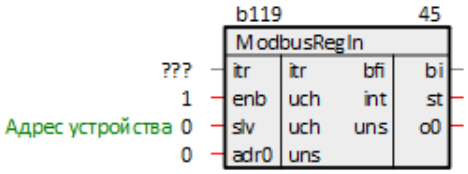


Рисунок 3.10 – Блок ModbusRegIn

3.3.4 Команда чтения входных регистров 0x04 (ModbusAInputIn)

Блок ***ModbusAInputIn*** отправляет команды **Modbus** на чтение регистров ввода (**Input Registers**).

При подключении к блоку **Master** блок **ModbusAInputIn** выполняет команду **0x04**.

Таблица 3.9 – Назначение входов и выходов ModbusAInputIn

Элемент	Описание
Входы	
itr	Связь с блоком ***Modbus Master***
enb	Разрешение работы
slv	Адрес ведомого устройства, с которого считывают данные
adr0	Адрес, с которого начинается чтение
Выходы	
bi	Указатель на блок, не используется
st	Статус работы: **0** – нет связи; **1** – есть связь, ошибок нет; **>1** – есть связь, код ошибки в старшем байте
o	Результаты чтения (циклический)

Для инициализации следует подключить вход **itr** к выходу блока Master и задать верный адрес устройства.

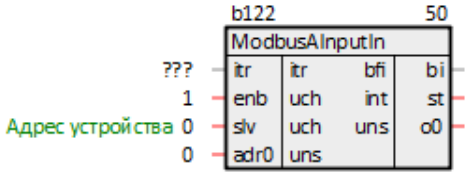


Рисунок 3.11 – Блок ModbusAInputIn

3.3.5 Команда чтения вещественного числа 0x03/0x04 (ModbusFltIn)

Блок ***ModbusFltIn*** отправляет команды **Modbus** на чтение вещественных чисел из регистров хранения (**Holding Registers**) или регистров ввода (**Input Registers**). Каждое число занимает два регистра.

При подключении к блоку **Master** блок **ModbusFltIn** выполняет команды **0x04** или **0x03**, при подключении к блоку **Slave** – **0x10** (для регистров хранения).

Таблица 3.10 – Назначение входов и выходов ModbusFltIn

Элемент	Описание
Входы	
itr	Связь с блоком ***Modbus Master***/***Modbus Slave***
enb	Разрешение работы (при подключении к ***Modbus Master***)
slv	Адрес ведомого устройства, с которого считывают данные
adr0	Адрес, с которого начинается чтение
hold/in	Выбор регистров для чтения – регистры хранения/регистры ввода (**0**/**1**)
ord_w	Порядок слов в числе – прямой/обратный (**0**/**1**)
ord_b	Порядок байт в числе – прямой/обратный (**0**/**1**)
Выходы	
bi	Указатель на блок, не используется
st	Статус работы (при подключении к ***Modbus Master***): **0** – нет связи; **1** – есть связь, ошибок нет; **>1** – есть связь, код ошибки в старшем байте
o	Результаты чтения (циклический)

Таблица 3.11 – Порядок слов и байт

ord_w	ord_b	Порядок байт
0	0	AB CD
0	1	BA DC
1	0	CD AB
1	1	DC BA

Для инициализации следует подключить вход **itr** к выходу блока Master или Slave и задать верный адрес устройства.

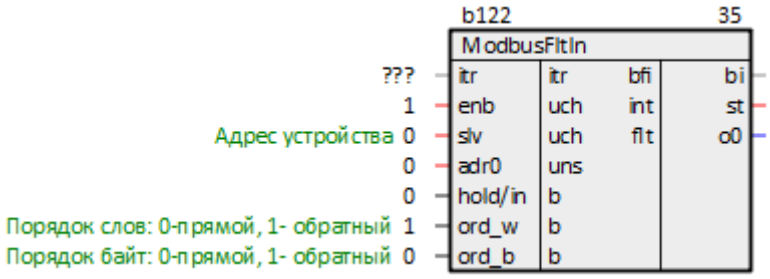


Рисунок 3.12 – Блок ModbusFltIn

3.3.6 Команда чтения результатов измерения аналогового входа (OwenFltIn)

Блок ***OwenFltIn*** отправляет команды **Modbus** на чтение вещественных чисел из регистров хранения (**Holding Registers**) или регистров ввода (**Input Registers**) модулей аналогового ввода ОВЕН МВ210-101.

При подключении к блоку **Master** блок **OwenFltIn** выполняет команды **0x04** или **0x03**.

Таблица 3.12 – Назначение входов и выходов OwenFltIn

Элемент	Описание
Входы	
itr	Связь с блоком ***Modbus Master***
enb	Разрешение работы
slv	Адрес ведомого устройства, с которого считывают данные
adr0	Адрес, с которого начинается чтение
hold/in	Выбор регистров для чтения – регистры хранения/регистры ввода (**0**/**1**)
Выходы	
bi	Указатель на блок, не используется
st	Статус работы: **0** – нет связи; **1** – есть связь, ошибок нет; **>1** – есть связь, ошибка
Циклические выходы	
rslt	Отображает результаты измерения
time	Время измерения
stsi	Код ошибки (см. таблицу 3.13)
vldi	Бит достоверности: **1** – данные достоверны; **0** – данные не достоверны
msk	Маска кода ошибки (см. таблицу 3.13)

Таблица 3.13 – Коды ошибок

Код ошибки	Маска	Описание
0xF0	1	Значение заведомо неверно
0xF6	2	Данные не готовы. Необходимо дождаться результатов первого измерения после включения модуля
0xF7	4	Датчик отключен
0xF8	8	Велика температура свободных концов ТП
0xF9	16	Мала температура свободных концов ТП
0xFA	32	Измеренное значение слишком велико
0xFB	64	Измеренное значение слишком мало
0xFC	128	Короткое замыкание датчика
0xFD	256	Обрыв датчика
0xFE	512	Отсутствие связи с АЦП
0xFF	1024	Некорректный калибровочный коэффициент

Для инициализации следует подключить вход **itr** к выходу блока Master и задать верный адрес устройства.

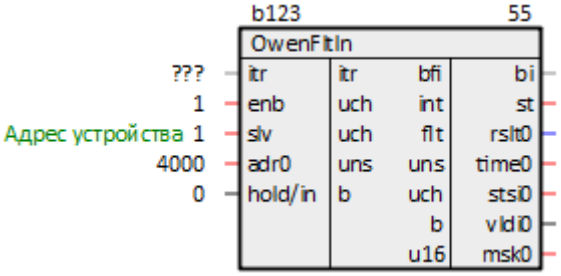


Рисунок 3.13 – Блок OwenFltIn

3.4 Команды записи

В данном разделе приведено описание блоков библиотеки ***paModbus***, которые реализуют команды протокола **Modbus** на запись параметров.

Таблица 3.14 – paModbus. Команды записи

Код функции	Имя функции	Команда	Блоки paModbus	
			Подключение к *Modbus Master*	**Подключение к *Modbus Slave***
5 (0x05)	Write Single Coil	Запись значения в один регистр флага	ModbusCoilOut	ModbusCoilIn
6 (0x06)	Write Single Register	Запись значения в один регистр хранения	ModbusRegOut	ModbusRegIn
15 (0x0F)	Write Multiple Coils	Запись значений в несколько регистров флагов	ModbusCoilsOut	-
16 (0x10)	Write Multiple Registers	Запись значений в несколько регистров хранения	ModbusRegsOut ModbusFltOut	ModbusFltIn

3.4.1 Команда записи одного флага 0x05 (ModbusCoilOut)

Блок **ModbusCoilOut** отправляет команды **Modbus** на запись в регистр флага (**Coil**) дискретного вывода.

При подключении к блоку **Master**блок **ModbusCoilOut** выполняет команду **0x05**, при подключении к **Slave** – **0x01**.

Для записи **1** в флаг на вход следует подать любое положительное число. Для записи **0** – подать **0**.

Таблица 3.15 – Назначение входов и выходов ModbusCoilOut

Элемент	Описание
Входы	
enb	Разрешение работы (при подключении к ***Modbus Master***)
slv	Адрес ведомого устройства, на который записывают данные
adr0	Адрес, с которого начинается запись
in	Входы записи данных (циклический)
Выходы	
bo	Связь с блоком ***Modbus Master***/***Modbus Slave***
st	Статус работы (при подключении к ***Modbus Master***): **0** – нет связи; **1** – есть связь, ошибок нет; **>1** – есть связь, код ошибки в старшем байте

Для инициализации следует подключить выход **bo** к входу блока Master или Slave и задать верный адрес устройства.

ВНИМАНИЕ
Блоки записи обязательно следует размещать в программе **перед** блоком Master/Slave (см. свойства блоков ***Порядок***).

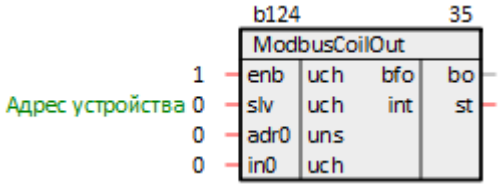


Рисунок 3.14 – Блок ModbusCoilOut

3.4.2 Команда записи одного регистра хранения 0x06 (ModbusRegOut)

Блок ***ModbusRegOut*** отправляет команды **Modbus** на запись регистра хранения (**Holding Register**).

При подключении к блоку **Master** блок **ModbusRegOut** выполняет команду **0x06**, при подключении к **Slave** – **0x03**.

Таблица 3.16 – Назначение входов и выходов ModbusRegOut

Элемент	Описание
Входы	
enb	Разрешение работы (при подключении к ***Modbus Master***)
slv	Адрес ведомого устройства, на который записывают данные
adr0	Адрес, с которого начинается запись
in	Входы записи данных (циклический)
Выходы	
bo	Связь с блоком ***Modbus Master***/***Modbus Slave***
st	Статус работы (при подключении к ***Modbus Master***): **0** – нет связи; **1** – есть связь, ошибок нет; **>1** – есть связь, код ошибки в старшем байте

Для инициализации следует подключить выход **bo** к входу блока Master или Slave и задать верный адрес устройства.

ВНИМАНИЕ
Блоки записи обязательно следует размещать в программе **перед** блоком Master/Slave (см. свойства блоков ***Порядок***).

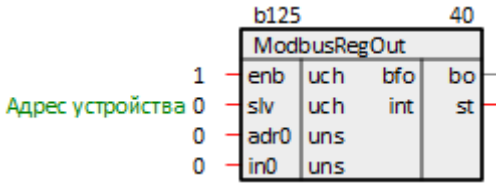


Рисунок 3.15 – Блок ModbusRegOut

3.4.3 Команда записи нескольких флагов 0x0F (ModbusCoilsOut)

Блок ***ModbusCoilsOut*** отправляет команды **Modbus** на запись нескольких флагов (**Coils**).

При подключении к блоку **Master** блок **ModbusCoilsOut** выполняет команду **0x0F**, при подключении к **Slave** – **0x01**.

Таблица 3.17 – Назначение входов и выходов ModbusCoilsOut

Элемент	Описание
Входы	
enb	Разрешение работы (при подключении к ***Modbus Master***)
slv	Адрес ведомого устройства, на который записывают данные
adr0	Адрес, с которого начинается запись
in	Входы записи данных (циклический)
Выходы	
bo	Связь с блоком ***Modbus Master***/***Modbus Slave***
st	Статус работы (при подключении к ***Modbus Master***): **0** – нет связи; **1** – есть связь, ошибок нет; **>1** – есть связь, код ошибки в старшем байте

Для инициализации следует подключить выход **bo** к входу блока Master или Slave и задать верный адрес устройства.

ВНИМАНИЕ
Блоки записи обязательно следует размещать в программе **перед** блоком Master/Slave (см. свойства блоков ***Порядок***).

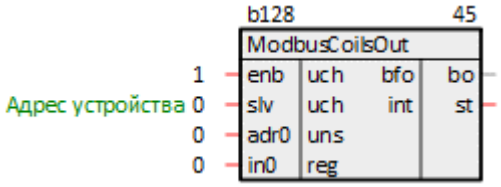


Рисунок 3.16 – Блок ModbusCoilsOut

3.4.4 Команда записи нескольких регистров хранения 0x10 (ModbusRegsOut)

Блок ***ModbusRegsOut*** отправляет команды **Modbus** на запись нескольких регистров хранения (**Holding Registers**).

При подключении к блоку **Master** блок **ModbusRegsOut** выполняет команду **0x10**, при подключении к **Slave** – **0x03**.

Таблица 3.18 – Назначение входов и выходов ModbusRegsOut

Элемент	Описание
Входы	
enb	Разрешение работы (при подключении к ***Modbus Master***)
slv	Адрес ведомого устройства, на который записывают данные
adr0	Адрес, с которого начинается запись
in	Входы записи данных (циклический)
Выходы	

Продолжение таблицы 3.18

Элемент	Описание
bo	Связь с блоком ***Modbus Master***/***Modbus Slave***
st	Статус работы (при подключении к ***Modbus Master***): **0** – нет связи; **1** – есть связь, ошибок нет; **>1** – есть связь, код ошибки в старшем байте

Для инициализации следует подключить выход **bo** к входу блока **Master** или **Slave** и задать верный адрес устройства.

ВНИМАНИЕ
Блоки записи обязательно следует размещать в программе **перед** блоком Master/Slave (см. свойства блоков ***Порядок***).

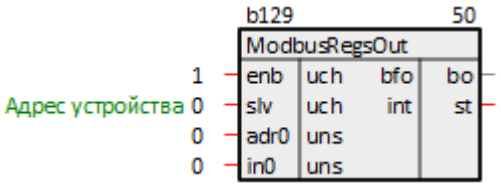


Рисунок 3.17 – Блок ModbusRegsOut

3.4.5 Команда записи вещественного числа 0x10 (ModbusFltOut)

Блок ***ModbusFltOut*** отправляет команды **Modbus** на запись вещественных чисел в регистры хранения (**Holding Registers**). Каждое число занимает два регистра.

При подключении к блоку **Master** блок **ModbusFltOut** выполняет команду **0x10**, при подключении к блоку **Slave** – **0x03**.

Таблица 3.19 – Назначение входов и выходов ModbusFltOut

Элемент	Описание
Входы	
enb	Разрешение работы (при подключении к ***Modbus Master***)
slv	Адрес ведомого устройства, на который записывают данные
adr0	Адрес, с которого начинается запись
ord_w	Порядок слов в числе – прямой/обратный (**0**/**1**)
ord_b	Порядок байт в числе – прямой/обратный (**0**/**1**)
in	Входы записи данных (циклический)
Выходы	
bo	Связь с блоком ***Modbus Master***/***Modbus Slave***
st	Статус работы (при подключении к ***Modbus Master***): **0** – нет связи; **1** – есть связь, ошибок нет; **>1** – есть связь, код ошибки в старшем байте

Таблица 3.20 – Порядок слов и байт

ord_w	ord_b	Порядок байт
0	0	AB CD
0	1	BA DC
1	0	CD AB
1	1	DC BA

Для инициализации следует подключить выход **bo** к входу блока Master или Slave и задать верный адрес устройства.

ВНИМАНИЕ
Блоки записи обязательно следует размещать в программе **перед** блоком Master/Slave (см. свойства блоков ***Порядок***).

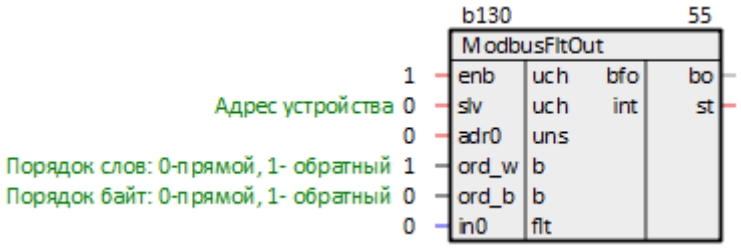


Рисунок 3.18 – Блок ModbusFltOut

3.5 Буфер чтения/записи уставок с плавающей точкой (BufSupFltEx)

Блок **BufSupFltEx** представляет собой двунаправленный буфер данных интерфейса. Блок сохраняет данные в бинарном файле на диске контроллера (так же, как блок **SaverEx** из библиотеки **paCore**).

Отличие блока ***BufSupFltEx*** от блока ***BufSupEx*** из библиотеки ***paCore*** заключается в том, что он позволяет записывать по протоколу Modbus уставки в формате с плавающей точкой. Пример работы с блоком приведен в разделе 5.2.

Поскольку операции файлового ввода/вывода занимают значительное время, данный блок следует размещать только в ***Фоне***.

Таблица 3.21 – Назначение входов и выходов BufSupFltEx

Элемент	Описание
Входы	
inter	Связь от интерфейса, к которому принадлежит данный буфер
group	Номер группы (константный)
fnm	Абсолютный путь и имя файла (может быть пустым – задается автоматически), расширение игнорируется. При сохранении данных на внешнем накопителе следует использовать путь, указанный на выходе блока ***210-SD-USB*** из библиотеки ***paOwenIO*** (константный)
mask	Не используется
rst	Сброс ошибок записи
wr	Запись на диск
Циклические входы	
dan	Значение, которое записывается в буфер при **czap** = 1
czap	Запись значения **dan**
typ	Тип параметра: **AI**, **AO** – вещественное значение (константный)
adr	Адрес параметра (константный)
ini	Значение для инициализации (константный)
min	Минимум, если принятое значение меньше **min**, то оно игнорируется
max	Максимум, если принятое значение больше **max**, то оно игнорируется
Выходы	
pkt	Подключение к блокам ***OpcUAClient***, ***UABufSups*** из библиотеки ***paOpcUA***
next	Имя следующего файла
enb	Запись разрешена
sts	Статус: **0** – после сброса; **1** – записан; **2** – прочитан; **<0** – ошибка
good	Количество удачных записей
bad	Количество ошибок записи
rej	Количество отклоненных записей
Циклические выходы	

Продолжение таблицы 3.21

Элемент	Описание
dan	Значение параметра, полученное по интерфейсу или на вход **dan** (после проверки на **min** и **max**)
chn	Признак изменения, выставляется в **1** на один цикл выполнения программы, если значение **dan** изменилось
zap	Признак записи, выставляется в **1** на один цикл выполнения программы, если значение с входа **dan** было записано

Номер группы **group** используется в качестве Slave ID при подключении к блоку интерфейса Modbus Slave.

Имя файла и путь к нему задается на входе **fnm**. Имя может быть пустым, тогда имя файла будет выбрано автоматически по индексу блока и файл сохраняется в рабочую директорию контроллера.

Адрес переменной **adr** зависит от интерфейса, к которому подключен буфер, например, адрес регистра Modbus.

Если файл существует на диске, выходы инициализируются сохраненными значениями. Если файла не существует – выходы инициализируются значениями инициализации **ini**.

Запись в файл производится при изменении значений на входах **dan** или по интерфейсу. Если файла на диске не существует и выходы **dan** приняли значения **ini**, то можно записать их на диск принудительно с помощью команды **wr**.

Для сохранности данных одновременно на диске находятся два файла, соответствующие одному архиву. Если контроллер будет перезагружен в момент записи на диск, данные не пропадут, а будут доступны предыдущие значения переменных, записанные в другом файле. При чтении содержимое файла контролируется с помощью контрольной суммы, и только при ее корректности выдается на выходы (поэтому, например, при добавлении новой переменной, значения, записанные в файл, сбросятся на значения **ini**).

Если при записи файла на диск происходит однократная ошибка, блок пытается переименовать текущий файл и снова произвести запись. Если повторная запись оказывается удачной, то продолжается работа в обычном режиме, а выход **bad** инкрементируется. Необходимо принять меры по диагностике или замене носителя, поскольку сбои при записи могут быть следствием скорого выхода его из строя.

Файл, на котором произошел сбой, остается на диске под тем же именем с добавленным к нему суффиксом, равным метке времени сбоя (в мс от 1 января 1970 г). Удалять его не следует, чтобы повторно не использовать потенциально сбойный сектор. Если происходит повторный сбой записи, то блок блокируется (выход **enb** = 0) и больше не производит попыток переименований файлов и записи до тех пор, пока ошибки не будут сброшены фронтом на входе **rst**.

ВНИМАНИЕ
При изменении числа входов блока ***BufSupFltEx*** файлы на диске перезаписываются.

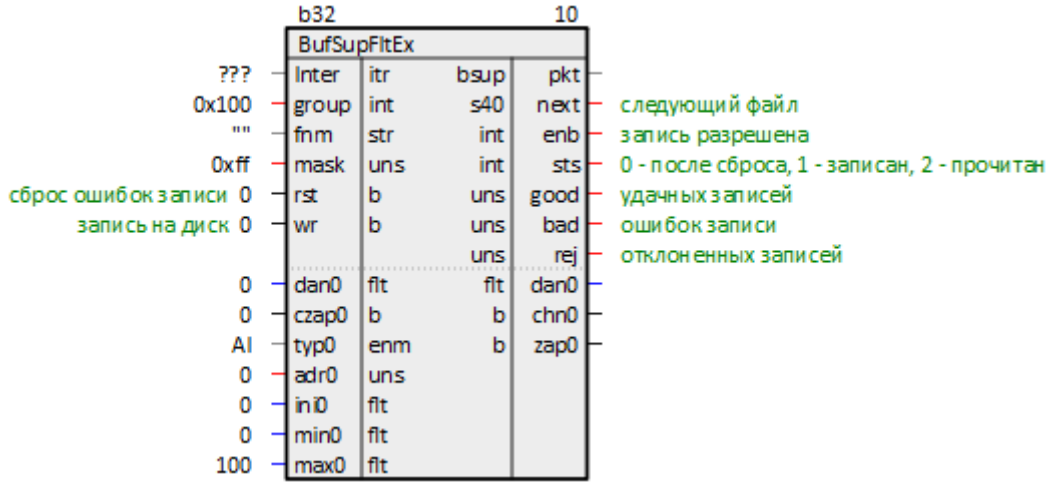


Рисунок 3.19 – Буфер чтения/записи уставок с плавающей точкой (BufSupFltEx)

3.6 Диагностика и управление обменом

Практически у каждого блока, который используют для обмена по протоколу Modbus, есть выход статуса работы для диагностики обмена.

Таблица 3.22 – Статусы блоков обмена по протоколу Modbus RTU

Блок	Имя переменной	Значение	Описание
Функциональные блоки настройки COM-портов			
Порт RS-485	stat	<0	Ошибка
		1	Корректная работа
Порт RS-232	stat	<0	Ошибка
		1	Корректная работа
Modbus RTU			
Modbus RTU Master	sts	0	Корректная работа
		1	Slave не отвечает
		2	Неверная контрольная сумма
Modbus RTU Slave	sts	-	Не используется

Таблица 3.23 – Статусы блоков обмена по протоколу Modbus TCP

Блок	Имя переменной	Значение	Описание
Функциональные блоки настройки TCP-соединений			
TcpIpSrA	stat	0	Есть подключения
		>0	Нет подключений
TcpIpClA	stat	0	Есть связь с TCP/IP-сервером
		>0	Нет связи с TCP/IP-сервером
Modbus TCP			
Modbus TCP Master	sts	0	Есть связь с TCP/IP-сервером
		>0	Нет связи или Slave не отвечает
Modbus TCP Slave	sts	0	Есть связь с Master
		>0	Нет связи или нет запросов от Master

Таблица 3.24 – Статусы блоков чтения/записи при подключении к Modbus RTU/TCP Master

Блок	Имя переменной	Значение	Описание
Команды чтения			
ModbusCoilIn	st	0	Нет связи
		1	Есть связь, нет ошибок
		>1	Есть связь, код ошибки в старшем байте
ModbusDInputIn	st	0	Нет связи
		1	Есть связь, нет ошибок
		>1	Есть связь, код ошибки в старшем байте
ModbusRegIn	st	0	Нет связи
		1	Есть связь, нет ошибок
		>1	Есть связь, код ошибки в старшем байте
ModbusAInputIn	st	0	Нет связи
		1	Есть связь, нет ошибок

Продолжение таблицы 3.24

Блок	Имя переменной	Значение	Описание
		>1	Есть связь, код ошибки в старшем байте
ModbusFltIn	st	0	Нет связи
		1	Есть связь, нет ошибок
		>1	Есть связь, код ошибки в старшем байте
OwenFltIn	st	0	Нет связи
		1	Есть связь, нет ошибок
		>1	Есть связь, код ошибки в старшем байте
Команды записи			
ModbusCoilOut	st	0	Нет связи
		1	Есть связь, нет ошибок
		>1	Есть связь, код ошибки в старшем байте
ModbusRegOut	st	0	Нет связи
		1	Есть связь, нет ошибок
		>1	Есть связь, код ошибки в старшем байте
ModbusCoilsOut	st	0	Нет связи
		1	Есть связь, нет ошибок
		>1	Есть связь, код ошибки в старшем байте
ModbusRegsOut	st	0	Нет связи
		1	Есть связь, нет ошибок
		>1	Есть связь, код ошибки в старшем байте
ModbusFltOut	st	0	Нет связи
		1	Есть связь, нет ошибок
		>1	Есть связь, код ошибки в старшем байте

Блоки ***Modbus RTU Slave***, ***Modbus TCP Slave***, ***Modbus RTU Master*** и ***Modbus TCP Master*** имеют вход **enbl** для разрешения работы блока. Если задать на входе значение:

- **1**, то блок реализует протокол обмена;
- **0** – блок не работает, в выходе статуса **sts** устанавливается 0.

Блоки чтения/записи имеют входы **enb** для разрешения работы блоков. Вход **enb** не используется при подключении к блокам ***Modbus RTU*/*TCP Slave***. Если задать на входе значение:

- **1**, то блок читает/записывает регистры подключенного устройства;
- **0** – блок не работает, в выходе статуса **st** при этом устанавливается **0**.

4 Методика настройки обмена по протоколу Modbus

4.1 Общая методика конфигурирования интерфейсов

Настройка обмена по протоколу **Modbus** в **Полигон** состоит из следующих действий:

Рисунок 4.1 – Последовательность конфигурирования Modbus в Полигон

Сначала необходимо выбрать интерфейс для обмена – COM-порт или Ethernet. Для COM-порта нужно выбрать и настроить блок COM-порта. Для Ethernet нужно добавить и настроить TCP/IP-сервер или TCP/IP-клиент.

Затем необходимо выбрать режим работы интерфейса – Master или Slave, добавить соответствующий блок протокола и соединить с блоком COM-порта/TCP-соединения.

Если интерфейс работает в режиме Master, то следует добавить блоки команд для опроса подчиненных устройств, указать их адреса и адреса опрашиваемых/записываемых регистров.

Если интерфейс работает в режиме Slave, следует добавить блоки поддерживаемых команд и адреса выделяемых регистров.

4.2 Настройка ПЛК в режиме Modbus RTU Master

Для настройки ПЛК в режиме ***Modbus RTU Master*** следует выполнить следующие действия:

1. Создать новую ***Программу*** в проекте в месте работы ***Фон*** (блок Modbus RTU Master рекомендуется размещать в фоне). В свойствах созданной программы задать ***Имя*** – *Modbus* (или любое другое).
2. Создать внутри программы ***Страницу***. Добавить свойство ***Комментарии*** и задать *Modbus RTU Master* (или любой другой).

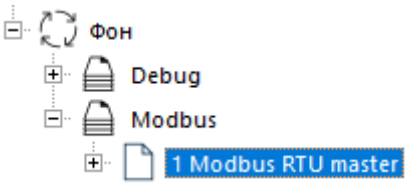


Рисунок 4.2 – Создание страницы для настройки опроса в режиме Modbus RTU Master

3. Создать на странице блок настройки COM-порта из библиотеки ***paOwenIO***. Задать номер используемого COM-порта (для ***210-RS485***), задать сетевые настройки интерфейса.

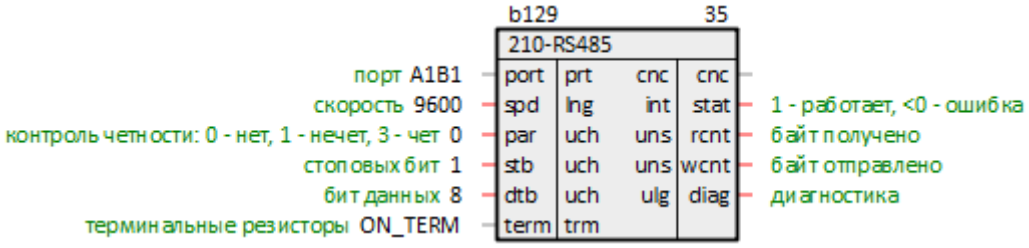


Рисунок 4.3 – Добавление блока настройки COM-порта

4. Создать на странице блок Modbus RTU Master из библиотеки ***paModbus***. Задать настройки Master-устройства.

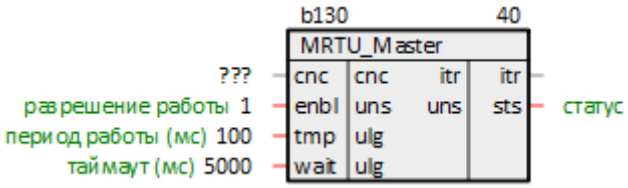


Рисунок 4.4 – Добавление блока Modbus RTU Master

5. Соединить выход блока COM-порта **cnc** с соответствующим входом блока ***Modbus RTU Master***.

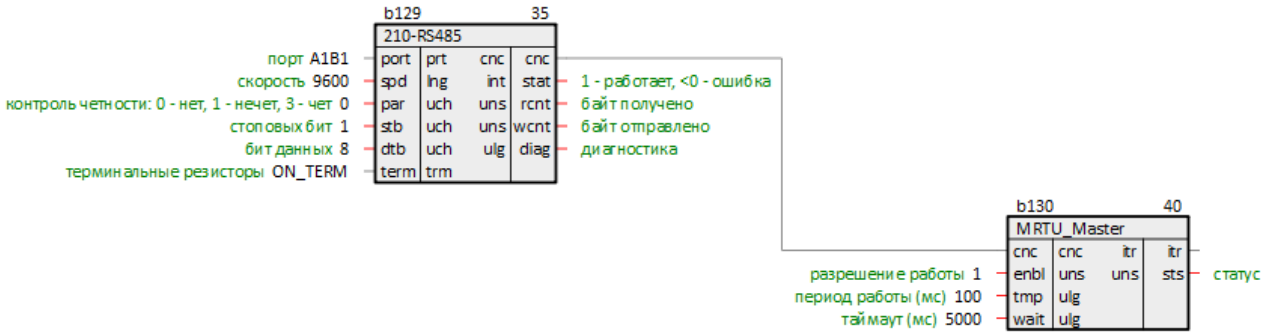


Рисунок 4.5 – Связь блока настройки COM-порта с Modbus RTU Master

6. Создать на странице блоки команд на чтение данных из Slave-устройств. Задать адреса Slave-устройств. Задать адреса опрашиваемых регистров.

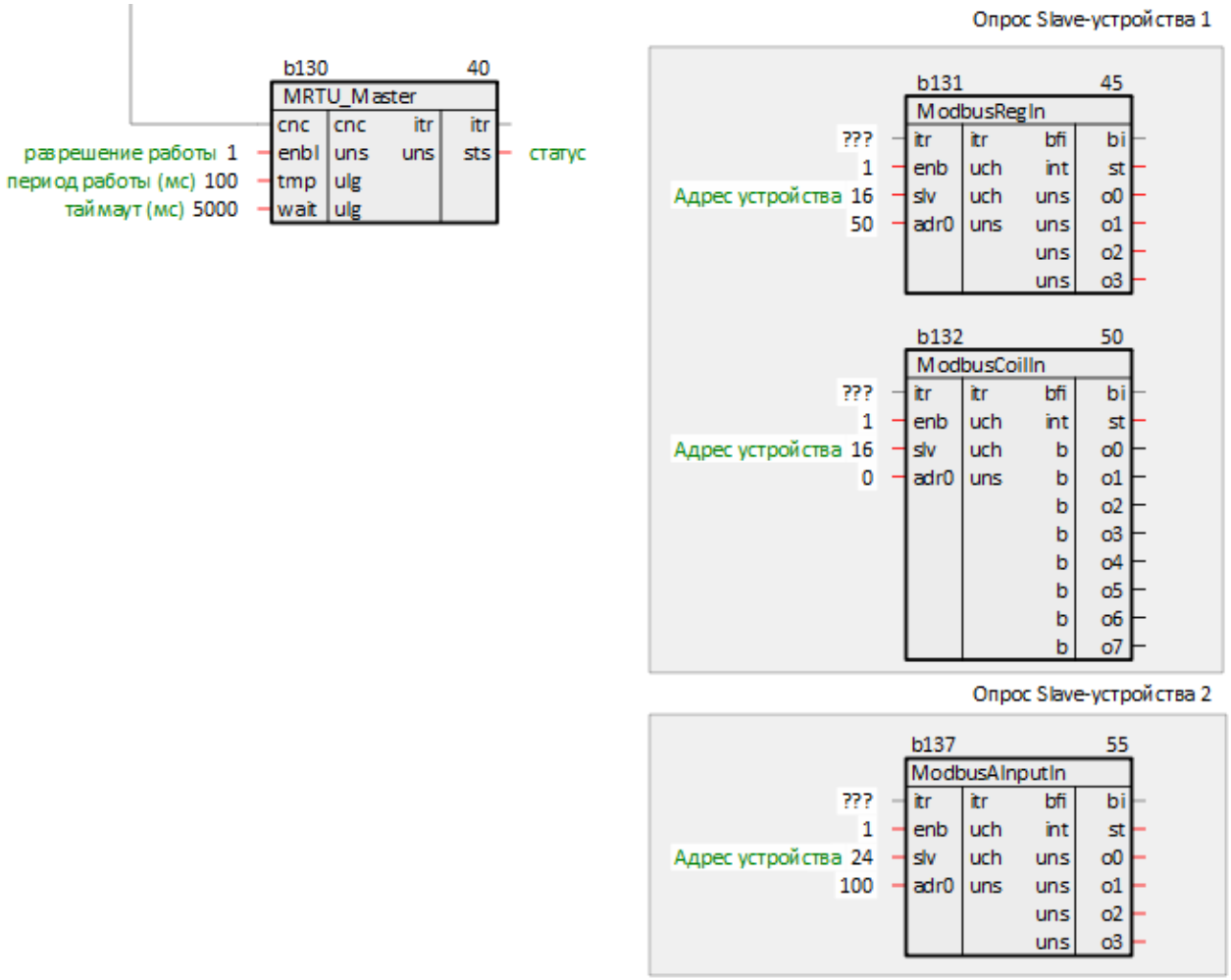


Рисунок 4.6 – Добавление и настройка блоков команд на чтение

7. Соединить выходы блока ***Modbus RTU Master*** **itr** с соответствующими входами блоков команд чтения регистров.

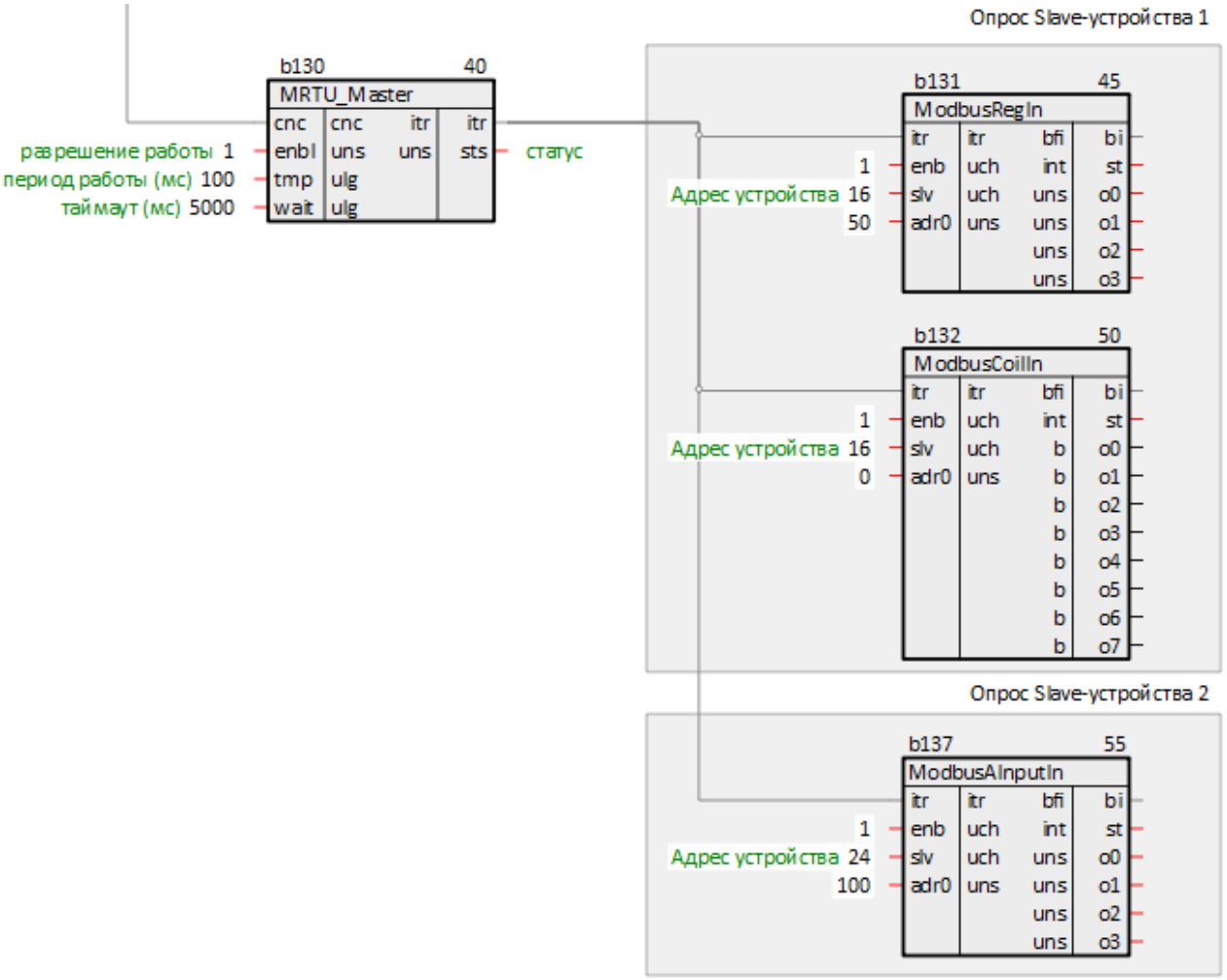


Рисунок 4.7 – Связь блоков команд чтения с Modbus RTU Master

8. Если требуется, можно соединить выходы **o** блоков чтения с входами других блоков в проекте.
9. Создать блоки команд на запись данных в Slave-устройства. Задать адреса Slave-устройств. Задать адреса записываемых регистров.

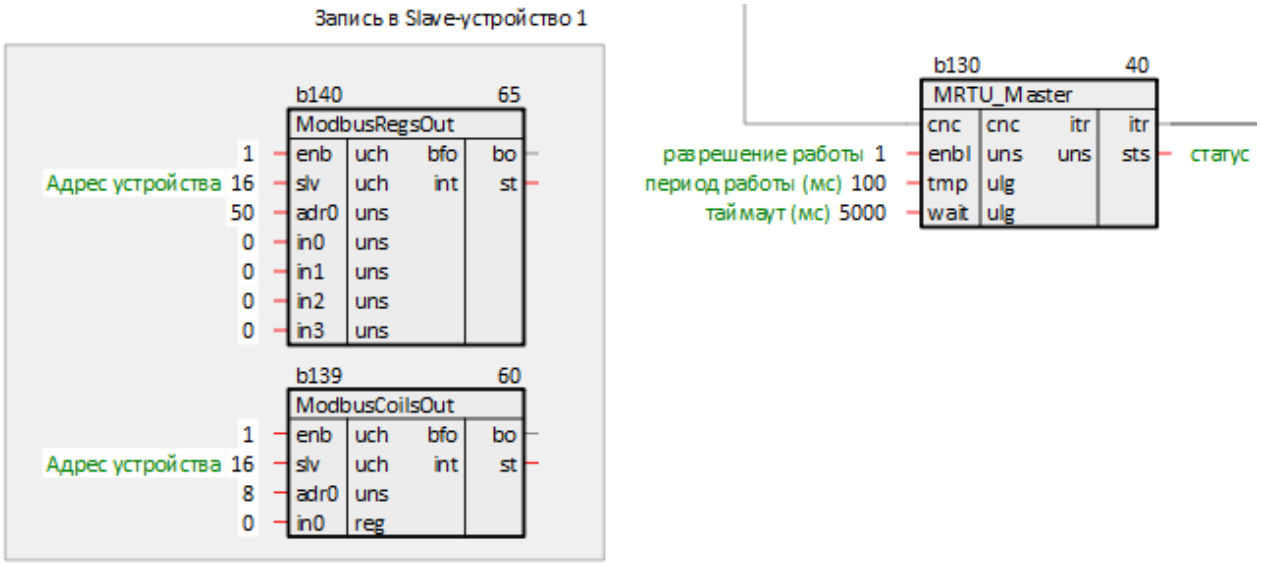


Рисунок 4.8 – Добавление и настройка блоков команд на запись

10. Создать у блока ***Modbus RTU Master*** входы типа **bfo** количеством, соответствующим количеству добавленных команд на запись. Затем соединить выходы блоков записи с добавленными входами **bo** блока ***Modbus RTU Master***.

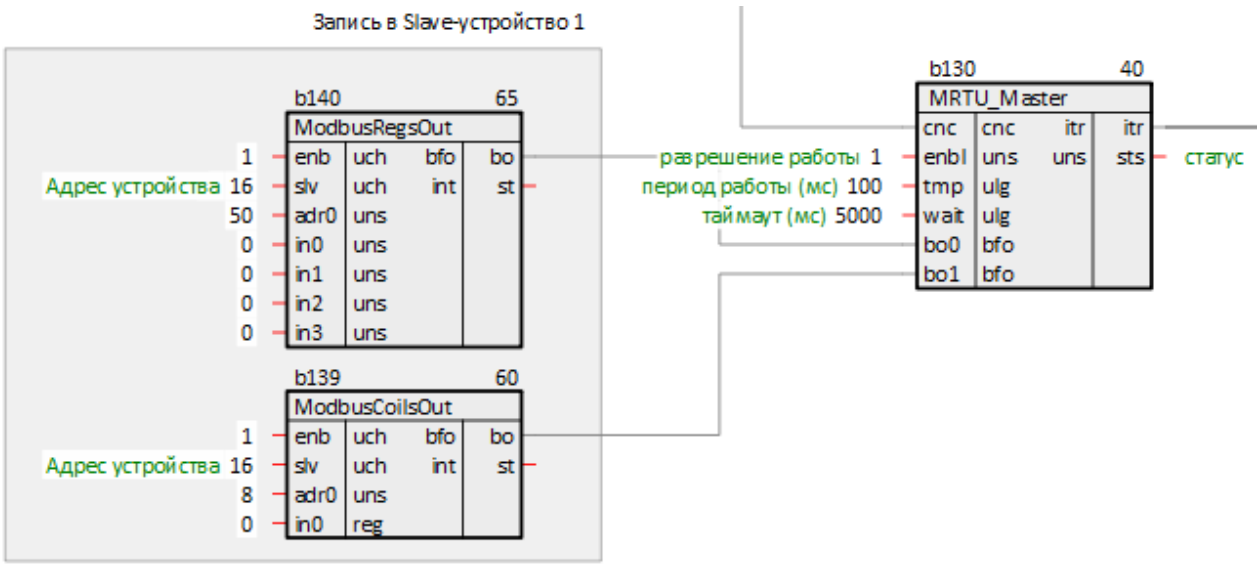


Рисунок 4.9 – Связь блоков команд записи с Modbus RTU Master

11. Если требуется, можно соединить входы **in** блоков записи с выходами других блоков в проекте.
12. Задать порядок выполнения блоков ***По потоку данных***.

Таким образом, получится следующий вид страницы *Modbus RTU Master*:

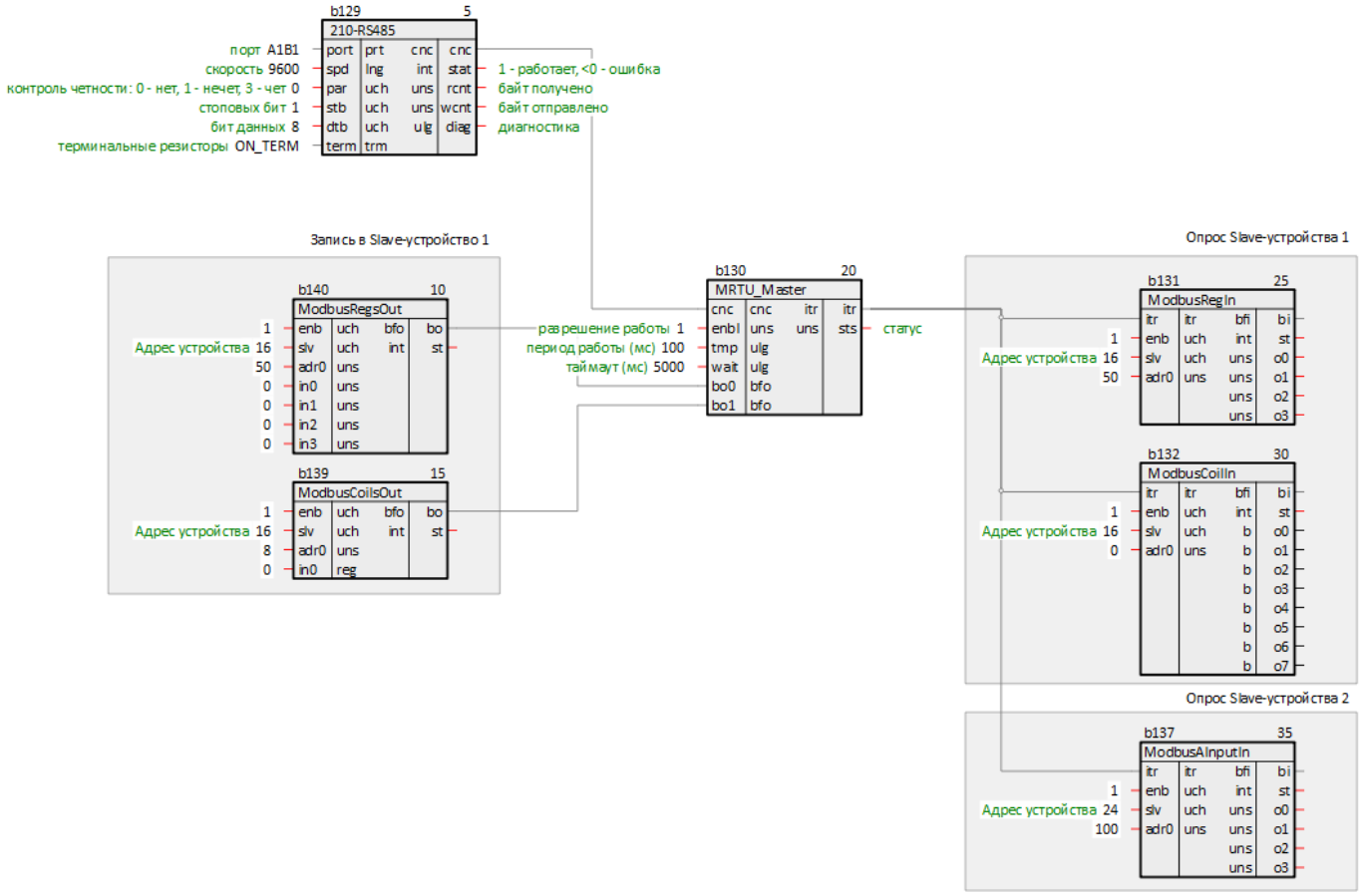


Рисунок 4.10 – Настройка Modbus RTU Master

Пример настройки ПЛК в режиме Modbus RTU Master для опроса модулей Mx110 приведен в разделе 6.1.

4.3 Настройка ПЛК в режиме Modbus RTU Slave

Для настройки ПЛК в режиме ***Modbus RTU Slave*** следует:

1. Создать новую ***Программу*** в проекте в месте работы ***Фон*** (блок Modbus RTU Slave рекомендуется размещать в фоне). В свойствах созданной программы задать ***Имя*** – *Modbus* (или любое другое).
2. Создать внутри программы ***Страницу***. Добавить свойство ***Комментарии*** и задать *Modbus RTU Slave* (или любой другой).

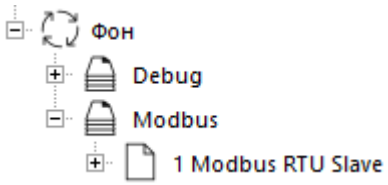


Рисунок 4.11 – Создание страницы для настройки опроса в режиме Modbus RTU Slave

3. Создать на странице блок настройки COM-порта из библиотеки ***paOwenIO***. Задать номер используемого COM-порта (для ***210-RS485***), задать сетевые настройки интерфейса.

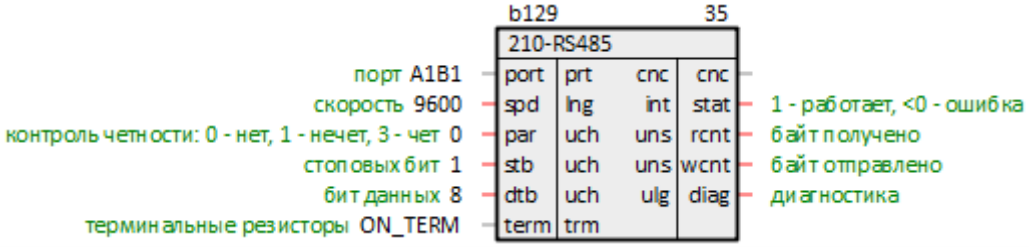


Рисунок 4.12 – Добавление блока настройки COM-порта

4. Создать на странице блок Modbus RTU Slave из библиотеки ***paModbus***. Задать настройки Slave-устройства.

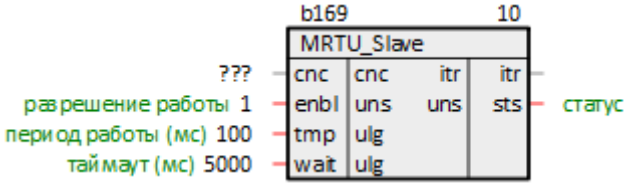


Рисунок 4.13 – Добавление блока Modbus RTU Slave

5. Соединить выход блока COM-порта **cnc** с соответствующим входом блока ***Modbus RTU Slave***.

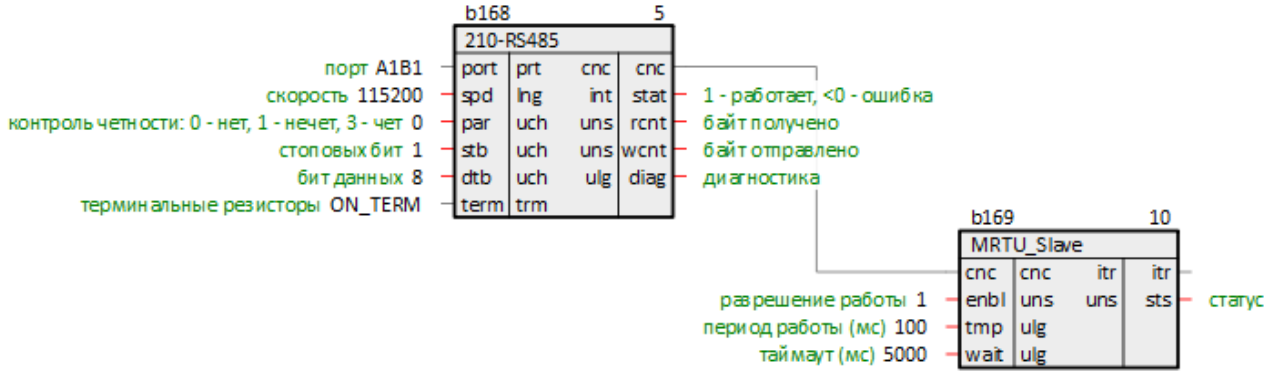


Рисунок 4.14 – Связь блока настройки COM-порта с Modbus RTU Slave

6. Создать на странице блоки команд на чтение/запись данных Slave-устройства. Задать адрес Slave-устройства. Задать адреса выделяемых регистров. Если необходимо и записывать, и считывать один и тот же регистр, то необходимо соединить выход блока записи со сходом блока чтения.
7. Соединить выходы блока ***Modbus RTU Slave*** **itr** с соответствующими входами блоков команд записи регистров.
8. Если требуется, можно соединить выходы **o** блоков записи с входами других блоков в проекте.
9. Создать у блока ***Modbus RTU Slave*** входы типа **bfo** количеством, соответствующим количеству добавленных команд на чтение. Затем соединить выходы блоков чтения с добавленными входами **bo** блока ***Modbus RTU Slave***.

10. При необходимости соединить входы **in** блоков чтения с выходами других блоков в проекте.
11. Задать порядок выполнения блоков ***По потоку данных***.

Таким образом, получится следующий вид страницы *Modbus RTU Slave*:

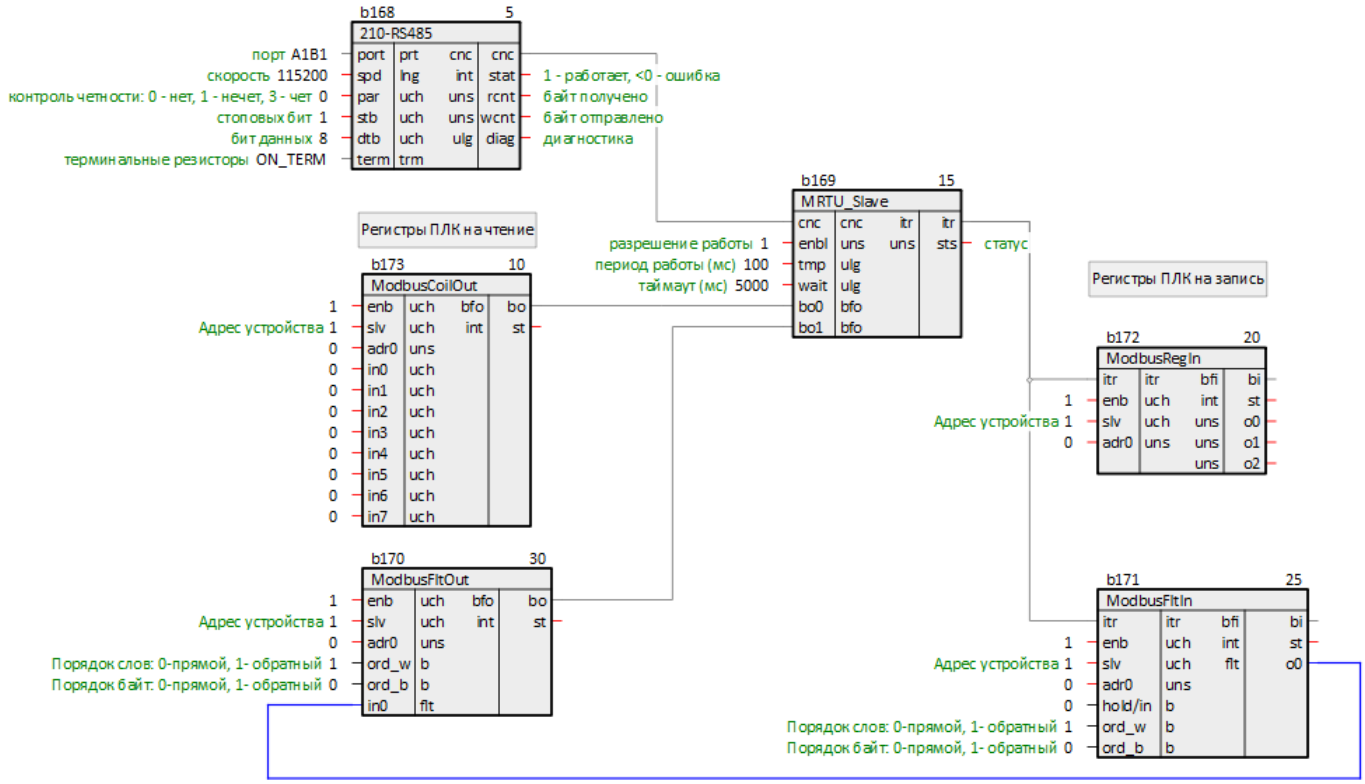


Рисунок 4.15 – Настройка Modbus RTU Slave

Пример настройки ПЛК в режиме Modbus RTU Slave для опроса Owen OPC Server приведен в разделе 6.2.

4.4 Настройка ПЛК в режиме Modbus TCP Master

Для настройки ПЛК в режиме ***Modbus TCP Master*** следует:

1. Создать новую ***Программу*** в проекте в месте работы ***Фон*** (блоки TcpIpCIA и Modbus TCP Master рекомендуется размещать в фоне). В свойствах созданной программы задать ***Имя*** – *Modbus* (или любое другое).
2. Создать внутри программы ***Страницу***. Добавить свойство ***Комментарии*** и задать *Modbus TCP Master* (или любой другой).

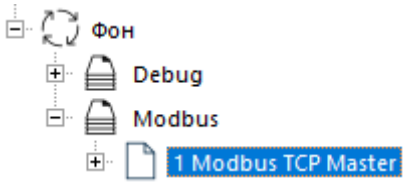


Рисунок 4.16 – Создание страницы для настройки опроса в режиме Modbus TCP Master

3. Создать на странице блок TCP/IP-клиента TcpIpCIA из библиотеки ***paCore***. Задать локальные порт и IP адрес TCP-клиента и удаленные порт и адрес TCP-сервера.

   При настройке блока ***TcpIpCIA*** удобно использовать некоторые свойства модуля. Для этого можно использовать технологию SQL-запросов. Это позволяет изменять IP адреса и порты в одном месте, и использовать эти значения в разных частях проекта.

   Запрос IP адреса (prop_ip):

   ```
   "<sql>SELECT value FROM blocks_prop WHERE indx=:module AND type="prop_ip"</sql>"
   ```

   Запрос пользовательского свойства ***Пользовательское свойство 00*** (prop_0):

   ```
   <sql> SELECT value FROM blocks_prop WHERE indx=:module AND type="prop_0"</sql>
   ```

Для каждого опрашиваемого прибора необходимо добавлять свой блок TCP/IP-клиента.

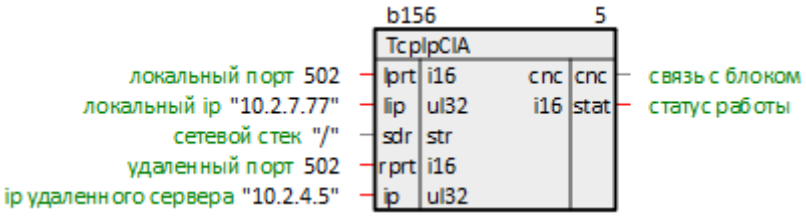


Рисунок 4.17 – Добавление блока TCP/IP-клиента

4. Создать на странице блок Modbus TCP Master из библиотеки ***paModbus***. Задать настройки Master-устройства.

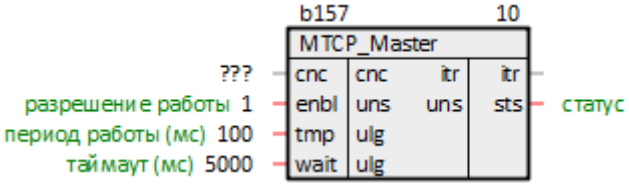


Рисунок 4.18 – Добавление блока Modbus TCP Master

5. Соединить выход блока ***TcpIpCIA*** **cnc** с соответствующим входом блока ***Modbus TCP Master***.

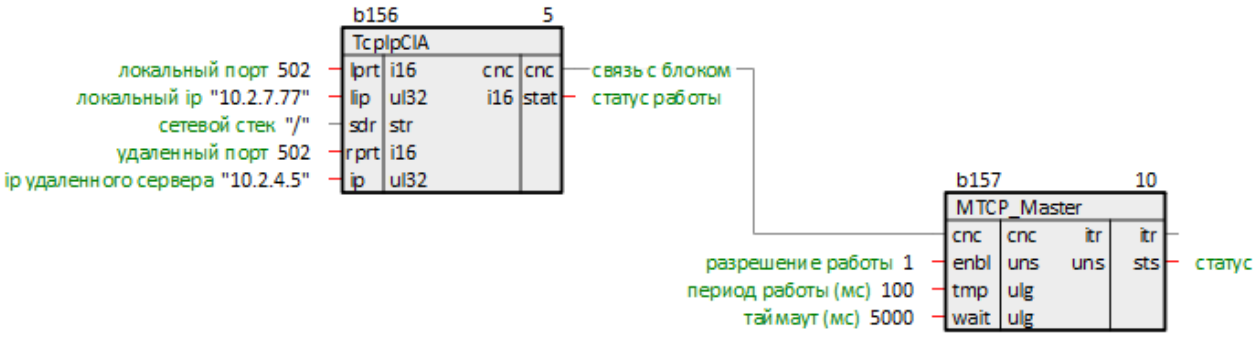


Рисунок 4.19 – Связь блока TCP/IP-клиента с Modbus TCP Master

6. Создать на странице блоки команд на чтение данных из Slave-устройства. Задать ID Slave-устройства. Задать адреса опрашиваемых регистров.

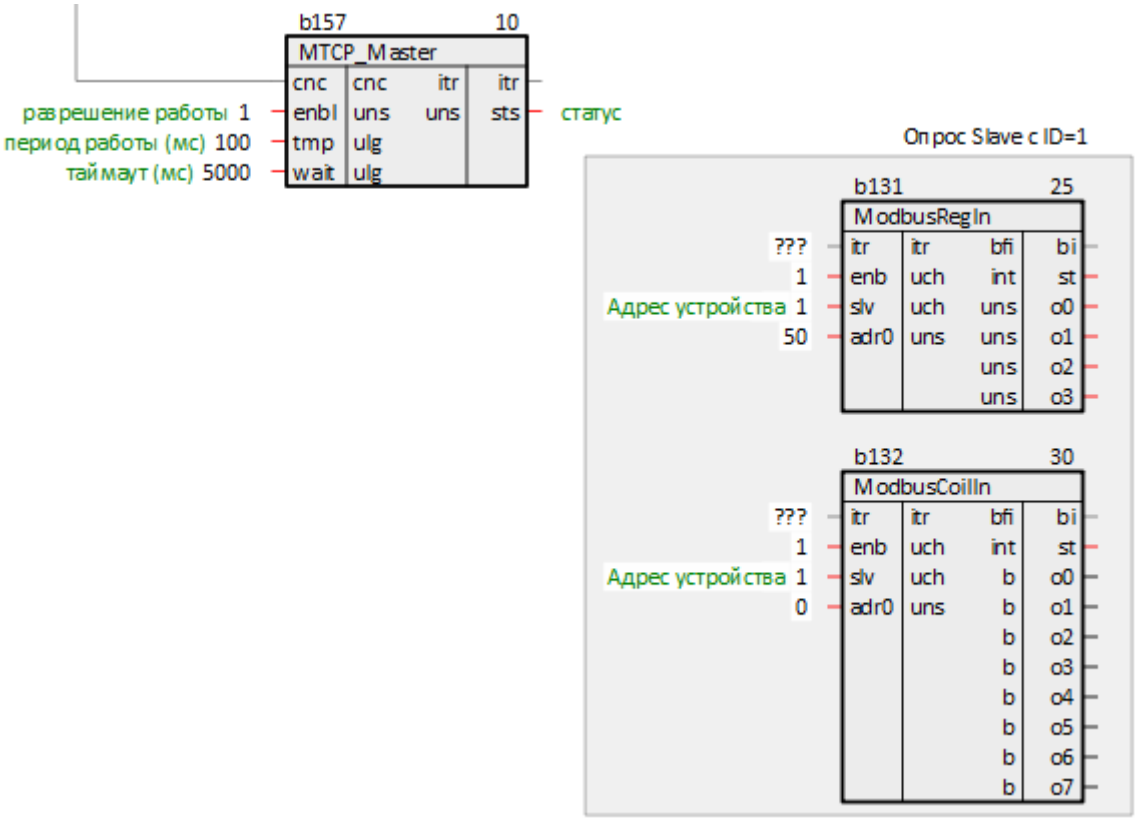


Рисунок 4.20 – Добавление и настройка блоков команд на чтение

7. Соединить выходы блока ***Modbus TCP Master*** **itr** с соответствующими входами блоков команд чтения регистров.

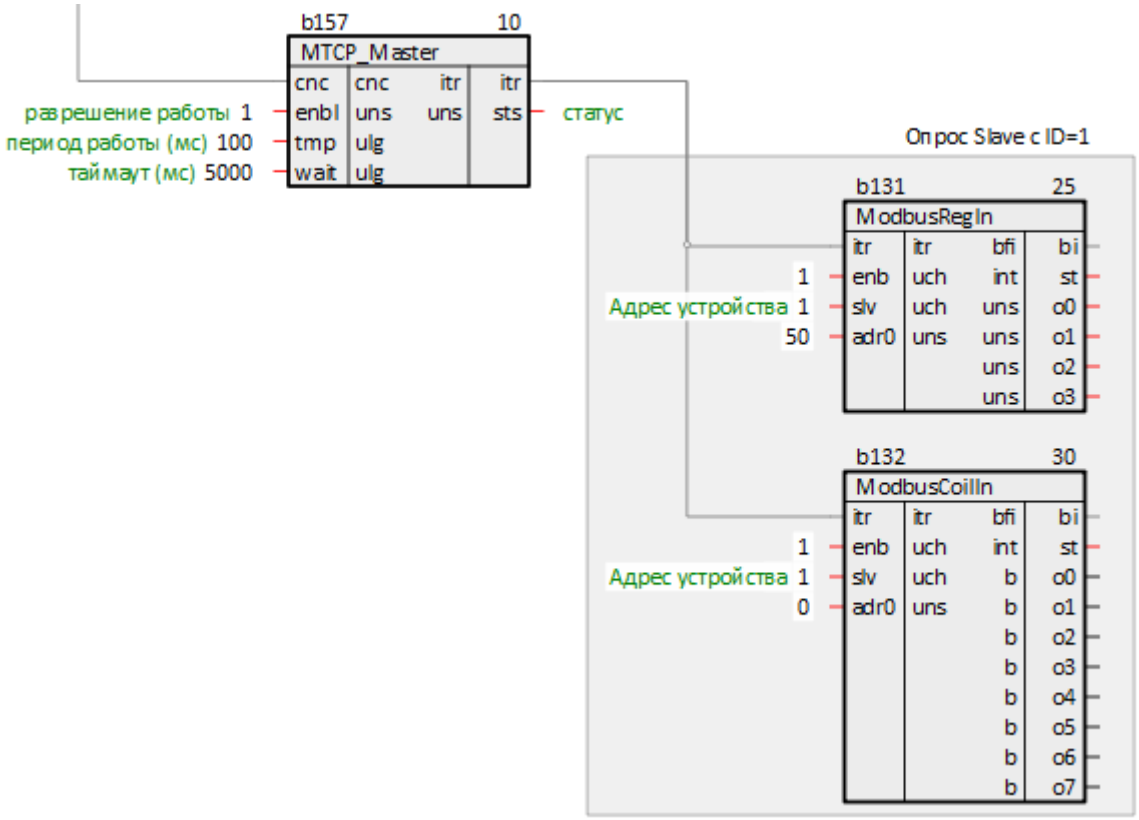


Рисунок 4.21 – Связь блоков команд чтения с Modbus TCP Master

8. Если требуется, можно соединить выходы **o** блоков чтения с входами других блоков в проекте.

9. Создать блоки команд на запись данных в Slave-устройства. Задать ID Slave-устройства. Задать адреса записываемых регистров.

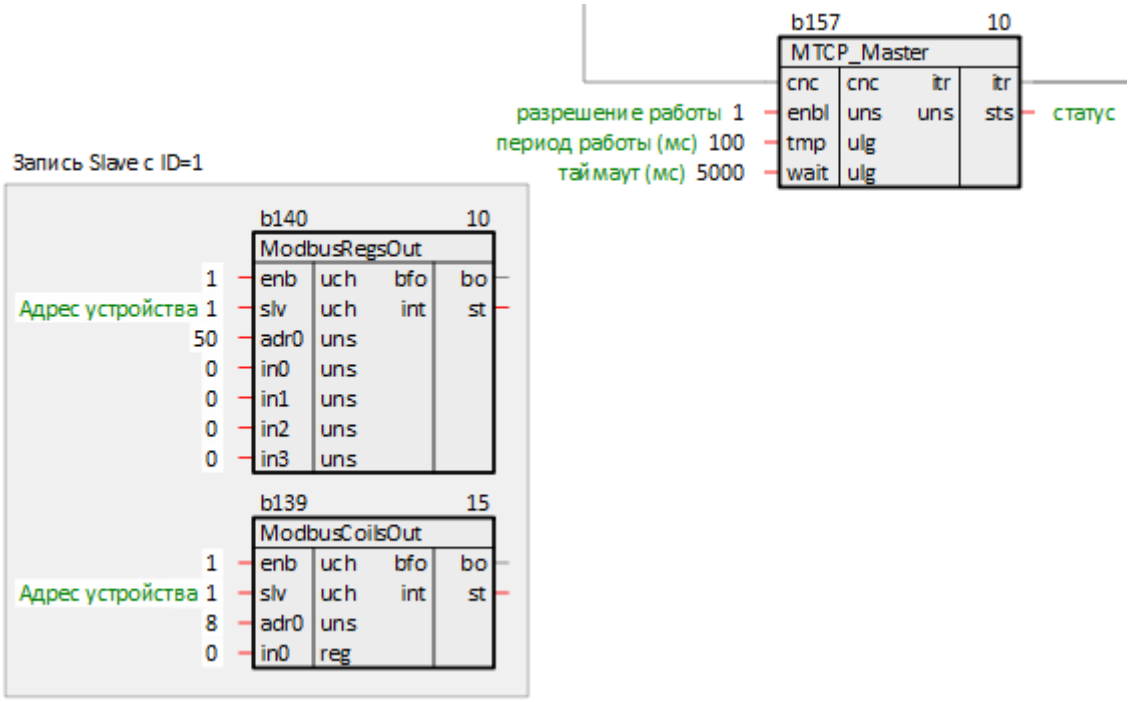


Рисунок 4.22 – Добавление и настройка блоков команд на запись

10. Создать у блока ***Modbus TCP Master*** входы типа **bfo** количеством, соответствующим количеству добавленных команд на запись. Затем соединить выходы блоков записи с добавленными входами **bo** блока ***Modbus TCP Master***.

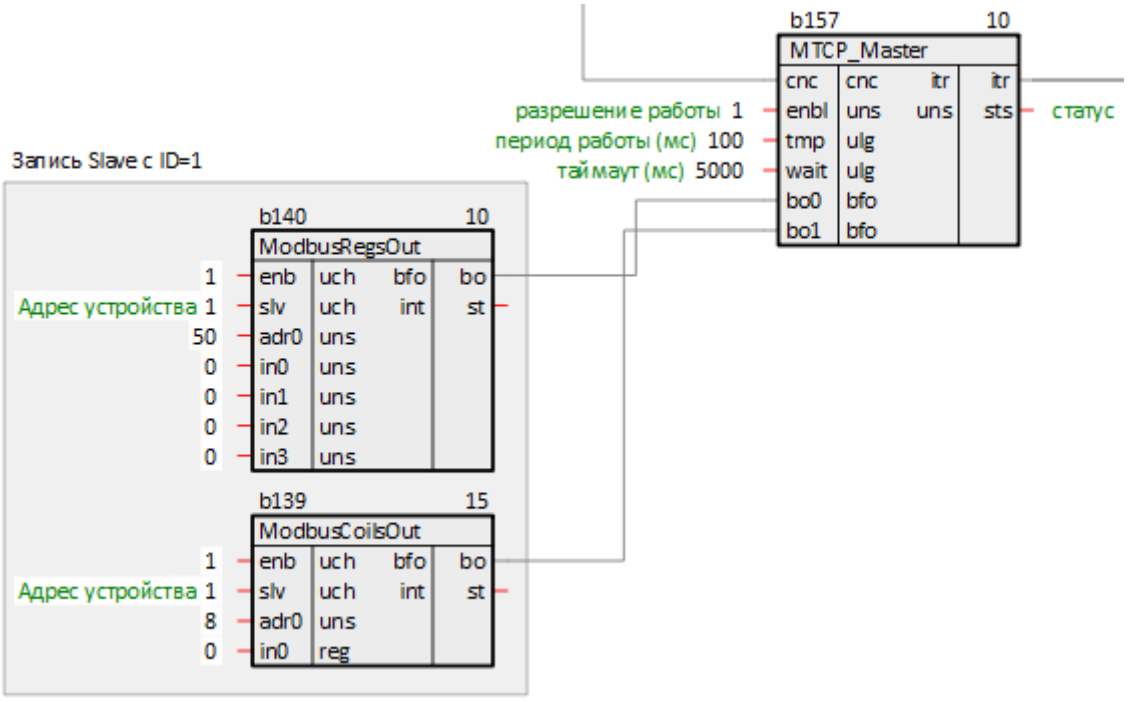


Рисунок 4.23 – Связь блоков команд записи с Modbus TCP Master

11. При необходимости соединить входы **in** блоков записи с выходами других блоков в проекте.
12. Задать порядок выполнения блоков ***По потоку данных***.

Таким образом, получится следующий вид страницы Modbus TCP Master:

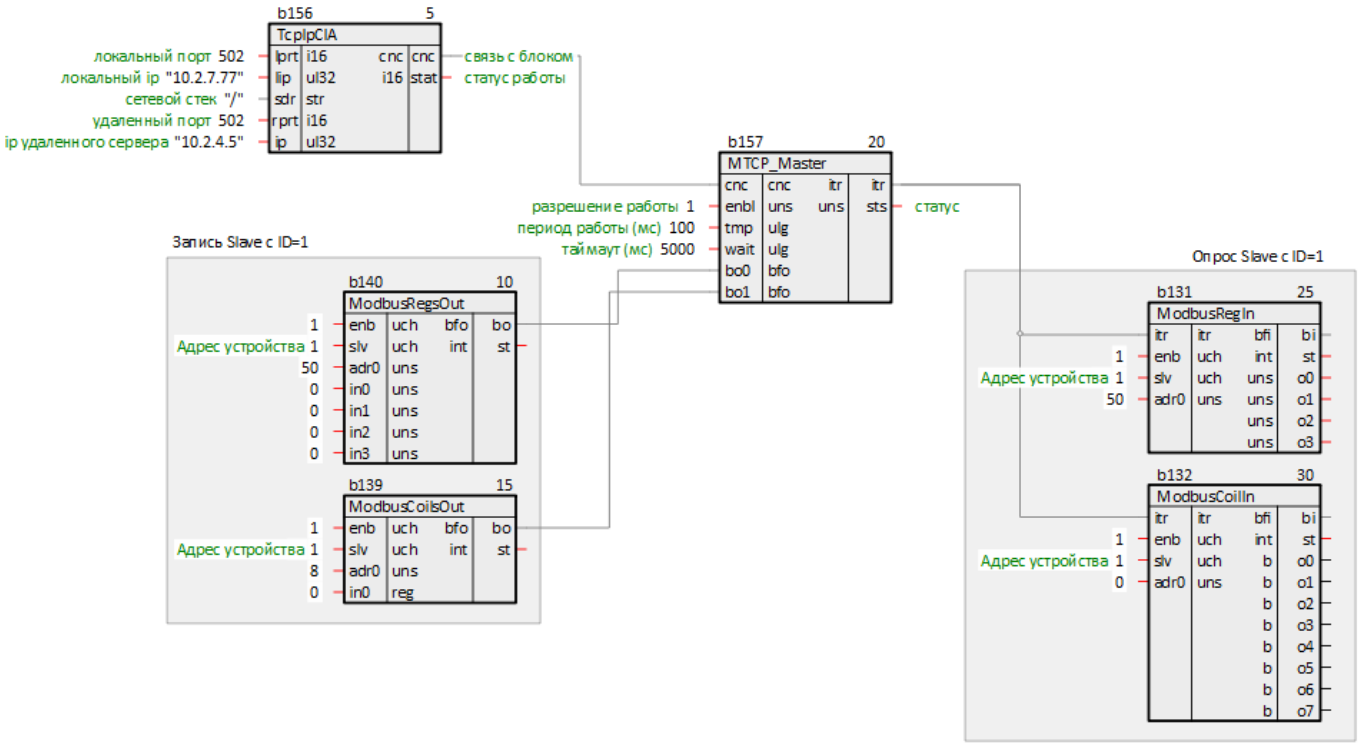


Рисунок 4.24 – Настройка Modbus TCP Master

Пример настройки ПЛК в режиме Modbus TCP Master для опроса модулей Mx210 приведен в разделе 6.3.

4.5 Настройка ПЛК в режиме Modbus TCP Slave

Для настройки ПЛК в режиме ***Modbus TCP Slave*** следует выполнить следующие действия:

1. Создать новую ***Программу*** в проекте в месте работы ***Фон*** (блоки TcpIpSrA и Modbus TCP Slave рекомендуется размещать в фоне). В свойствах созданной программы задать ***Имя*** – *Modbus* (или любое другое).
2. Создать внутри программы ***Страницу***. Добавить свойство ***Комментарии*** и задать *Modbus TCP Slave* (или любой другой).

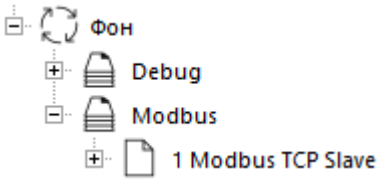


Рисунок 4.25 – Создание страницы для настройки опроса в режиме Modbus TCP Slave

3. Создать на странице блок TCP/IP-сервера TcpIpSrA из библиотеки ***paCore***. Задать локальные порт и IP-адрес TCP-сервера.

   К блоку TCP-сервера можно подключить до **20** клиентов.

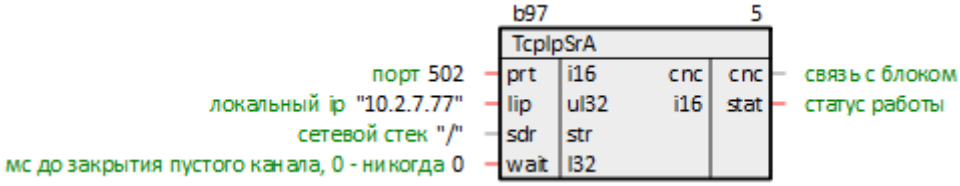


Рисунок 4.26 – Добавление блока TCP/IP-сервера

При настройке блока ***TcpIpSrA*** удобно использовать некоторые свойства модуля. Для этого можно использовать технологию SQL-запросов. Это позволяет изменять IP адрес и порт в одном месте, и использовать эти значения в разных частях проекта.

Запрос IP адреса (prop_ip):

```
"<sql>SELECT value FROM blocks_prop WHERE indx=:module AND type="prop_ip"</sql>"
```

Запрос пользовательского свойства ***Пользовательское свойство 00*** (prop_0):

```
<sql> SELECT value FROM blocks_prop WHERE indx=:module AND type="prop_0"</sql>
```

4. Создать на странице блок Modbus TCP Slave из библиотеки ***paModbus***. Задать настройки Slave-устройства.

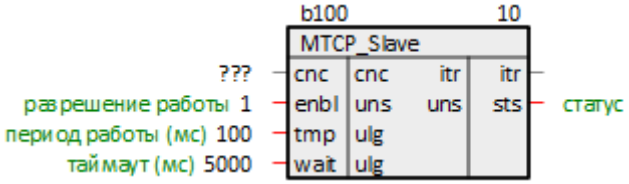


Рисунок 4.27 – Добавление блока Modbus TCP Slave

5. Соединить выход блока ***TcpIpSrA*** **cnc** с соответствующим входом блока ***Modbus TCP Slave***.

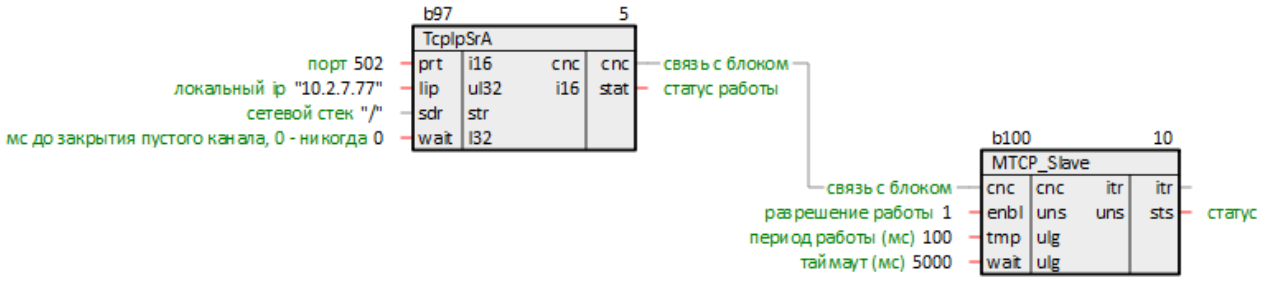


Рисунок 4.28 – Связь блока TCP/IP-сервера с Modbus TCP Slave

6. Создать на странице блоки команд на чтение/запись данных Slave-устройства. Задать Slave ID-устройства. Задать адреса выделяемых регистров. Если необходимо и записывать, и считывать один и тот же регистр, то необходимо соединить выход блока записи с сходом блока чтения.
7. Соединить выходы блока ***Modbus TCP Slave*** **itr** с соответствующими входами блоков команд записи регистров.
8. Если требуется, можно соединить выходы **o** блоков записи с входами других блоков в проекте.
9. Создать у блока ***Modbus TCP Slave*** входы типа **bfo** количеством, соответствующим количеству добавленных команд на чтение. Затем соединить выходы блоков чтения с добавленными входами **bo** блока ***Modbus TCP Slave***.
10. Если требуется, можно соединить входы **in** блоков чтения с выходами других блоков в проекте.
11. Задать порядок выполнения блоков ***По потоку данных***.

Таким образом, получится следующий вид страницы *Modbus TCP Slave*:

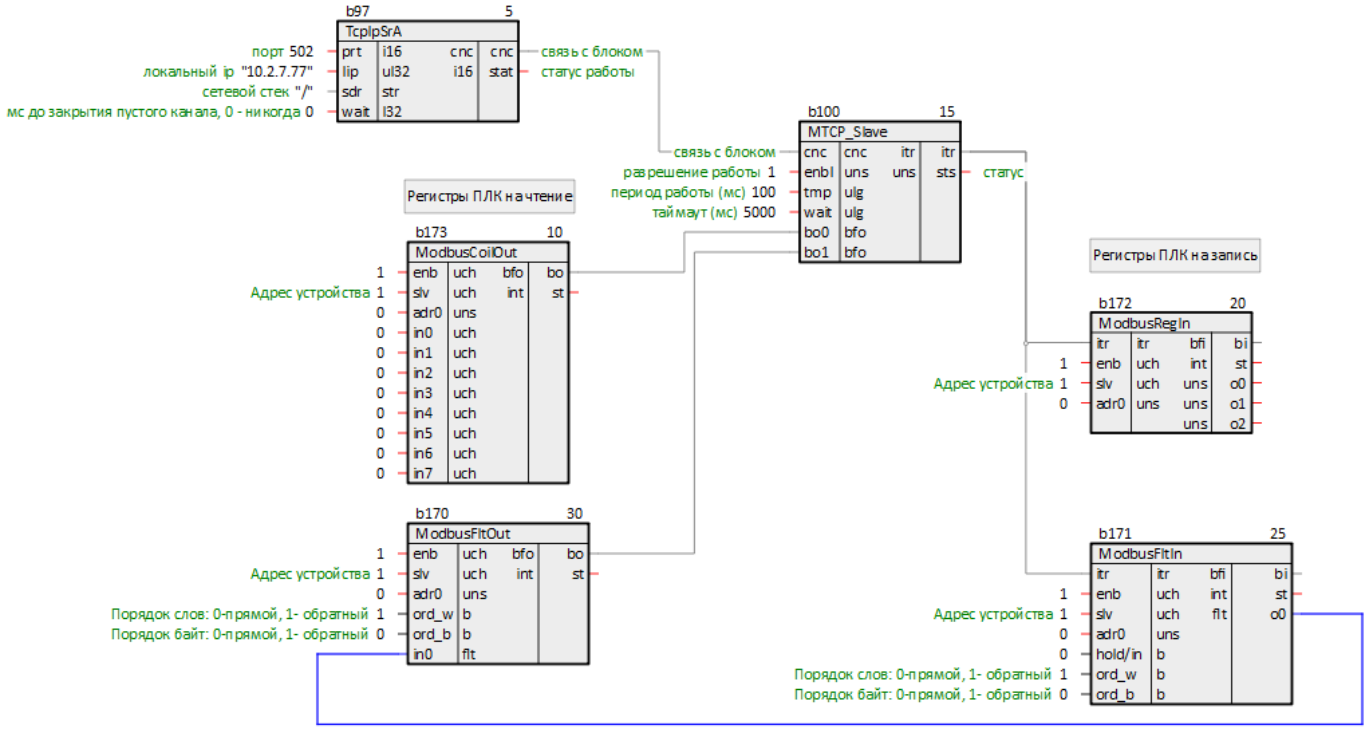


Рисунок 4.29 – Настройка Modbus TCP Slave

Пример настройки ПЛК в режиме Modbus TCP Slave для опроса Owen OPC Server приведен в разделе 6.4.

5 Запись уставок по протоколу Modbus

5.1 Запись целочисленных уставок по протоколу Modbus (BufSupEx)

Для записи уставок по протоколу Modbus используются блоки ***BufSupEx*** из библиотеки ***paCore***.

Вход **inter** блока ***BufSupEx*** подключается к выходу блока протокола Modbus TCP Slave или Modbus RTU Slave.

Мастер в сети Modbus может читать и записывать данные на диск. Для чтения блок ***BufSupEx*** реализует функцию **0x03**, для записи – **0x06** и **0x10**.

Вход **group** определяет Slave ID устройства (ID = **1** соответствует значение входа **0x100**).

Входы **dan** используются для записи уставок из программы контроллера. Для того, чтобы значение записалось на диск из программы, и его прочитал мастер сети Modbus, следует также подать импульс на соответствующий вход **czap**.

Входы **typ** определяют тип данных **dan**, при работе по Modbus могут принимать только значения **II**, **IO** (16-ти битный регистр), так как Modbus работает с целочисленными регистрами.

Про сохранение уставок с плавающей точкой по протоколу Modbus можно подробнее прочитать в разделе 5.2.

Входы **adr** определяют адреса выделяемых регистров Modbus.

Входы **min** и **max** задают минимальное и максимальное возможное значение **dan**. Если программа или мастер сети изменяет значение, то оно проверяется на условие соответствия этому диапазону.

Выходы **dan** отображают текущие значения уставок, сохраненные на диске.

Параметры на диске сохраняются в бинарных файлах с расширениями **.da1** и **.da2**.

ВНИМАНИЕ
При изменении числа входов блока ***BufSupEx*** файлы на диске перезаписываются.

Подробнее о возможностях и работе блока ***BufSupEx*** в документе Архивирование и сохранение уставок.

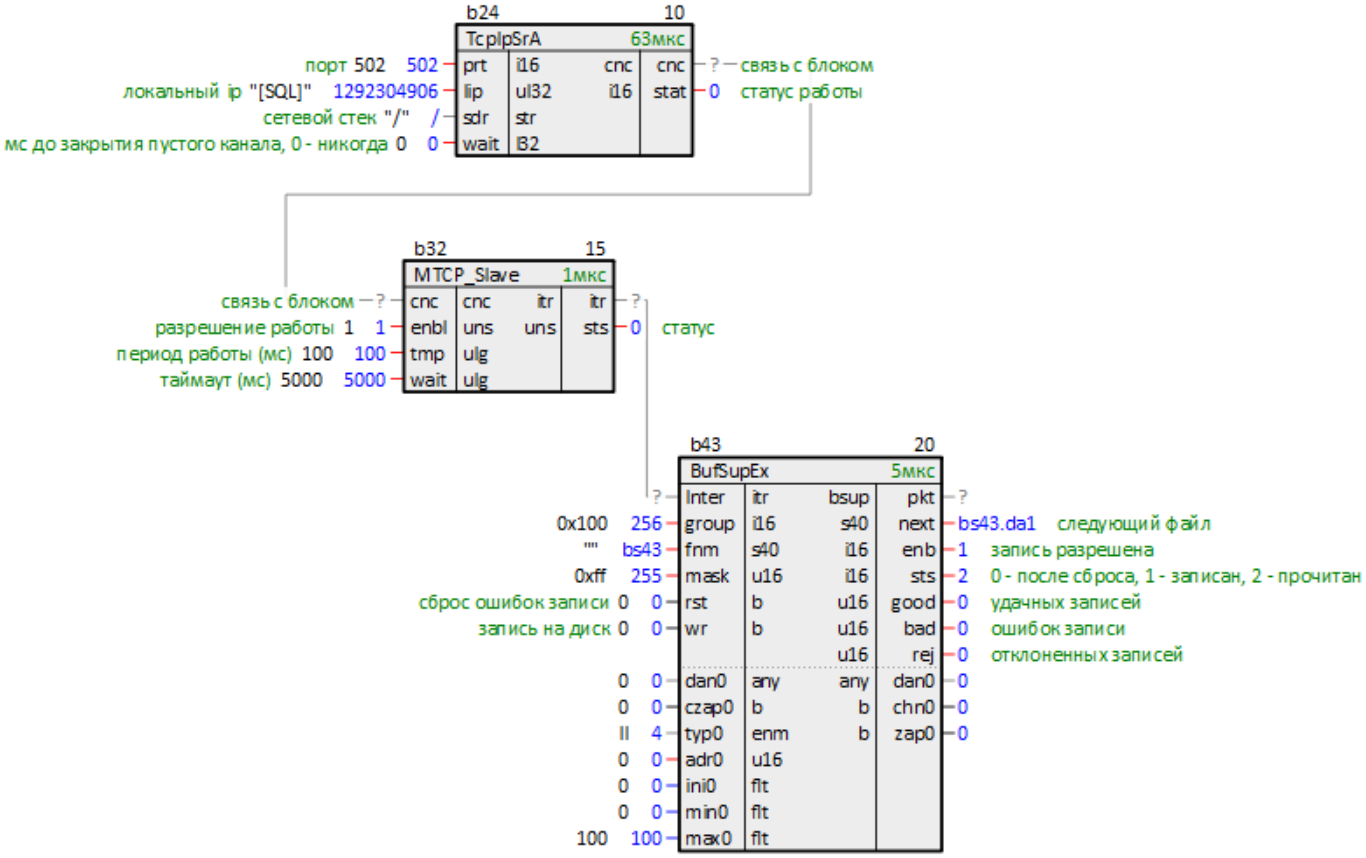


Рисунок 5.1 – Подключение BufSupEx к Modbus TCP Slave

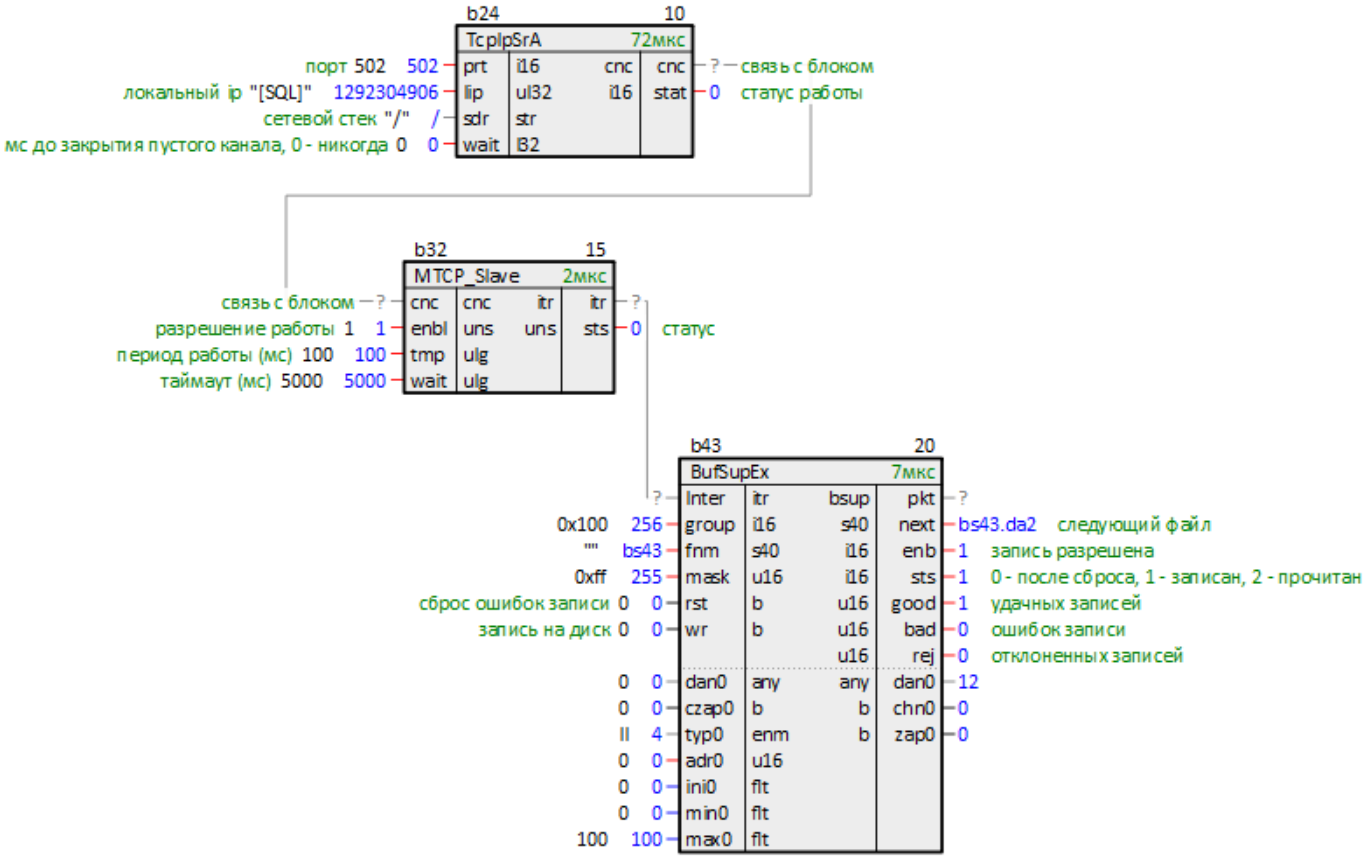


Рисунок 5.2 – Запись уставки мастером сети Modbus

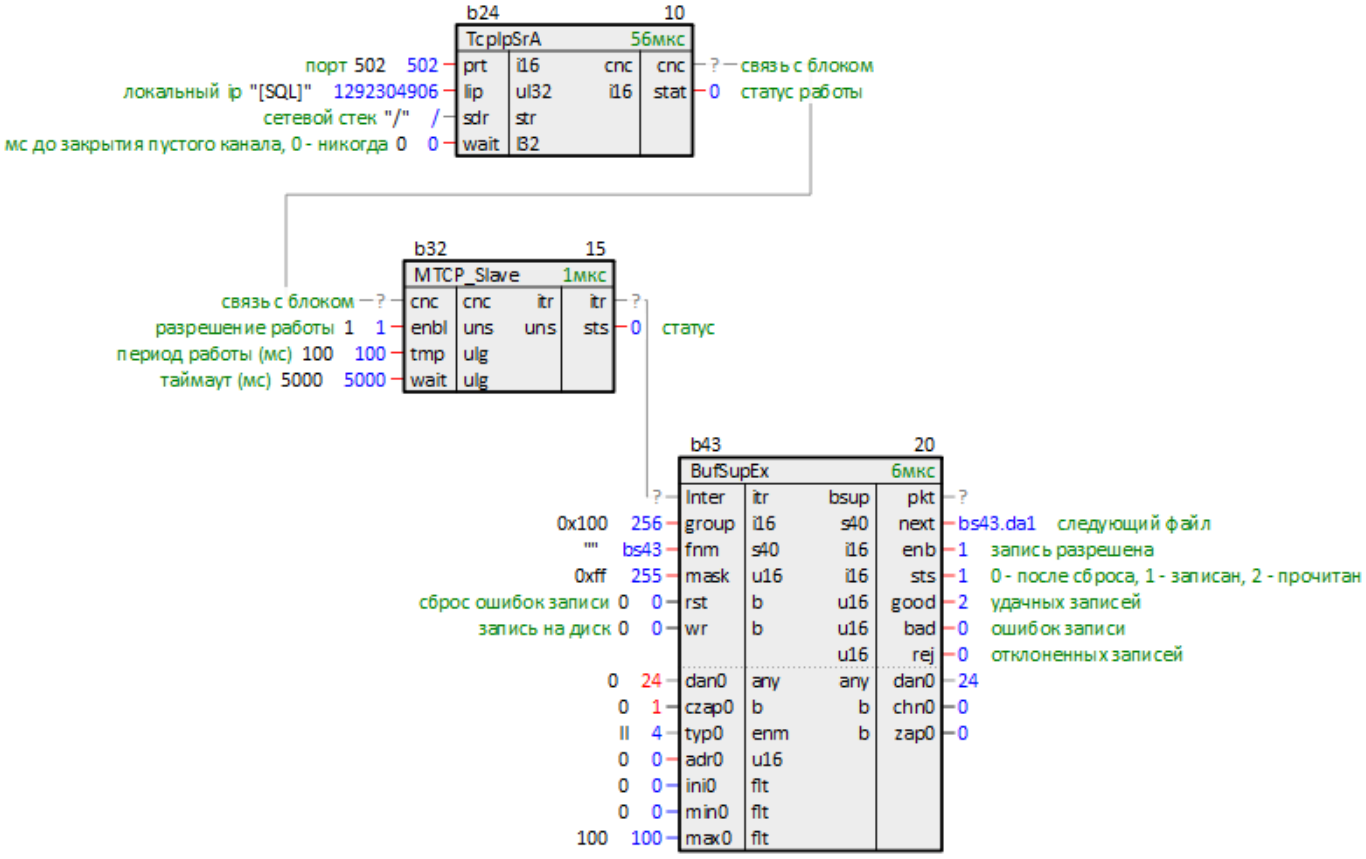


Рисунок 5.3 – Запись уставки из программы контроллера

5.2 Запись уставок с плавающей точкой по протоколу Modbus (BufSupFltEx)

Для записи уставок по протоколу Modbus используются блоки ***BufSupFltEx***.

Вход **inter** блока ***BufSupFltEx*** подключается к выходу блока протокола Modbus TCP Slave или Modbus RTU Slave.

Мастер в сети Modbus может читать и записывать данные на диск. Для чтения блок ***BufSupFltEx*** реализует функцию **0x03**, для записи – **0x10**.

Вход **group** определяет Slave ID устройства (ID = **1** соответствует значение входа **0x100**).

Входы **dan** используются для записи уставок из программы контроллера. Для того, чтобы значение записалось на диск из программы, и его прочитал мастер сети Modbus, следует также подать импульс на соответствующий вход **czap**.

Входы **typ** определяют тип данных **dan**, могут принимать только значения **AI**, **AO** (вещественное число).

Входы **adr** определяют адреса выделяемых регистров Modbus. Для опроса каждого значения **dan** выделяется два регистра Modbus.

Входы **min** и **max** задают минимальное и максимальное возможное значение **dan**. Если программа или мастер сети изменяет значение, то оно проверяется на условие соответствия этому диапазону.

Выходы **dan** отображают текущие значения уставок, сохраненные на диске.

Параметры на диске сохраняются в бинарных файлах с расширениями **.da1** и **.da2**.

ВНИМАНИЕ
При изменении числа входов блока ***BufSupFltEx*** файлы на диске перезаписываются.

Подробнее о работе блока ***BufSupFltEx*** в разделе 3.5.

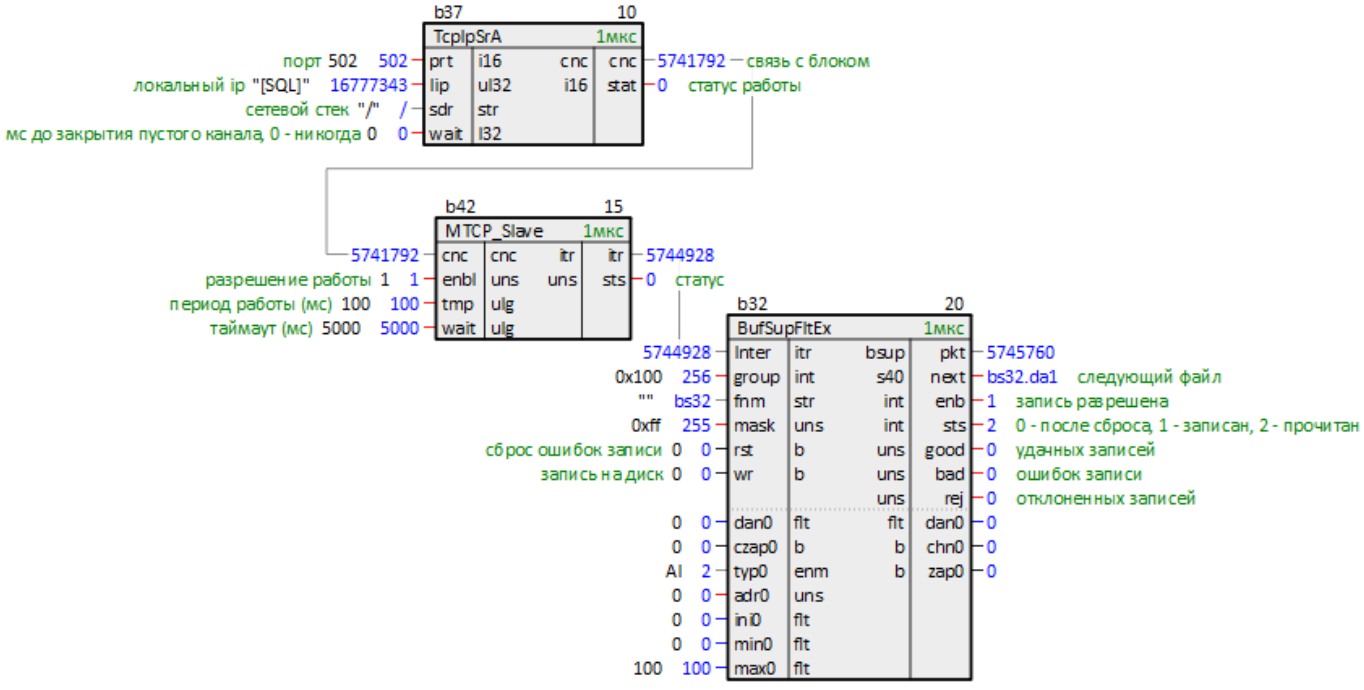


Рисунок 5.4 – Подключение BufSupFltEx к Modbus TCP Slave

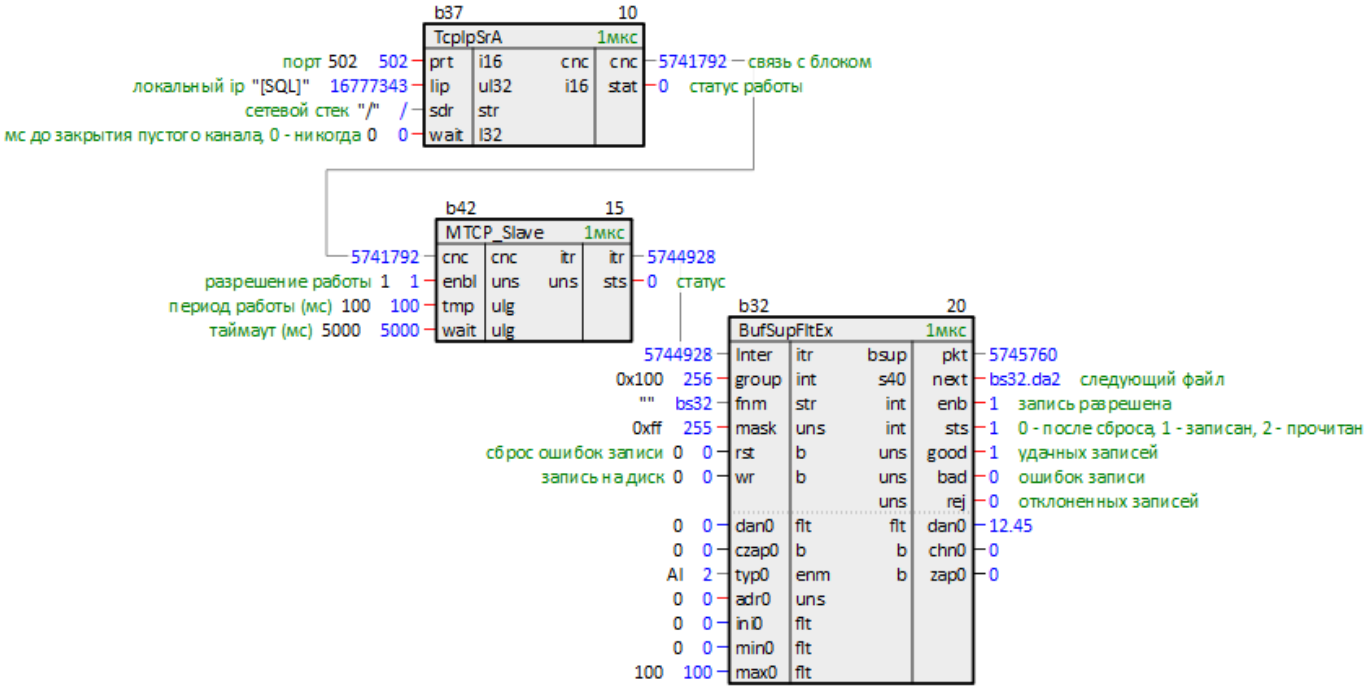


Рисунок 5.5 – Запись уставки мастером сети Modbus

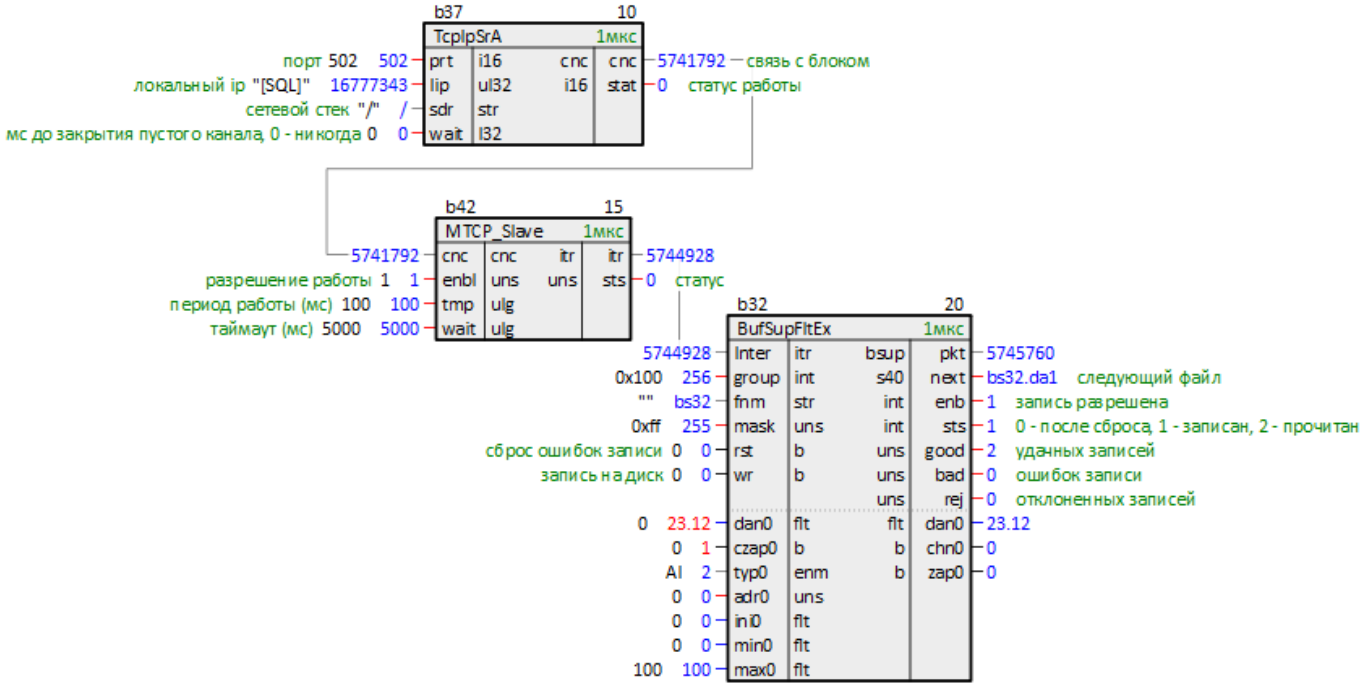


Рисунок 5.6 – Запись уставки из программы контроллера

6 Примеры настройки обмена по протоколу Modbus

6.1 ПЛК210 (Modbus RTU Master) и модули Mx110

В качестве примера будет рассмотрена настройка обмена с модулями Mx110 (МВ110-8А, МВ110-16Д и МУ110-8Р).

Реализуемый алгоритм: если значение первого аналогового входа модуля МВ110-8А превышает 30 и при этом первый дискретный вход модуля МВ110-16Д замкнут, то первый дискретный выход МУ110-8Р замыкается с задержкой 3 секунды. Во всех остальных случаях дискретный выход МУ110-8Р разомкнут.

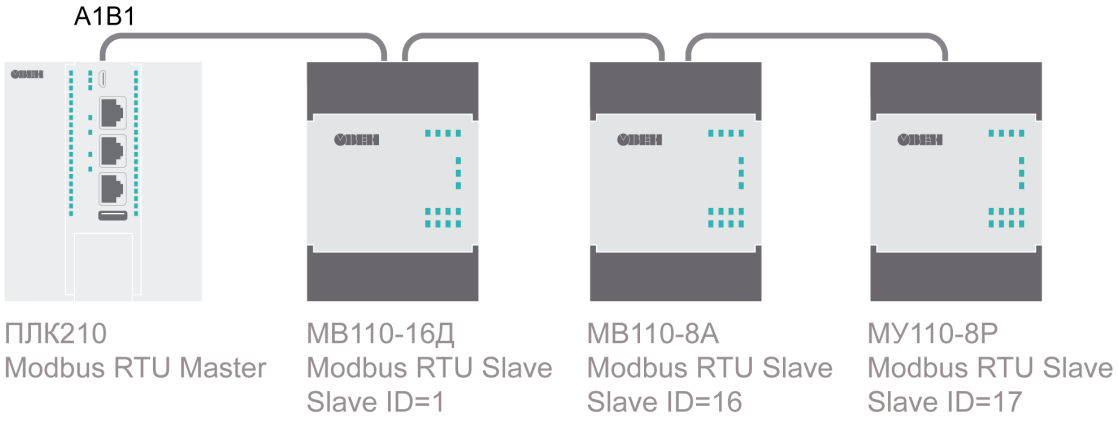


Рисунок 6.1 – Структурная схема примера

Пример создан в среде **Полигон** и подразумевает запуск на **ПЛК210** с прошивкой **3.x**.

Если требуется запустить проект на другом устройстве, следует изменить свойства ***ОС*** и ***Тип процессорной платы*** в окне ***Свойства*** модуля *MB_RTU_master_Mx110* на необходимые.

Пример доступен для скачивания по ссылке. Пароль для доступа к отладчику – **1**.

Таблица 6.1 – Сетевые параметры устройств в примере

Параметр	МВ110-16Д	МВ110-8А	МУ110-8Р
COM-порт ПЛК, к которому подключен модуль	A1B1		
Адрес модуля	1	16	17
Скорость обмена	9600		
Количество бит данных	8		
Контроль четности	Нет		
Количество стоп-бит	1		

Таблица 6.2 – Регистры модулей в примере

Модуль	Номер регистра DEC	Тип в устройстве	Функция Modbus	Описание
МВ110-8А	4, 5	FLOAT 32	0x03	Значение температуры со входа 1
МВ110-16Д	51	UINT 16	0x03	Битовая маска входов
МУ110-8Р	50	UINT 16	0x10	Битовая маска выходов

Для настройки обмена следует:

1. Настроить модули **Mx110** с помощью программы **ОВЕН Конфигуратор/Конфигуратор М110** в соответствии с таблицей 6.1. Подключить модули к контроллеру в соответствии с рисунком 6.1.

2. Создать новый проект **Полигон** (в примере — файл с именем *MB_RTU_master_Mx110*). Добавить в проект библиотеку ***paModbus***.
3. Добавить в место работы ***Фон*** программу с именем *Modbus_RTU_Master*.
4. Внутри программы добавить четыре ***Страницы***, в свойстве ***Комментарии*** которых указать соответственно *COM-порт*, *MB110-8A*, *MB110-16Д* и *МУ110-8Р*.

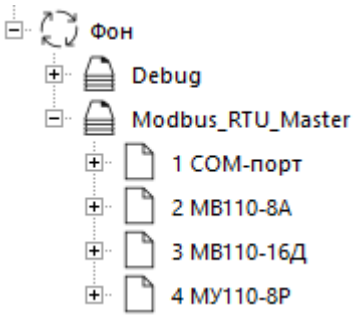


Рисунок 6.2 – Дерево проекта

5. Внутри страницы *COM-порт* создать блок 210-RS485 из библиотеки ***paOwenIO***. На входах блока задать значения в соответствии с таблицей 6.1.

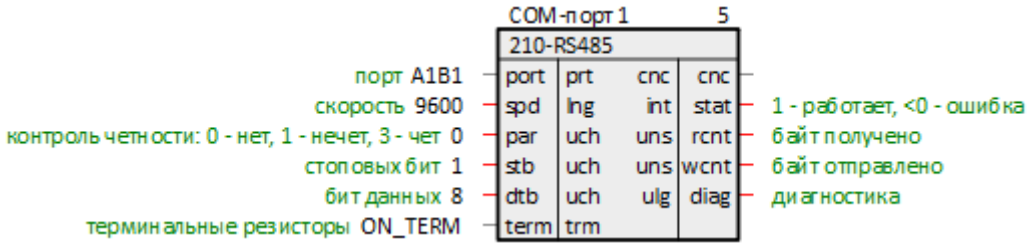


Рисунок 6.3 – Настройка блока COM-порт

6. Далее создать блок Modbus RTU Master. Соединить вход с соответствующим выходом блока ***210-RS485***.

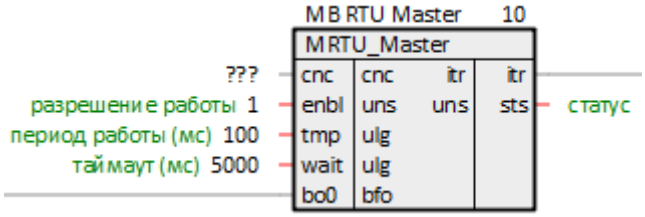


Рисунок 6.4 – Настройка блока Modbus RTU Master

7. Внутри страницы *MB110-8A* создать блок чтения значений с плавающей запятой ModbusFltIn. На входах блока задать значения в соответствии с таблицами 6.1 и 6.2.

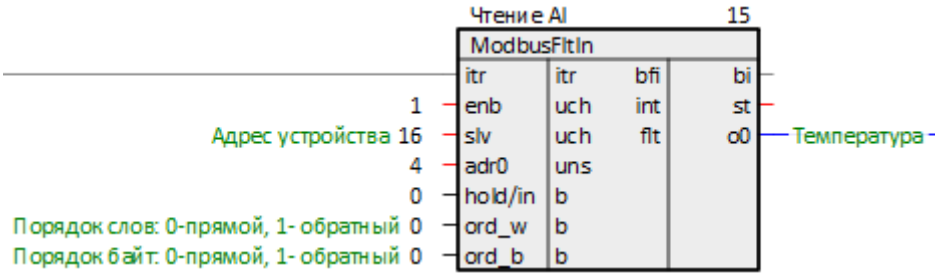


Рисунок 6.5 – Настройка блока ModbusFltIn

8. Внутри страницы *MB110-16Д* создать блок для чтения значений с дискретных входов ***ModbusRegIn***. На входах блока задать значения таблицами 6.1 и 6.2.
9. Создать блок выбора 16 битов из регистра ***FromReg16*** из библиотеки ***paCore*** и соединить выход блока ***ModbusRegIn*** **o0** с входом **reg** блока ***FromReg8***. Добавить к первому выходу блока ***FromReg8*** комментарий – *Вход MB110-16Д*.

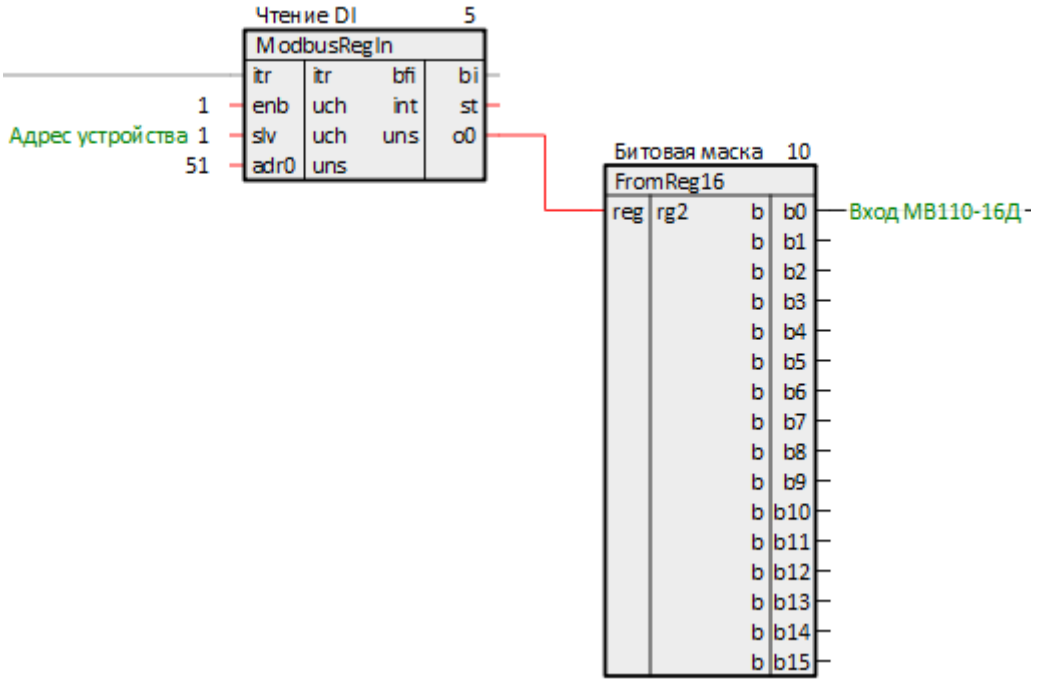


Рисунок 6.6 – Страница МВ110-16Д

10. Поставить на странице порядки ***По потоку данных***.
11. Внутри страницы *МУ110-8Р* создать блок для записи значений на дискретные выходы ***ModbusRegsOut***. На входах блока задать значения в соответствии с таблицами 6.1 и 6.2.
12. Создать блок объединения 8 битов в регистр ***ToReg8*** из библиотеки ***paCore*** и соединить вход блока ***ModbusRegsOut*****in0** с входом **reg** блока ***ToReg8***. Добавить к первому входу блока ***ToReg8*** комментарий – *Выход МУ110-8Р*.

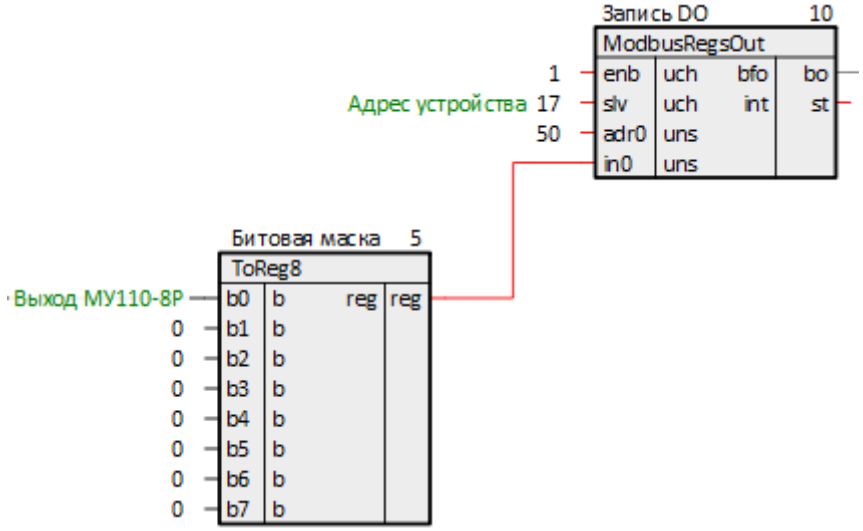


Рисунок 6.7 – Страница МУ110-8Р

13. Поставить на странице порядки ***По потоку данных***.
14. Соединить выход блока ***ModbusRTU Master*** **itr** с соответствующими входами блоков чтения на страницах *МВ110-8А* и *МВ110-16Д*.
15. Создать вход у блока ***Modbus RTU Master*** **bo0**, соединить его с соответствующим выходом блока записи на странице *МУ110-8Р*.
16. Создать в месте работы ***Таймер*** программу с названием *Mx110*.
17. Внутри программы создать страницы с комментариями *Значения с модулей* и *Обработка значений*.

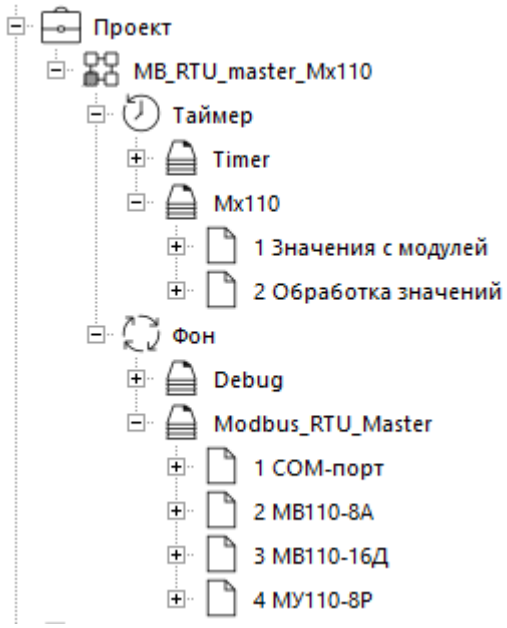


Рисунок 6.8 – Дерево программы

18. На странице *Значения с модулей* создать блоки ***TransBit*** и ***TransFlt*** из библиотеки ***paCore***.

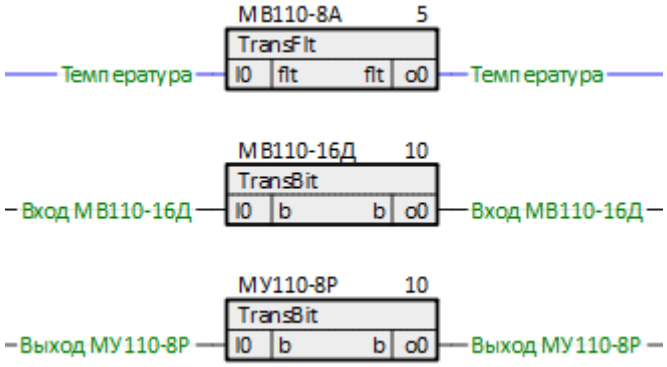


Рисунок 6.9 – Блоки TransBit и TransFlt

19. Соединить входы блоков со страницы *Значения с модулей* с выходами блоков со страниц *MB110-8A* и *MB110-16Д*, как показано на рисунках ниже.

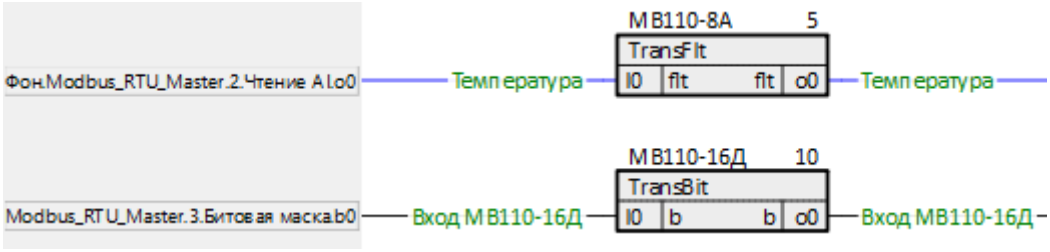


Рисунок 6.10 – Соединение блоков (Значения с модулей)

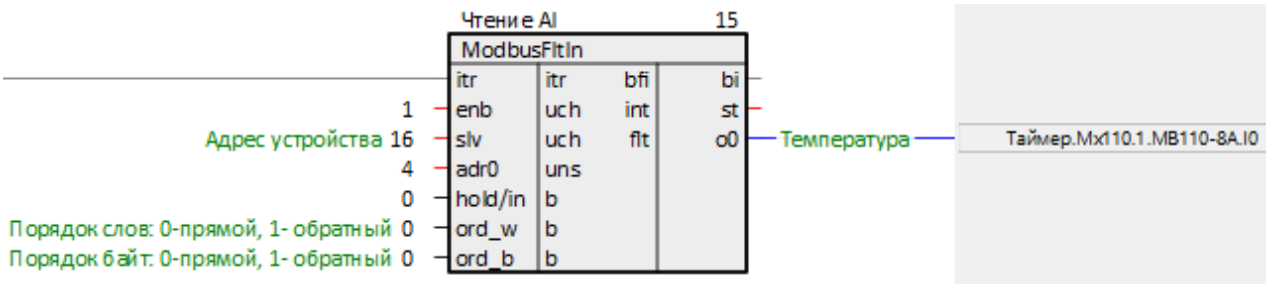


Рисунок 6.11 – Соединение блоков (MB110-8A)

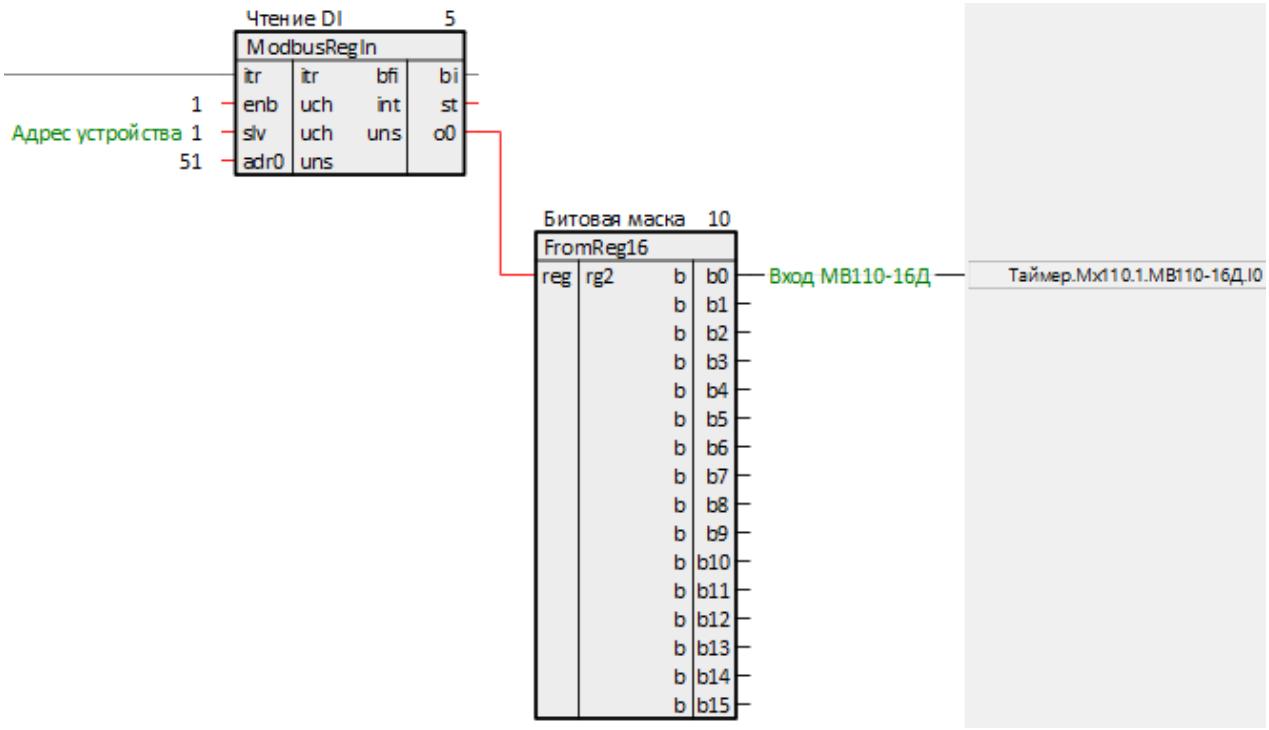


Рисунок 6.12 – Соединение блоков (МВ110-16Д)

20. Соединить выход блока со страницы *Значения с модулей* с входами блоков со страницы *МУ110-8Р*, как показано на рисунках ниже.

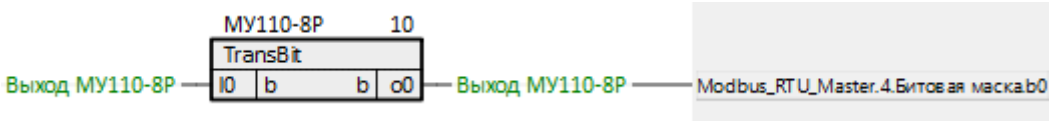


Рисунок 6.13 – Соединение блоков (Значения с модулей)

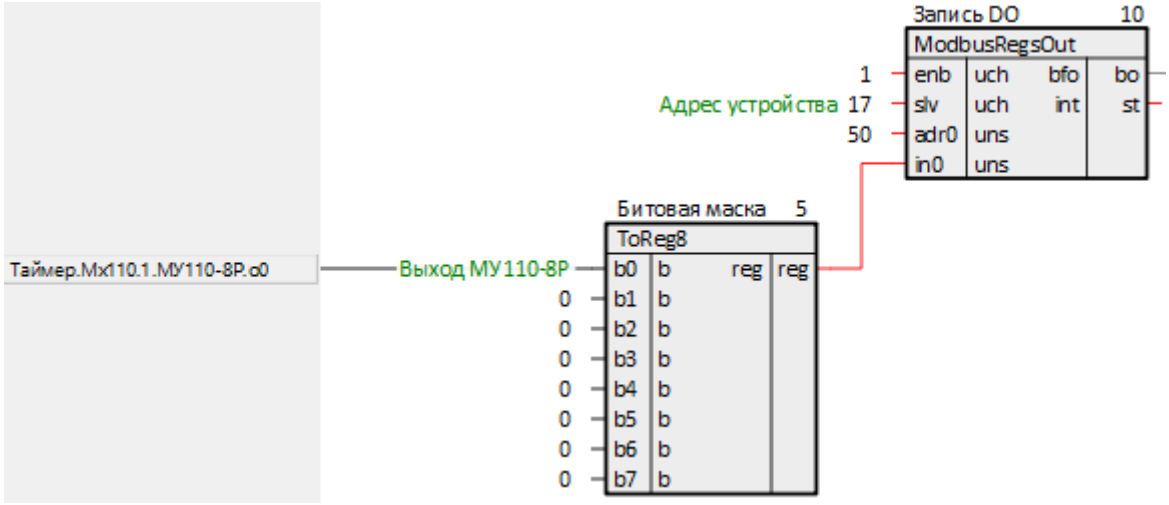


Рисунок 6.14 – Соединение блоков (МУ110-8А)

21. На странице *Обработка значений* создать блоки ***Cmpr***, ***AND*** и ***DelayOn*** из библиотеки ***paCore***. Соединить их и настроить в соответствии с рисунком ниже.

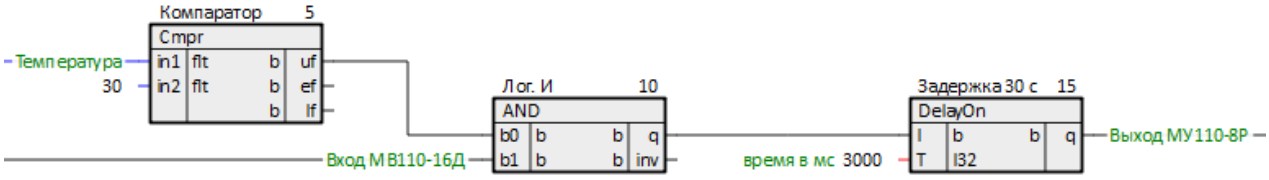


Рисунок 6.15 – Страница Обработка значений

22. Соединить входы блоков ***Cmpr*** и ***AND*** с выходами блоков со страницы *Значения с модулей*, как показано на рисунках ниже.

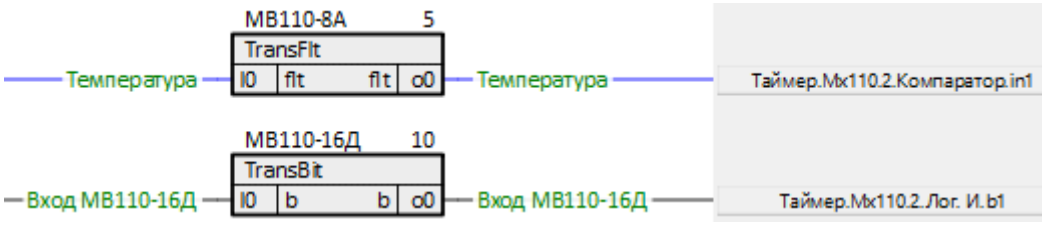


Рисунок 6.16 – Соединение блоков (Значения с модулей) с блоками Cmpr и AND

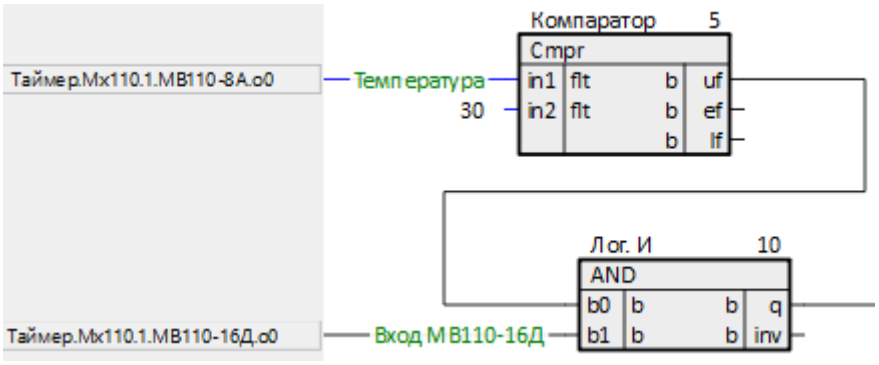


Рисунок 6.17 – Соединение блоков (Обработка значений)

23. Соединить выход блока ***DelayOn*** с входом блока со страницы *Значения с модулей*, как показано на рисунках ниже.

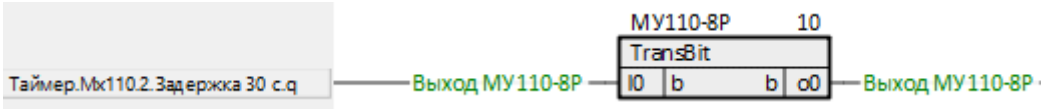


Рисунок 6.18 – Соединение блока (Значения с модулей) с блоком DelayOn

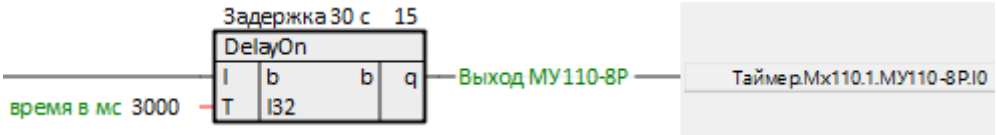


Рисунок 6.19 – Соединение блока (Обработка значений)

Для наладки работы собранной системы в примере используется окно представления ***График***. Для настройки графика необходимо сделать следующее.

24. Создать в модуле ***Раздел*** с именем *График*.

25. Добавить в раздел ***График*** выходы блоков ***TransBit*** и ***TransFlt*** – *Температура*, Вход МВ110-16Д и *Выход МУ110-8Р*. На странице *Значения с модулей* данные выходы должны подсветиться желтым.

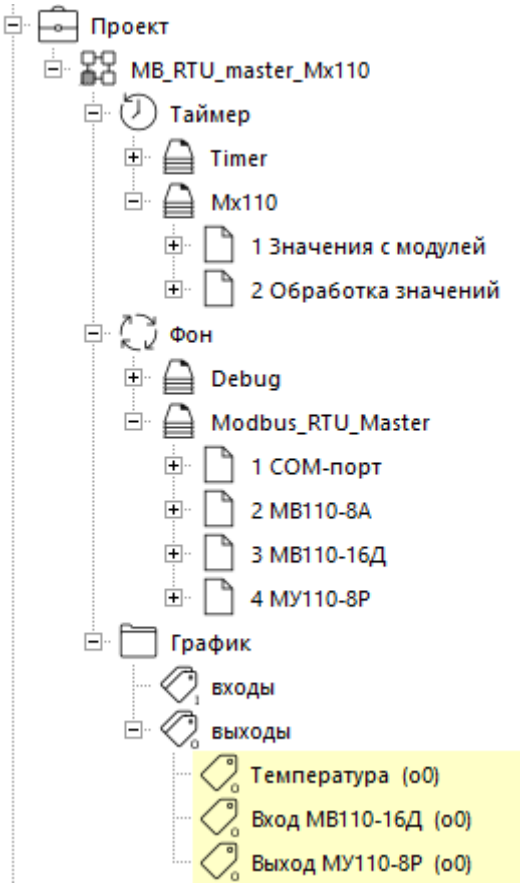


Рисунок 6.20 – Дерево проекта

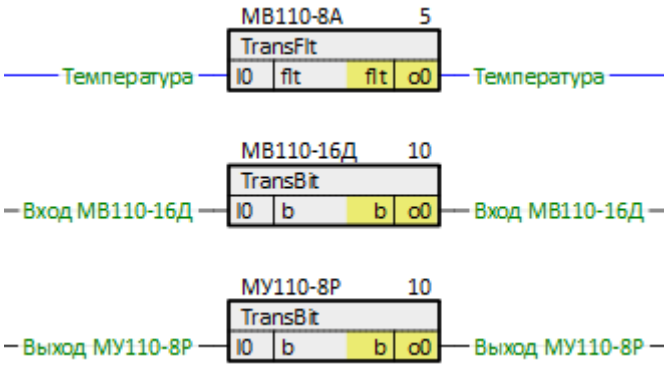


Рисунок 6.21 – Блоки TransBit и TransFlt с подсвеченными выходами

26. Открыть окно представления ***График*** и перетащить созданный раздел в верхнее поле окна.

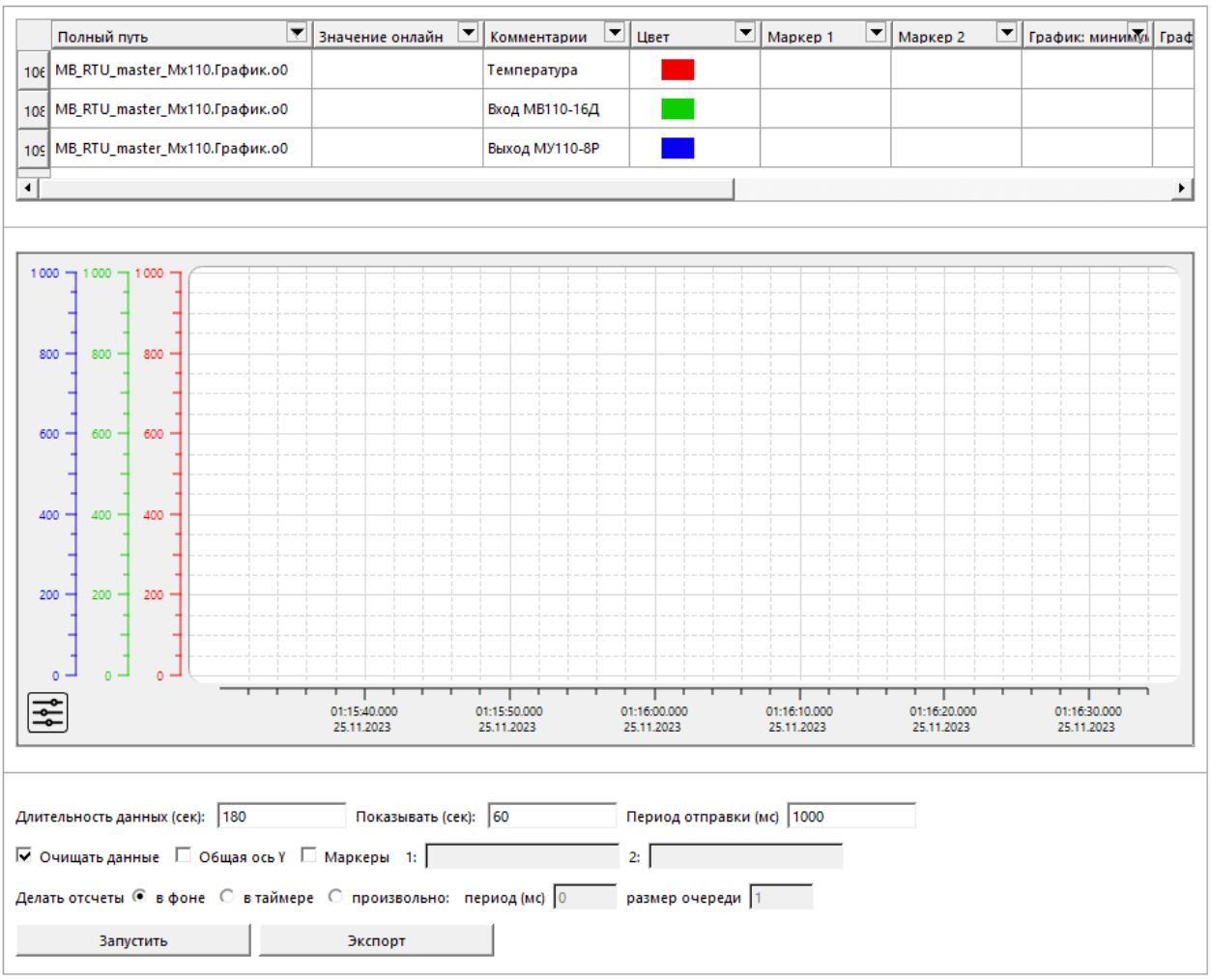


Рисунок 6.22 – Окно представления график

27. Запустить проект на контроллере, запустить отладчик и открыть график. Корректная работа системы показана на рисунке ниже.

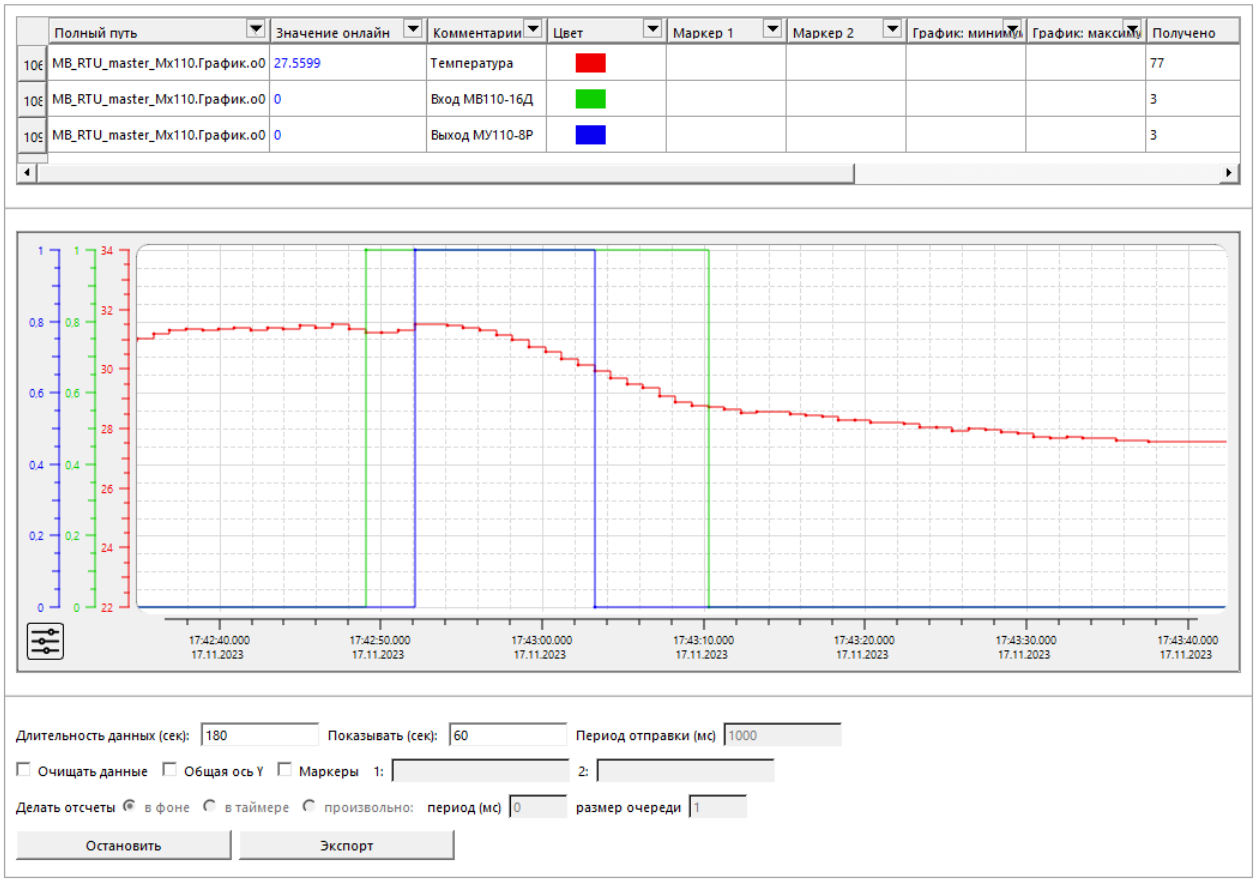


Рисунок 6.23 – Работа программы

6.2 ПЛК210 (Modbus RTU Slave) и Owen OPC Server

В качестве примера будет рассмотрена настройка обмена с Owen OPC Server, который будет использоваться в режиме **Modbus RTU Master**.

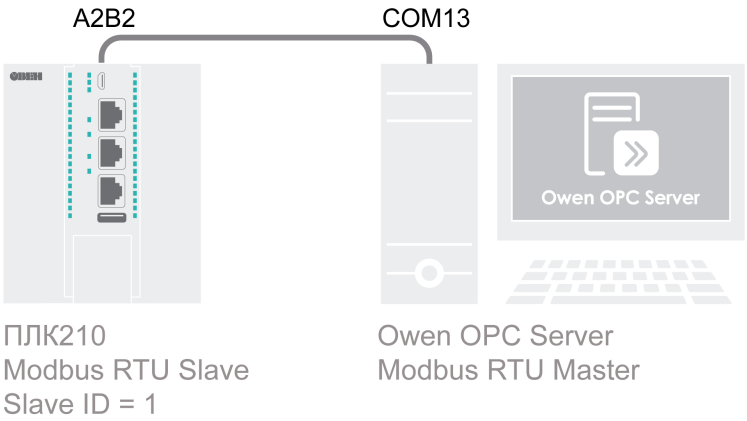


Рисунок 6.24 – Структурная схема примера

Пример создан в среде **Полигон** и подразумевает запуск на **ПЛК210** с прошивкой **3.x**.

Если требуется запустить проект на другом устройстве, следует изменить свойства ***ОС*** и ***Тип процессорной платы*** в окне ***Свойства*** модуля *MB_RTU_slave_Owen_OPC_server* на необходимые.

Пример доступен для скачивания по ссылке. Пароль для доступа к отладчику – **1**.

Таблица 6.3 – Сетевые параметры устройств в примере

Параметр	ПЛК210	Owen OPC Server
Режим работы	Slave	Master
COM-порт	A2B2	COM13

Продолжение таблицы 6.3

Параметр	ПЛК210	Owen OPC Server
Slave ID	1	-
Скорость обмена	115200	
Количество бит данных	8	
Контроль четности	Нет	
Количество стоп-бит	1	

Таблица 6.4 – Регистры/флаги ПЛК в примере

Адрес регистра/флага	Тип в устройстве	Область памяти
0	WORD	Holding Registers
0	BOOL	Coils
1, 2	REAL	Holding Registers

Для настройки обмена следует:

1. Подключить контроллер и ПК в соответствии с рисунком 6.24.
2. Создать новый проект **Полигон** (в примере с именем *MB_RTU_slave_Owen_OPC_server*). Добавить в проект библиотеку ***paModbus***.
3. Добавить в место работы ***Фон*** программу с именем *Modbus_RTU_Slave*.
4. Внутри программы добавить две ***Страницы***, в свойстве ***Комментарии*** которых указать соответственно *Modbus RTU Slave* и *Регистры Modbus*.

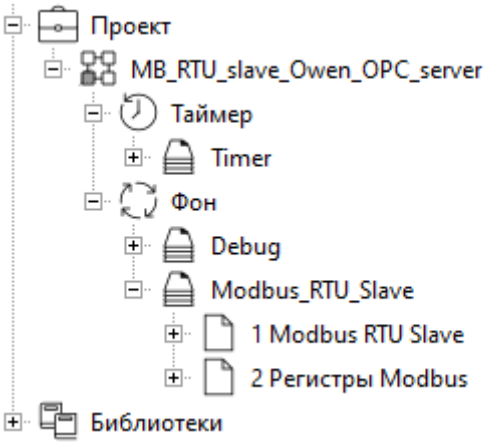


Рисунок 6.25 – Дерево проекта

5. Внутри страницы *Modbus RTU Slave* создать блок 210-RS485 из библиотеки ***paOwenIO***. На входах блока задать значения в соответствии с таблицей 6.3.

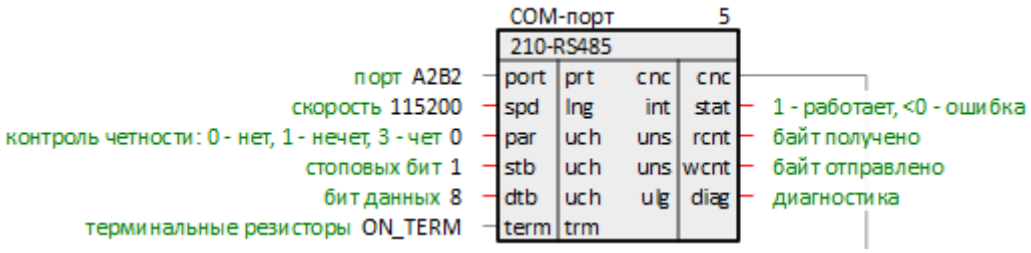


Рисунок 6.26 – Настройка блока COM-порта

6. Далее создать блок Modbus RTU Slave. Соединить вход **cnc** с соответствующим выходом блока ***210-RS485***.

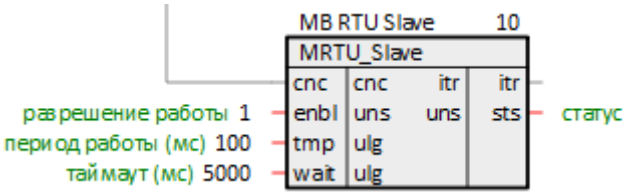


Рисунок 6.27 – Настройка блока Modbus RTU Slave

7. Затем создать блоки записи регистров в ПЛК (в соответствии с таблицей 6.4). Соединить входы блоков **itr** с советствующим выходом блока ***Modbus RTU Slave***.

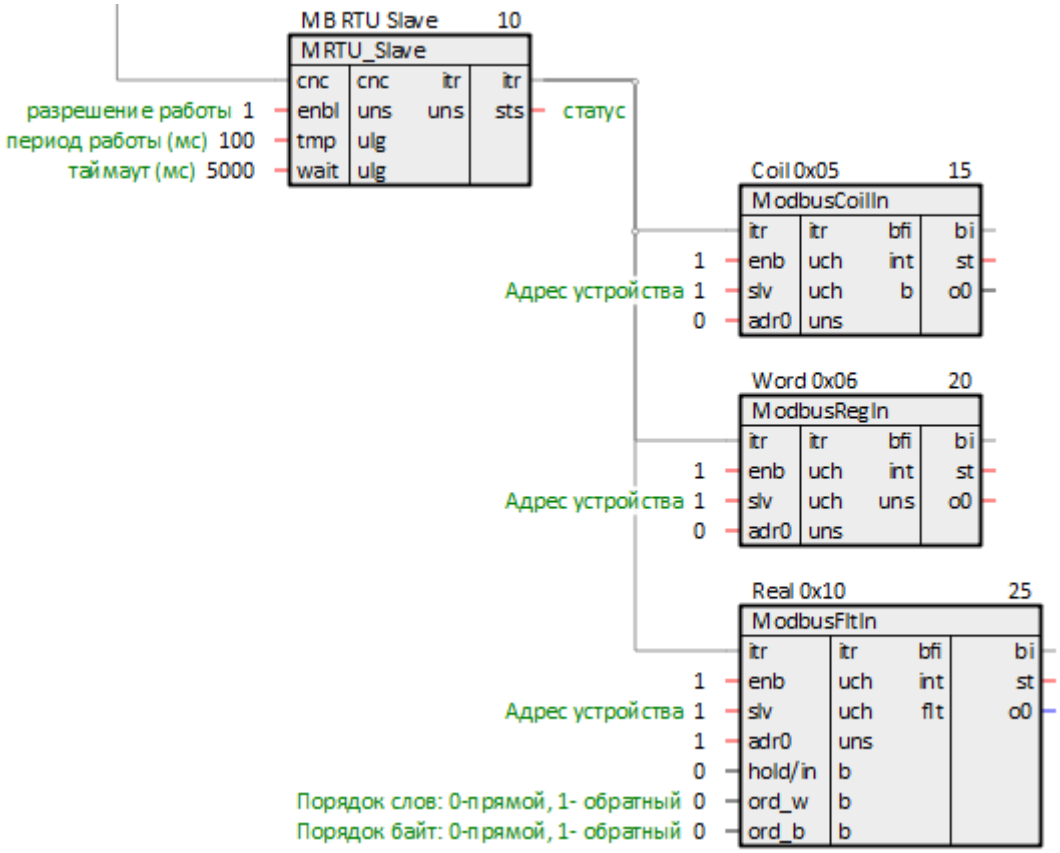


Рисунок 6.28 – Настройка блоков записи

8. Создать блоки чтения регистров из ПЛК (в соответствии с таблицей 6.4). Создать три входа **bo** у блока ***Modbus RTU Slave***. Соединить их с соответствующими выходами блоков чтения.

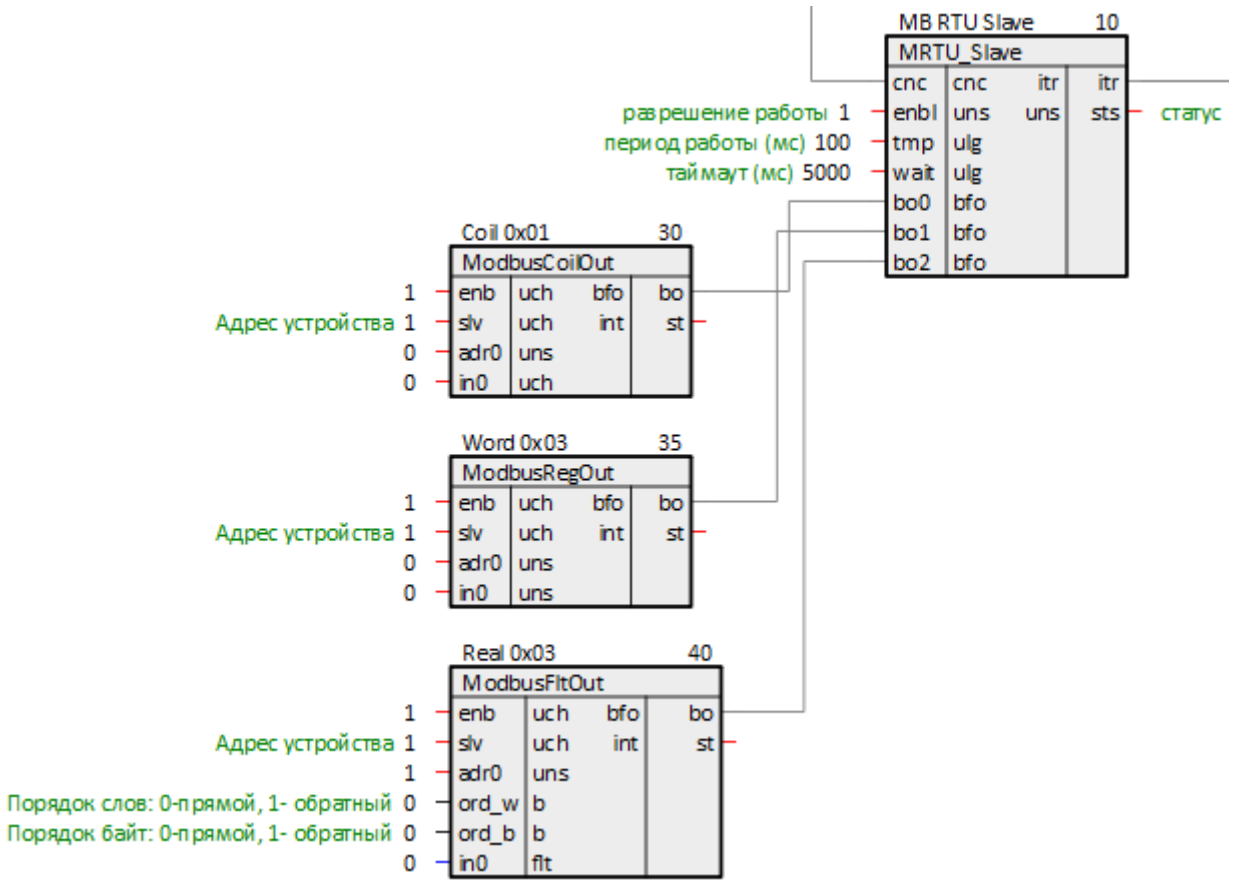


Рисунок 6.29 – Настройка блоков чтения

9. Для того, чтобы одновременно читать и записывать одни и те же значения мастером сети, следует соединить выходы блоков записи **o** со входами блоков чтения **in**. Для удобства в примере используются скрытые связи.

   Для создания скрытой связи следует в свойствах выхода **o** добавить свойства ***Полный алиас*** и ***Глобальная константа***. В свойстве ***Полный алиас*** задать имя новой константы.

   Повторить те же действия для всех блоков записи на странице.

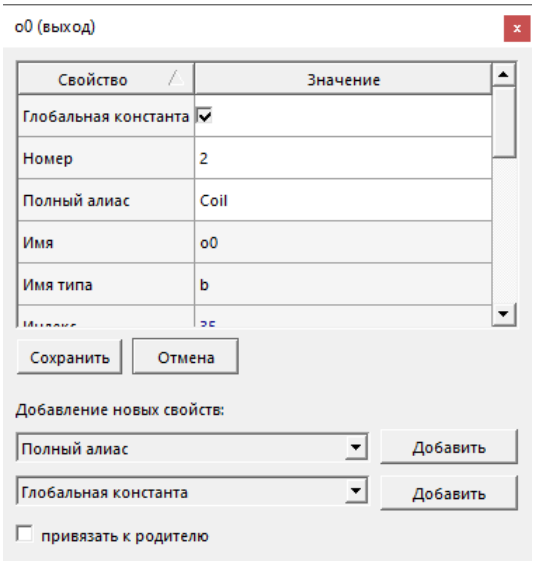


Рисунок 6.30 – Создание скрытой связи

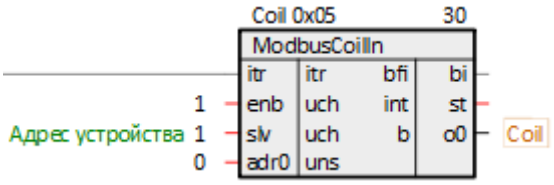


Рисунок 6.31 – Блок записи с константным выходом

10. У соответствующих входов блоков чтения **in** правой кнопкой мыши задать созданные глобальные константы.

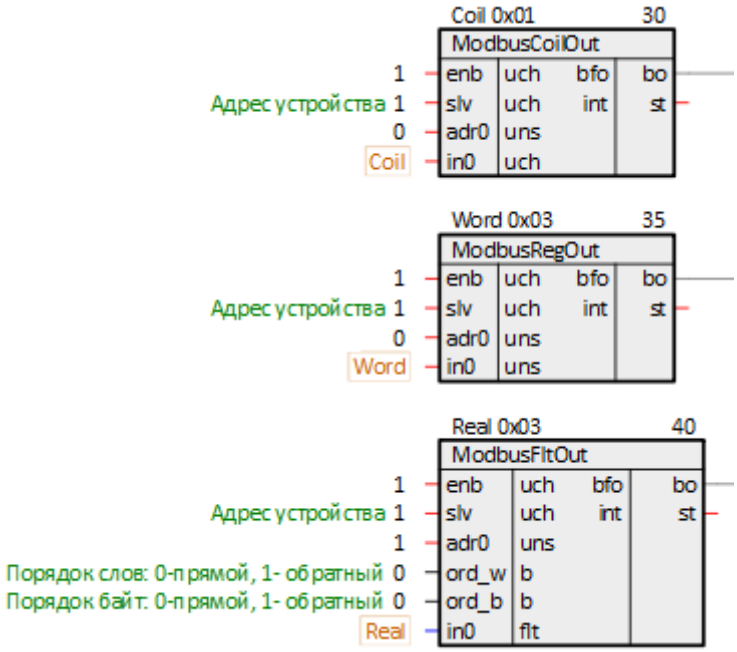


Рисунок 6.32 – Создание скрытой связи

11. Поставить на странице порядки ***По потоку данных***.

Таким образом, вид страницы *Modbus RTU Slave* примет вид как на рисунке ниже.

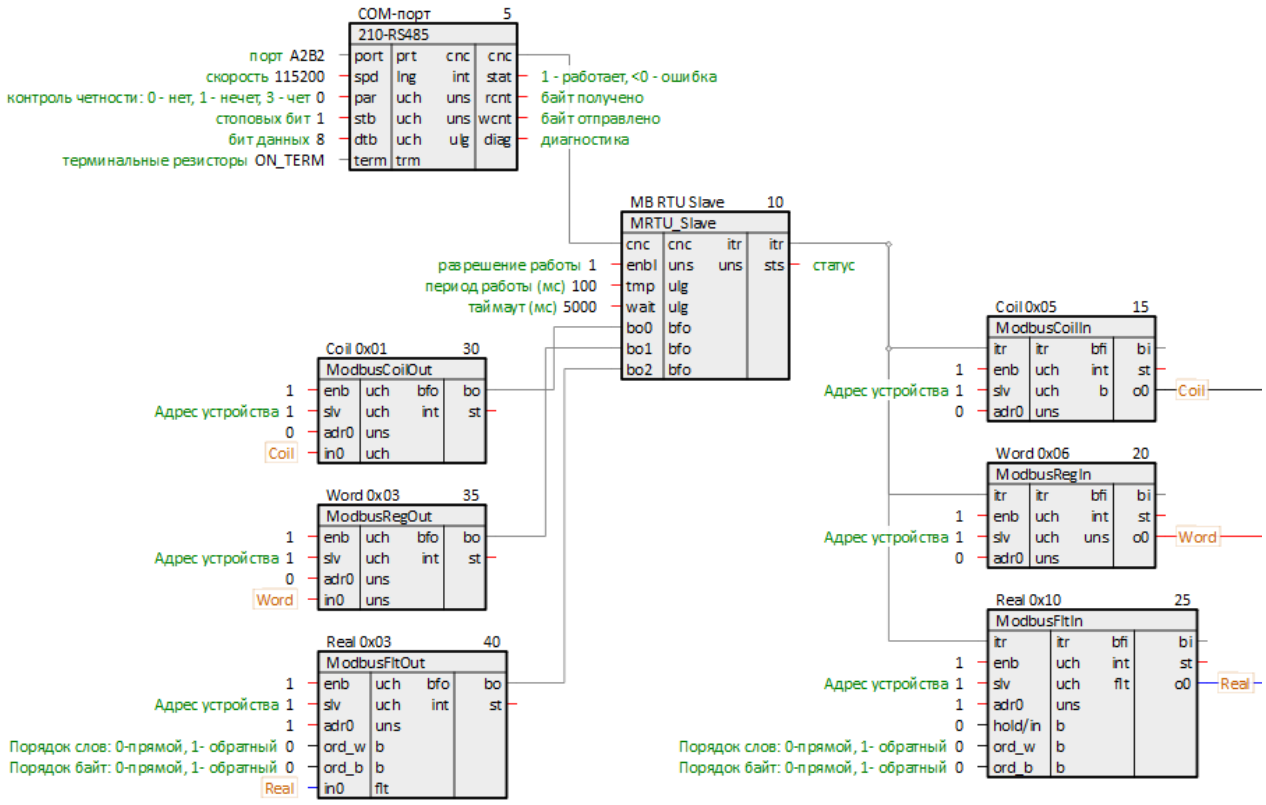


Рисунок 6.33 – Вид страницы Modbus RTU Slave

12. Для удобства можно вынести значения созданных регистров ПЛК на отдельную страницу *Регистры Modbus*.

    Для этого следует на странице *Регистры Modbus* создать блоки ***TransBit***, ***TransInt*** и ***TransFlt*** из библиотеки ***paCore***.

    На входы созданных блоков **I** задать созданные ранее константы. Выходы блоков **o** при необходимости соединить с другими блоками в проекте.

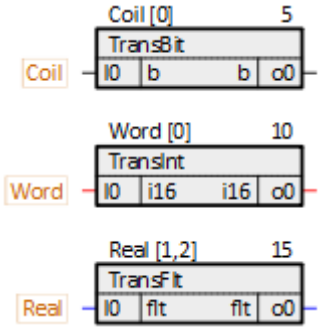


Рисунок 6.34 – Терминальные блоки

13. Установить и запустить Owen OPC Server.
14. Нажать правой кнопкой мыши на компонент ***Сервер*** и добавить узел.

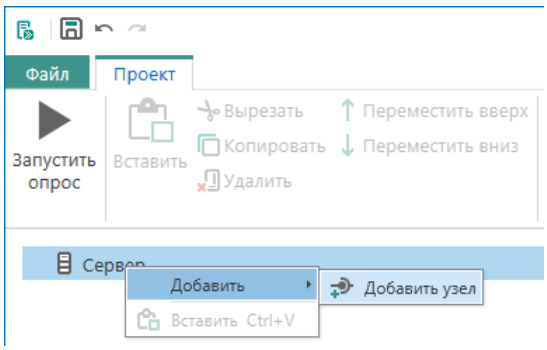


Рисунок 6.35 – Добавление узла

15. В свойствах добавленного узла задать протокол ***Modbus RTU*** и настройки интерфейса в соответствии с таблицей 6.3.

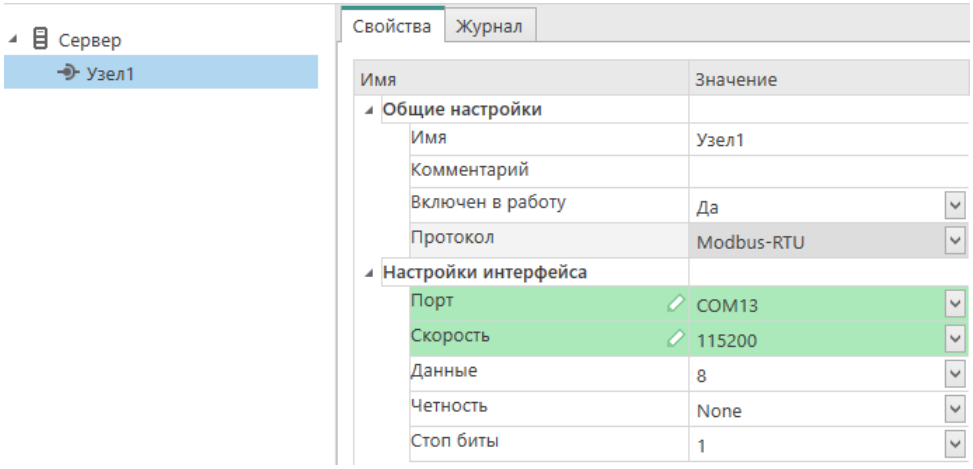


Рисунок 6.36 – Свойства узла

16. Добавить в узел ***Устройство***.

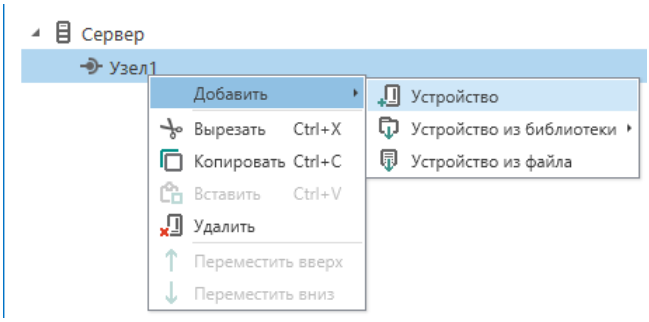


Рисунок 6.37 – Добавление устройства

17. Задать в устройстве свойства в соответствии с таблицей 6.3.

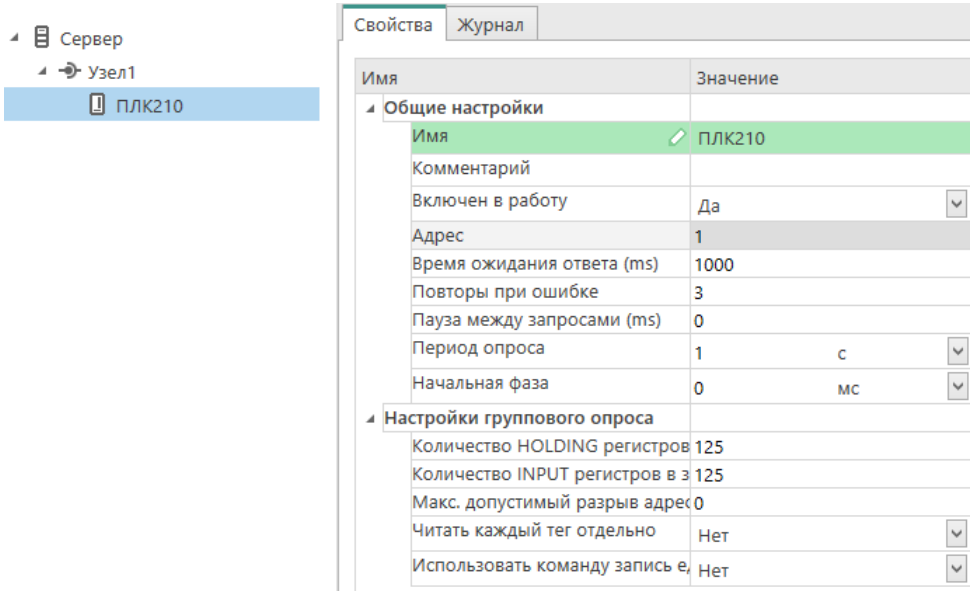


Рисунок 6.38 – Свойства устройства

18. Добавить в устройстве три ***Тега***.

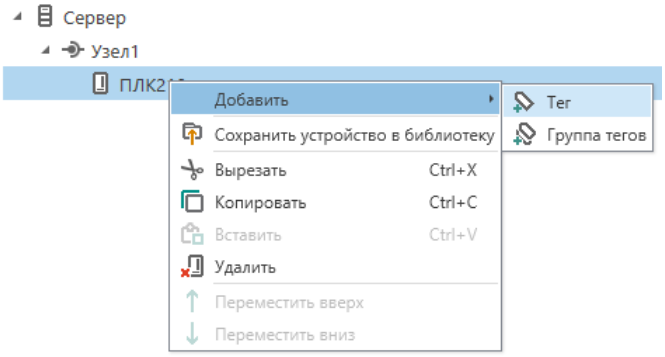


Рисунок 6.39 – Добавление тега

19. Задать созданным тегам свойства в соответствии с таблицей 6.4.

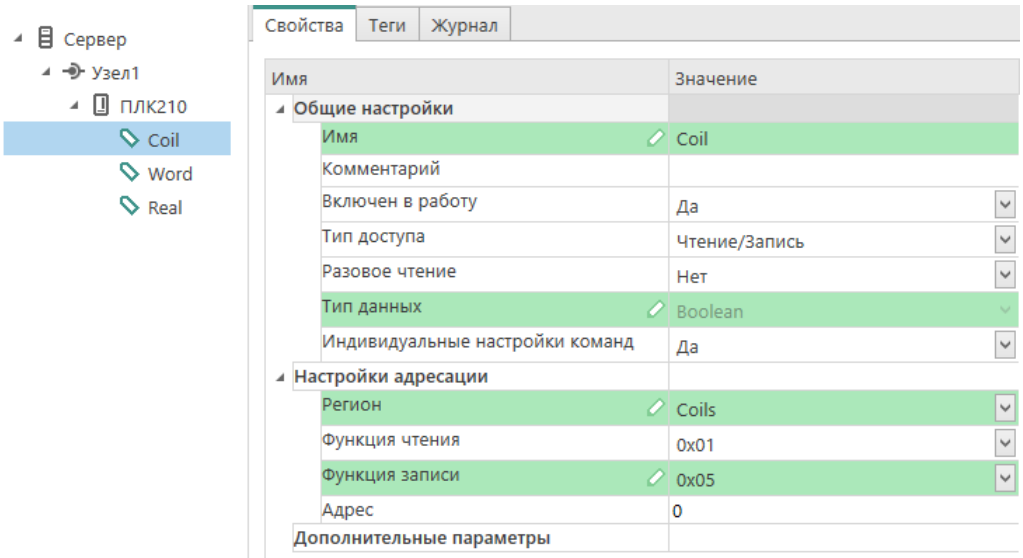


Рисунок 6.40 – Тег Coil

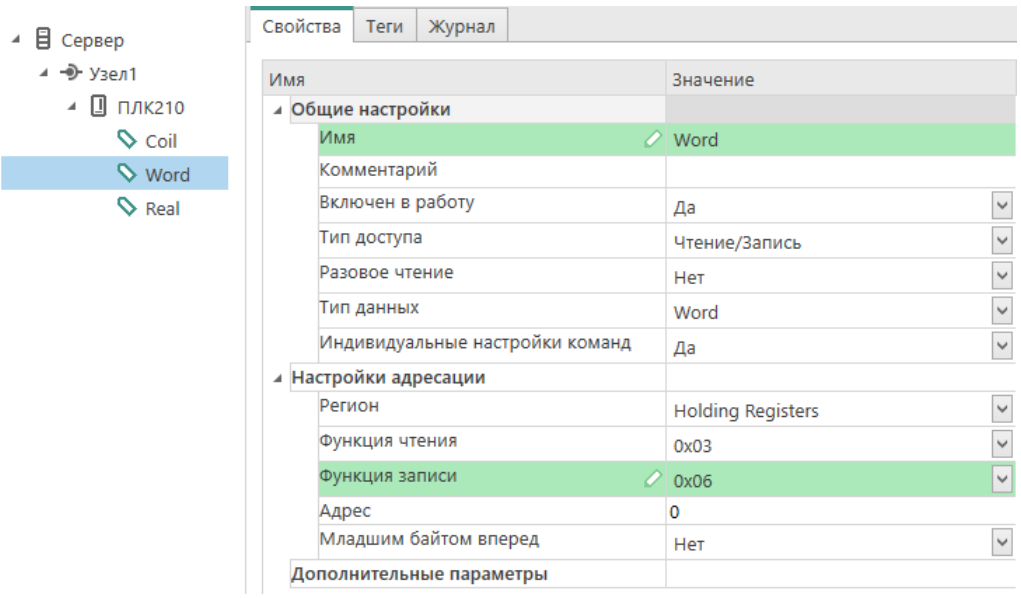


Рисунок 6.41 – Тег Word

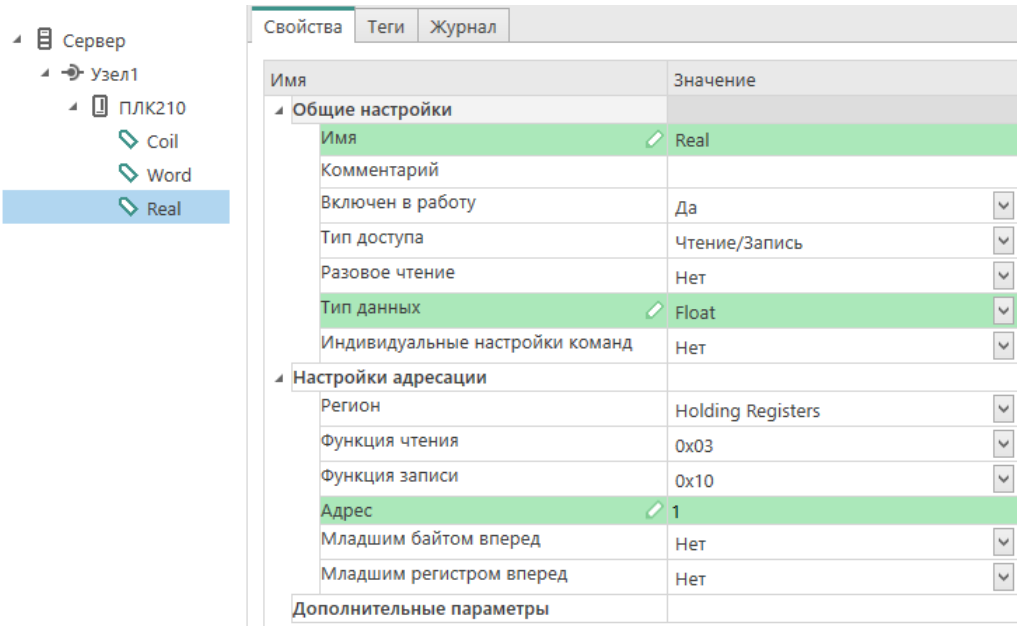


Рисунок 6.42 – Тег Real

20. Загрузить и открыть программу в ПЛК, запустить отладчик. Запустить опрос Owen OPC Server.
21. Корректный обмен данными показан на рисунках ниже.

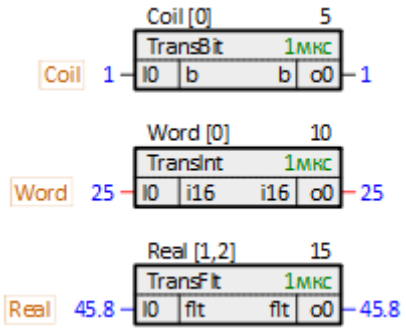


Рисунок 6.43 – Работа программы ПЛК

Теги | Устройства

Имя	Адрес	Значение	Тип данных	Качество	Комментарий
ПЛК210.Coil	Coils [0]	True	Boolean	GOOD	
ПЛК210.Word	Holding Registers [0]	25	Word	GOOD	
ПЛК210.Real	Holding Registers [1]	45,8	Float	GOOD	

Рисунок 6.44 – Работа Owen OPC Server

6.3 ПЛК210 (Modbus TCP Master) и модули Mx210

В качестве примера будет рассмотрена настройка обмена с модулями Mx210 (МВ210-101 и МК210-301).

Реализуемый алгоритм: если значение первого аналогового входа модуля МВ210-101 превышает 30 градусов, то на первом выходе модуля МК210-301 включается ШИМ-генератор. В любом другом случае выход находится в режиме переключения логического сигнала.

Первый выход МК210-301 соединяют с первым входом для отслеживания поступающих сигналов.

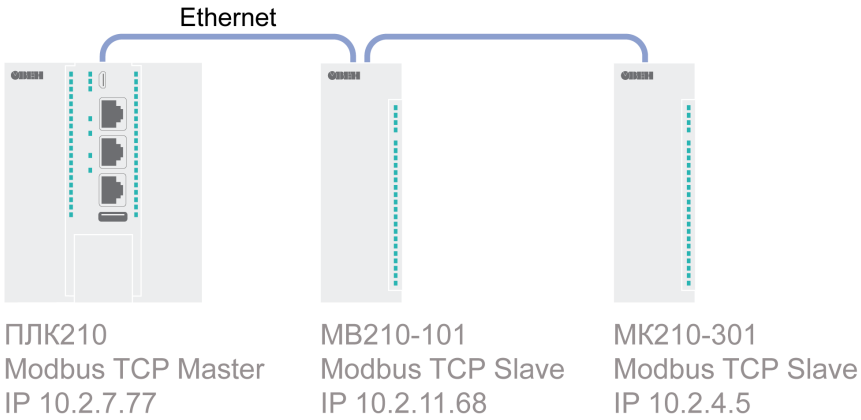


Рисунок 6.45 – Структурная схема примера

Пример создан в среде **Полигон** и подразумевает запуск на **ПЛК210** с прошивкой **3.x**.

Если требуется запустить проект на другом устройстве, следует изменить свойства ***ОС*** и ***Тип процессорной платы*** в окне ***Свойства*** модуля *MB_TCP_master_Mx210* на необходимые.

Пример доступен для скачивания по ссылке. Пароль для доступа к отладчику – **1**.

Таблица 6.5 – Сетевые параметры устройств в примере

Параметр	ПЛК210	МВ210-101	МК210-301
Режим работы	Master	Slave	Slave
IP адрес	10.2.7.77	10.2.11.68	10.2.4.5
Маска подсети	255.255.0.0		
IP адрес шлюза	10.2.1.1		
Порт	502		
Slave ID	-	1	1

Таблица 6.6 – Регистры модулей в примере

Модуль	Номер регистра DEC	Тип в устройстве	Функция Modbus	Описание
МВ210-101	4000…4002	FLOAT 32 и UINT 16	0x03	Значение входа 1 и время измерения входа 1
	4003…4005			Значение входа 2 и время измерения входа 2

Продолжение таблицы 6.6

Модуль	Номер регистра DEC	Тип в устройстве	Функция Modbus	Описание
	4005…4008			Значение входа 3 и время измерения входа 3
МК210-301	51	UINT 8	0x03	Битовая маска входов DI1…DI6
	272	UINT 16	0x06	Режим работы выхода DO1: 0 – переключение лог. сигнала; 1 – ШИМ
	308			Период ШИМ DO1: 1000…60000 (мс)
	340			Коэффициент заполнения ШИМ DO1: 0…1000 (0.1 %)
	470	UINT 8		Битовая маска установки состояния выходов DO1…DO8

Для настройки обмена следует:

1. Настроить модули **Mx210** с помощью программы **ОВЕН Конфигуратор** в соответствии с таблицей 6.5 (см. руководство **Mx210. Примеры настройки обмена**). Подключить модули к контроллеру в соответствии с рисунком 6.5.
2. Создать новый проект **Полигон** (в примере с именем *MB_TCP_master_Mx210*). Добавить в проект библиотеку ***paModbus***.
3. Добавить в место работы ***Фон*** программу с именем *Modbus_TCP_Client*.
4. Внутри программы добавить две ***Страницы***, в свойстве ***Комментарии*** которых указать соответственно *MB210-101* и *MK210-301*.

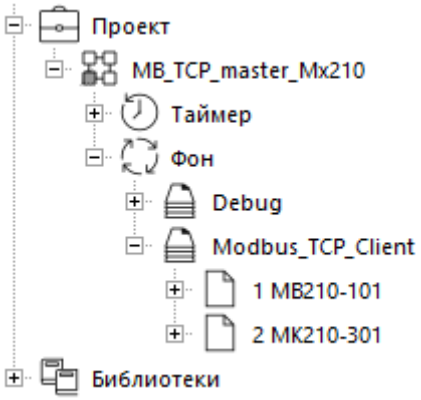


Рисунок 6.46 – Дерево проекта

5. Внутри страницы *MB210-101* создать блок TcpIpClA из библиотеки ***paCore***. На входах блока задать значения в соответствии с таблицей 6.5.

   В примере локальный IP-адрес контроллера взят по SQL-запросу – обращение к соответствующему свойству модуля *MB_TCP_master_Mx210*.

   Запрос IP-адреса:

   ```
   "<sql>SELECT value FROM blocks_prop WHERE indx=:module AND type="prop_ip"</sql>"
   ```

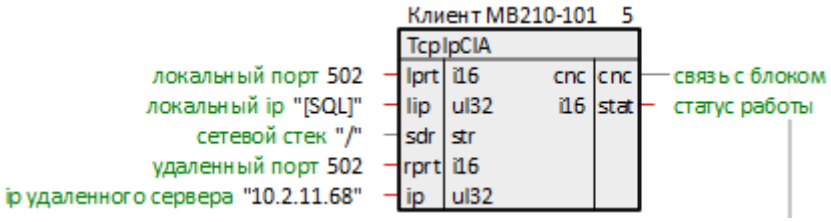


Рисунок 6.47 – Настройка блока TCP/IP-клиента

6. Далее создать блок Modbus TCP Master. Соединить вход **cnc** с соответствующим выходом блока ***TcpIpCIA***.

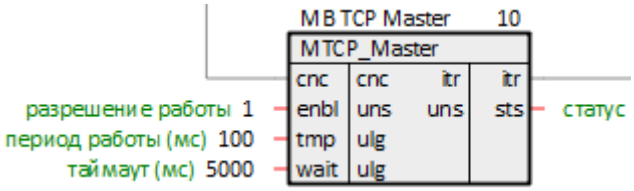


Рисунок 6.48 – Настройка блока Modbus TCP Master

7. Затем создать блок чтения результатов измерения с аналоговых входов OwenFltIn. Добавить к блоку два выхода. В комментариях к выходам **rslt** прописать, к каким входам MB210-101 они относятся. Соединить вход блока **itr** с соответствующим выходом блока ***Modbus TCP Master***.

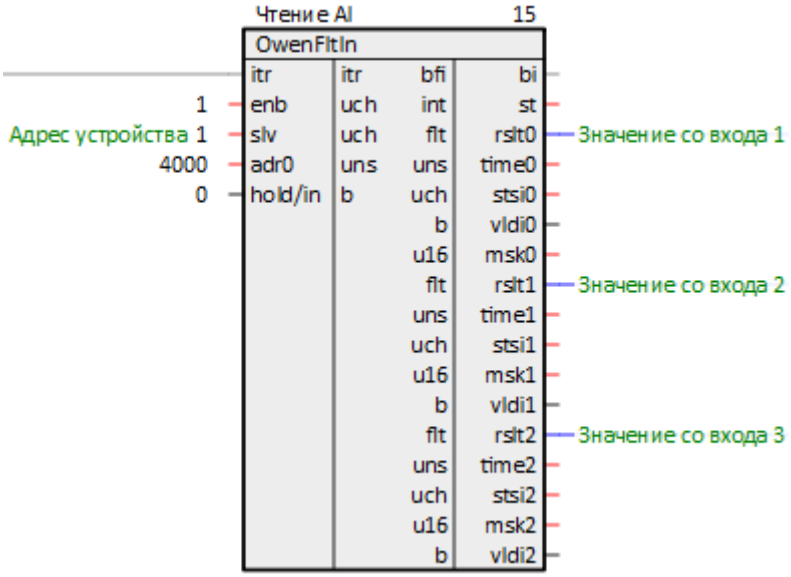


Рисунок 6.49 – Настройка блока OwenFltIn

8. Поставить на странице порядки ***По потоку данных***.
9. Повторить п. 5…6 на странице *МК210-301*.

Рисунок 6.50 – Настройка блоков TCP/IP-клиента и Modbus TCP Master

10. Создать блок для чтения значений с дискретных входов ***ModbusRegIn***. Задать на входе **slv** значение 1, на входе **adr0** – 51 (см. таблицу 6.6). Соединить вход блока **itr** с соответствующим выходом блока ***Modbus TCP Master***.

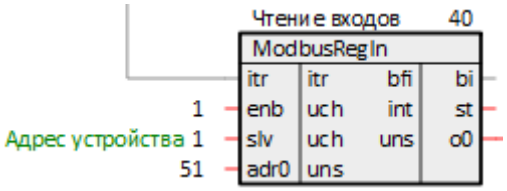


Рисунок 6.51 – Настройка блока ModbusRegIn

11. Создать блок выбора 8 битов из регистра ***FromReg8*** из библиотеки ***paCore*** и соединить выход блока ***ModbusRegIn*** **o0** с входом **reg** блока ***FromReg8***. Добавить к первому выходу блока ***FromReg8*** комментарий – *Вход 1*.

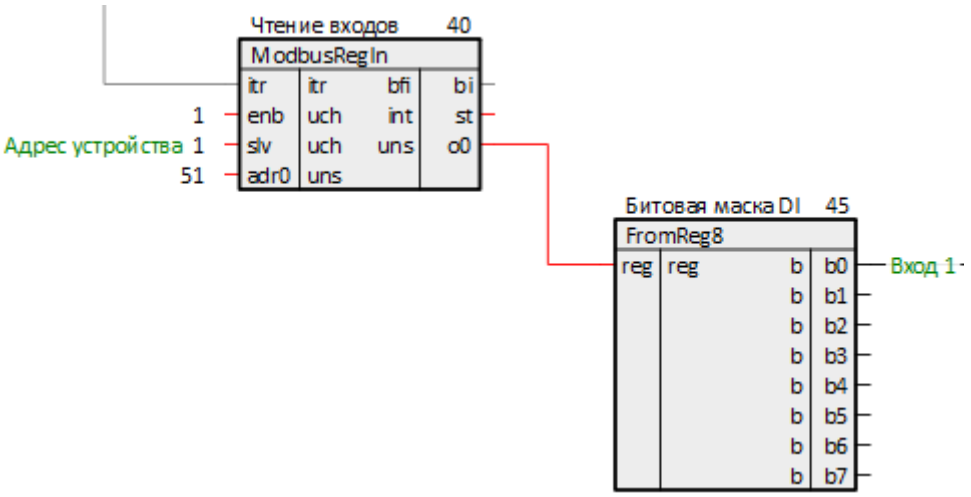


Рисунок 6.52 – Настройка блока FromReg8

12. Аналогично п. 10 создать блоки записи ***ModbusRegOut*** регистров **470**, **272**, **308**, **340** (см. таблицу 6.6). В качестве начальных значений периода и коэффициента заполнения ШИМ указать соответственно, 2000 и 500.

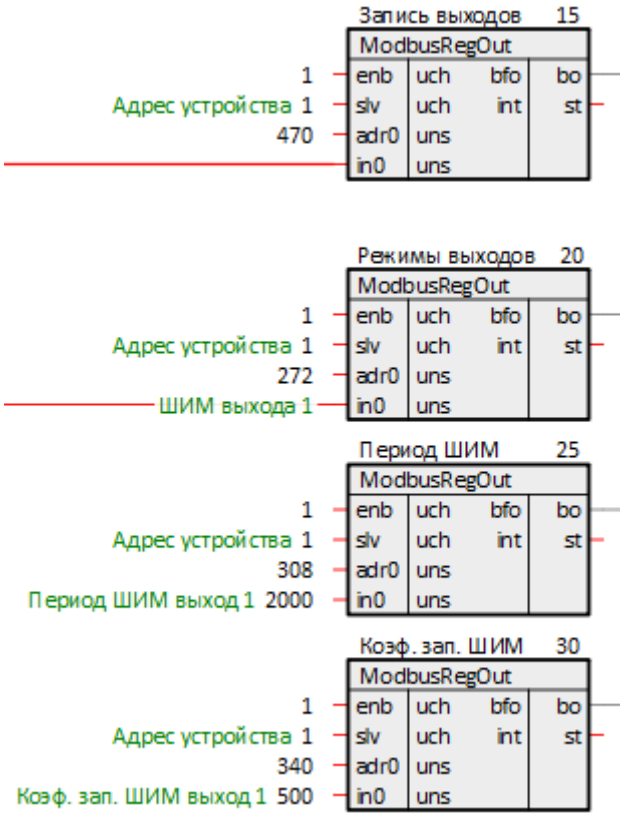


Рисунок 6.53 – Настройка блоков ModbusRegOut

13. Создать блок объединения 8 битов в регистр ***ToReg8*** из библиотеки ***paCore*** аналогично п. 11.

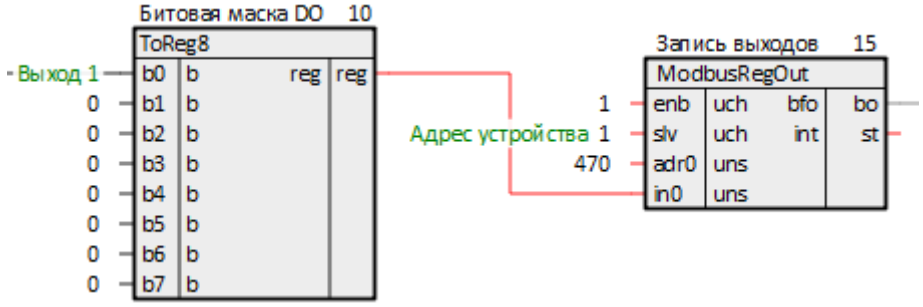


Рисунок 6.54 – Настройка блока ToReg8

14. Поставить на странице порядки ***По потоку данных***.

Итоговый вид страниц *МВ210-101* и *МК210-301* показан на рисунках ниже.

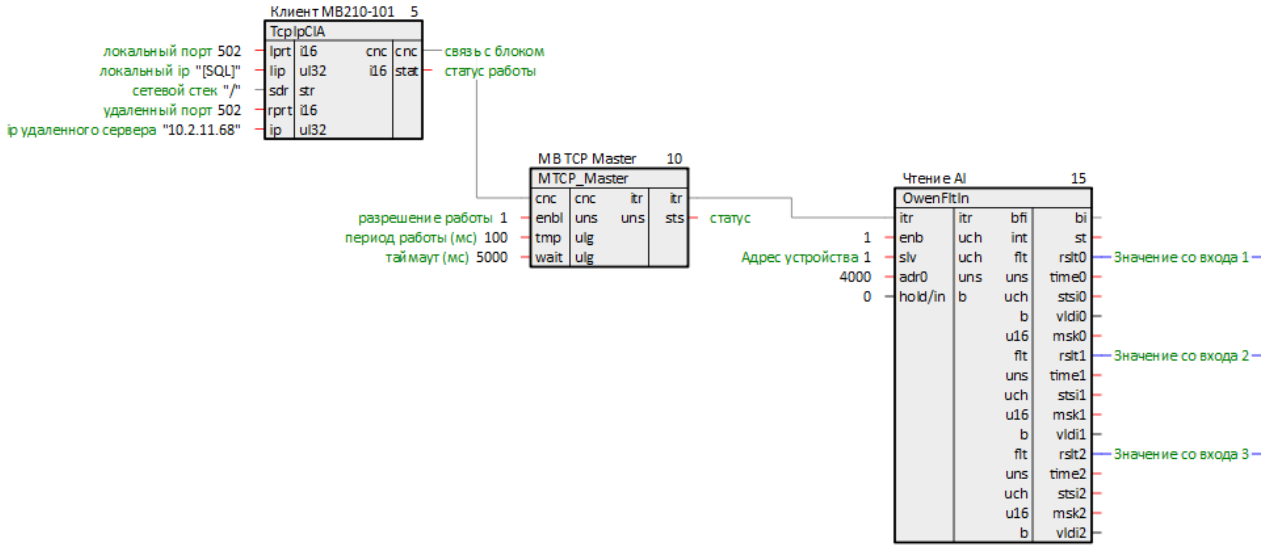


Рисунок 6.55 – Вид страницы MB210-101

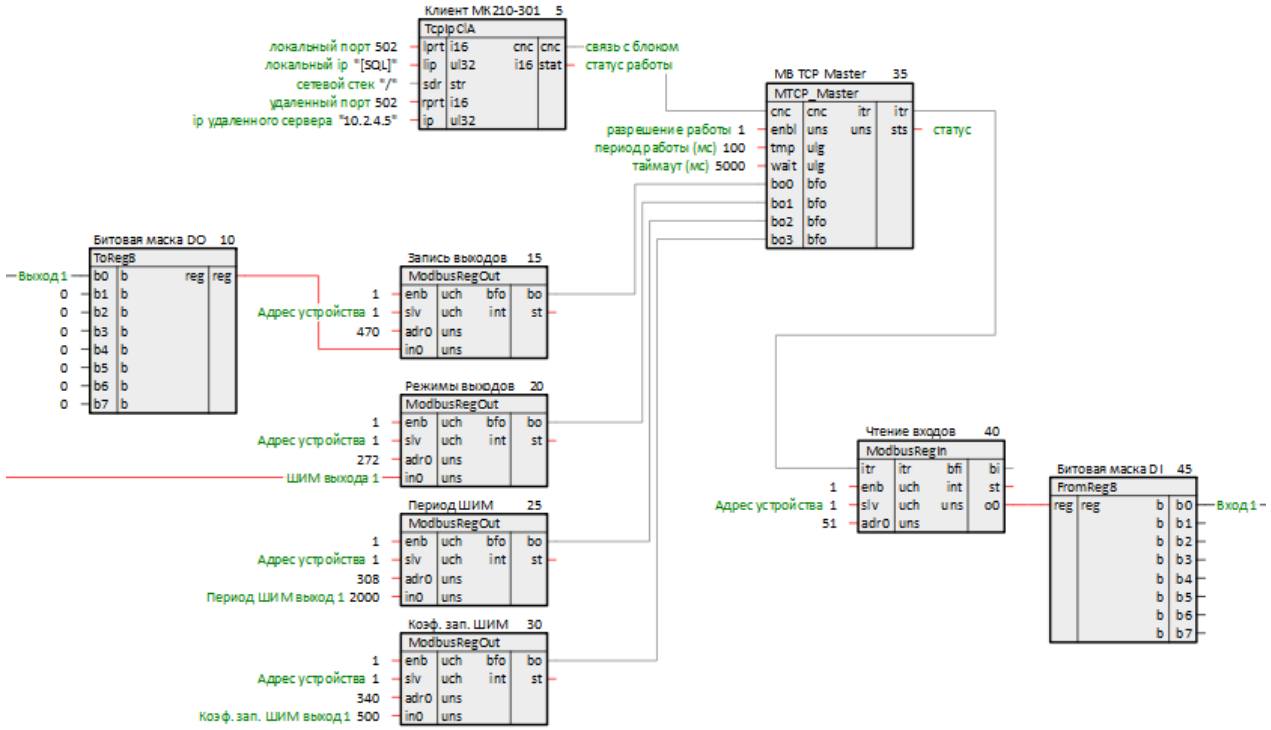


Рисунок 6.56 – Вид страницы MK210-101

Перейти к обработке значений входов/выходов.

15. Создать в месте работы ***Таймер*** программу с названием *Mx210*.

16. Внутри программы создать страницы с комментариями *Значения с модулей* и *Обработка значения температуры*.

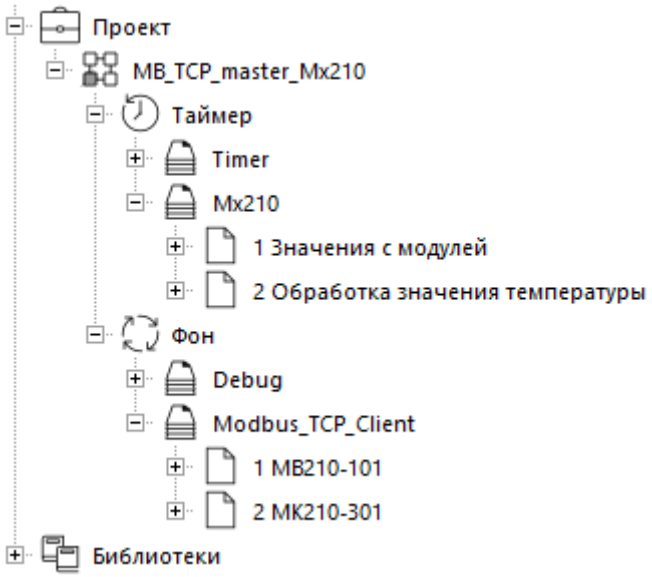


Рисунок 6.57 – Дерево программы

17. На странице *Значения с модулей* создать блоки ***TransBit*** и ***TransFlt*** из библиотеки ***paCore*** с тремя входами.

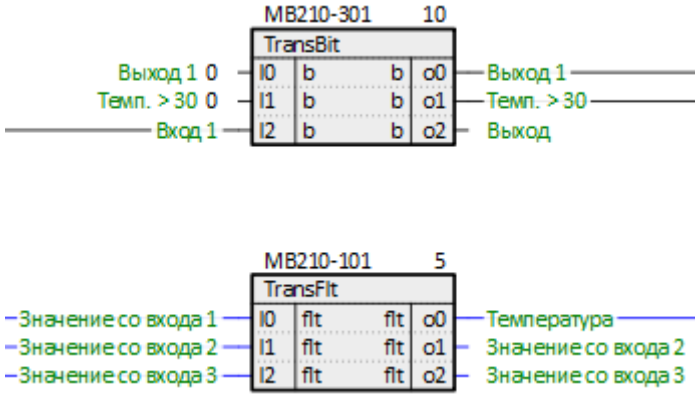


Рисунок 6.58 – Блоки TransBit и TransFlt

18. Соединить входы блоков со страницы *Значения с модулей* с выходами блоков со страниц *MB210-101* и *MK210-301*, как показано на рисунках ниже.

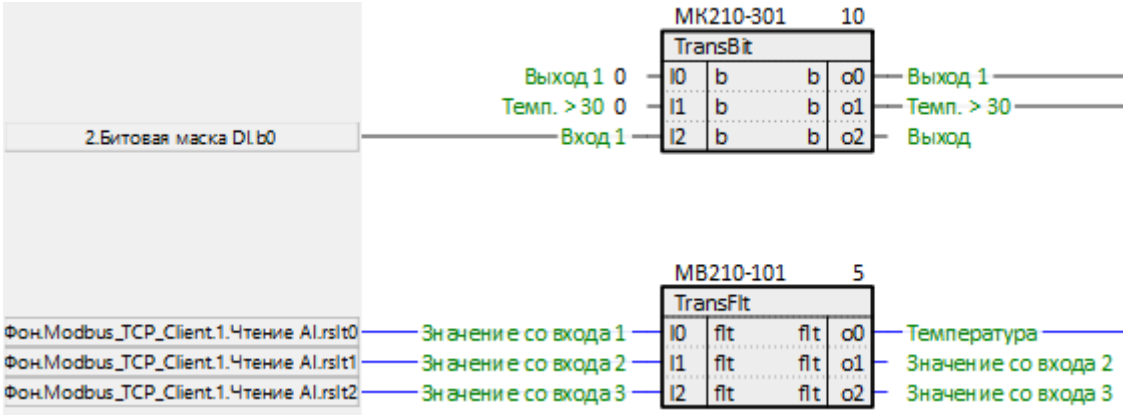


Рисунок 6.59 – Соединение блоков (Значения с модулей)

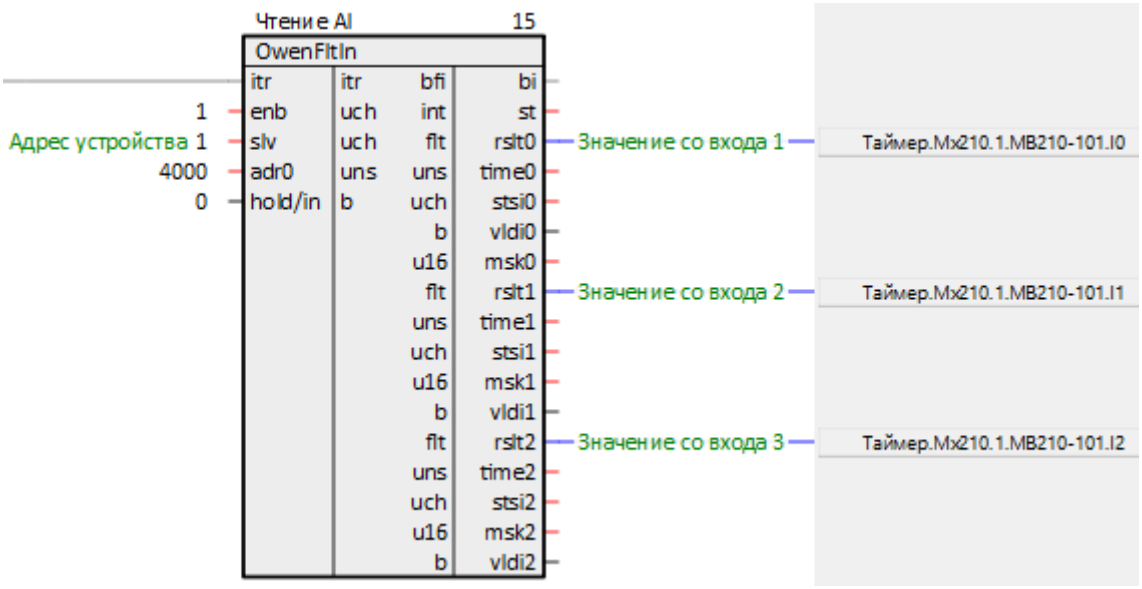


Рисунок 6.60 – Соединение блоков (MB210-101)

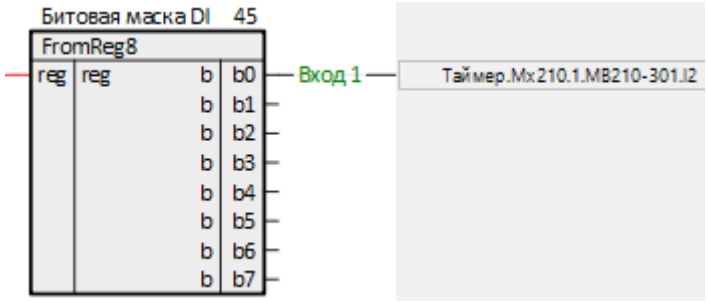


Рисунок 6.61 – Соединение блоков (МК210-301)

19. Соединить выходы блоков со страницы *Значения с модулей* с выходами блоков со страницы *MB210-101*, как показано на рисунках ниже.

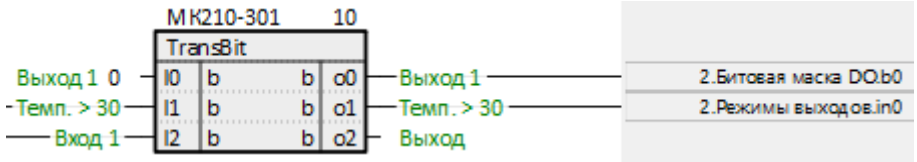


Рисунок 6.62 – Соединение блоков (Значения с модулей)

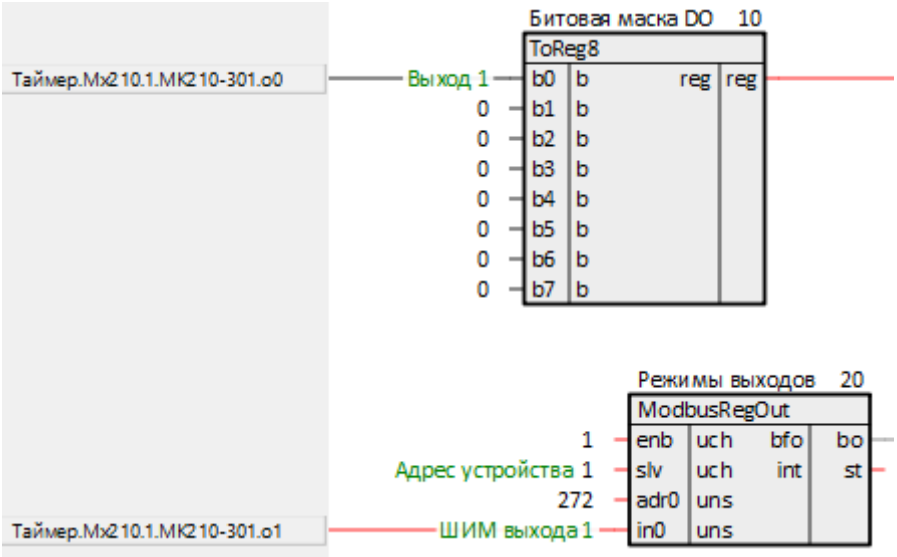


Рисунок 6.63 – Соединение блоков (МК210-301)

20. На странице *Обработка значения температуры* создать блоки ***Cmpr*** из библиотеки ***paCore***.

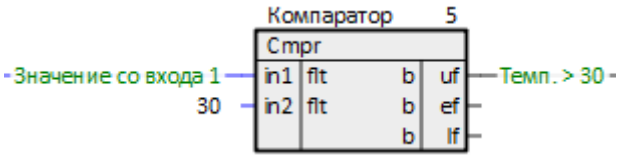


Рисунок 6.64 – Блок Cmpr

21. Соединить вход и выход блока ***Cmpr*** с выходом и входом блоков со страницы *Значения с модулей*, как показано на рисунках ниже.

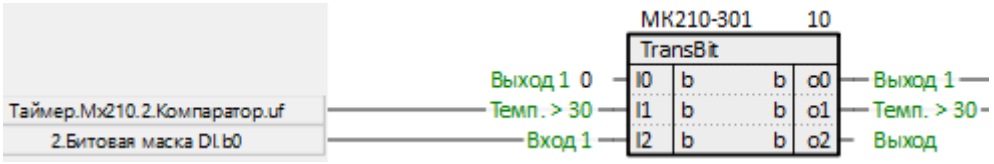


Рисунок 6.65 – Соединение блоков (Значения с модуля МК210–301) с блоком Cmpr

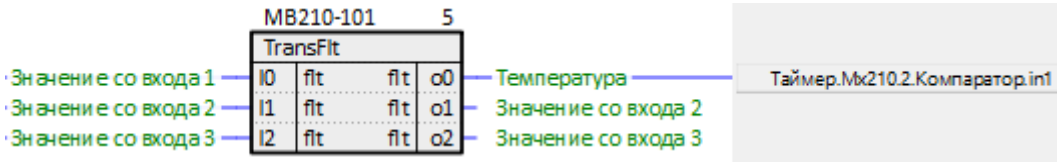


Рисунок 6.66 – Соединение блоков (Значения с модуля МВ210–101) с блоком Cmpr

Для наладки работы собранной системы в примере используется окно представления ***График***.

22. Создать в модуле ***Раздел*** с именем *График*.

23. Добавить в раздел ***График*** выходы блоков ***TransBit*** и ***TransFlt*** – *Выход* и *Температура*. На странице *Значения с модулей* данные выходы должны подсветиться желтым.

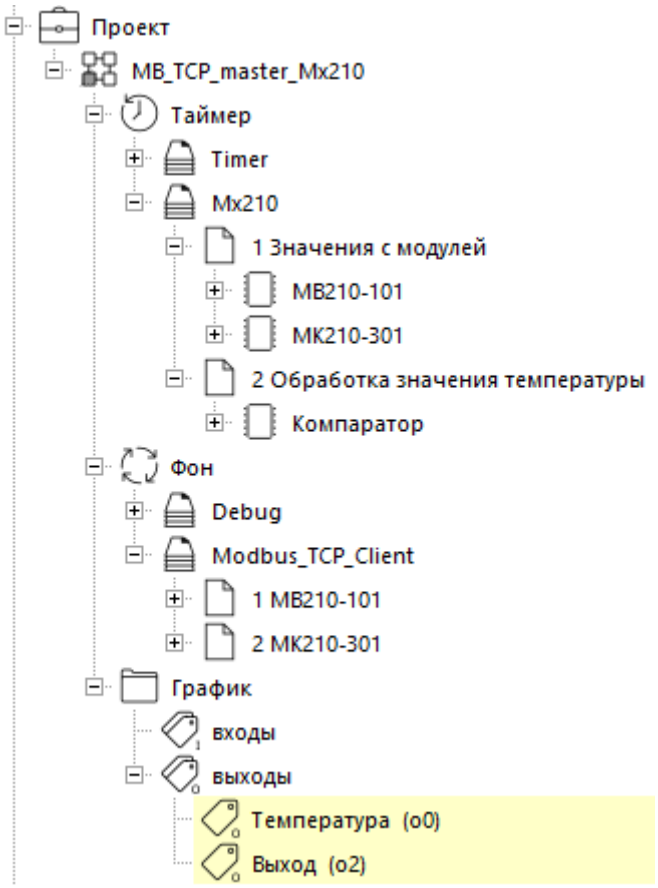


Рисунок 6.67 – Дерево проекта

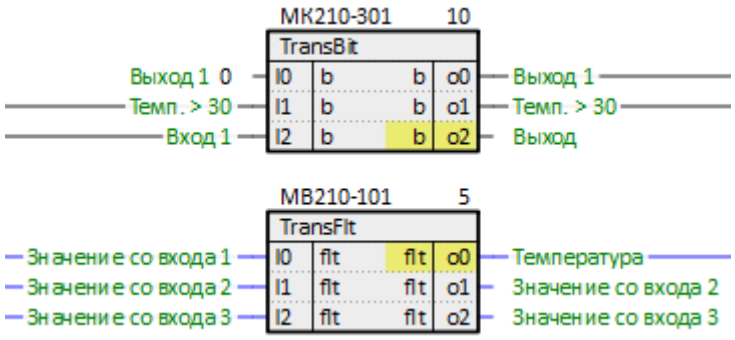


Рисунок 6.68 – Блоки TransBit и TransFlt с подсвеченными выходами

24. Открыть окно представления ***График*** и перетащить созданный раздел в верхнее поле окна.

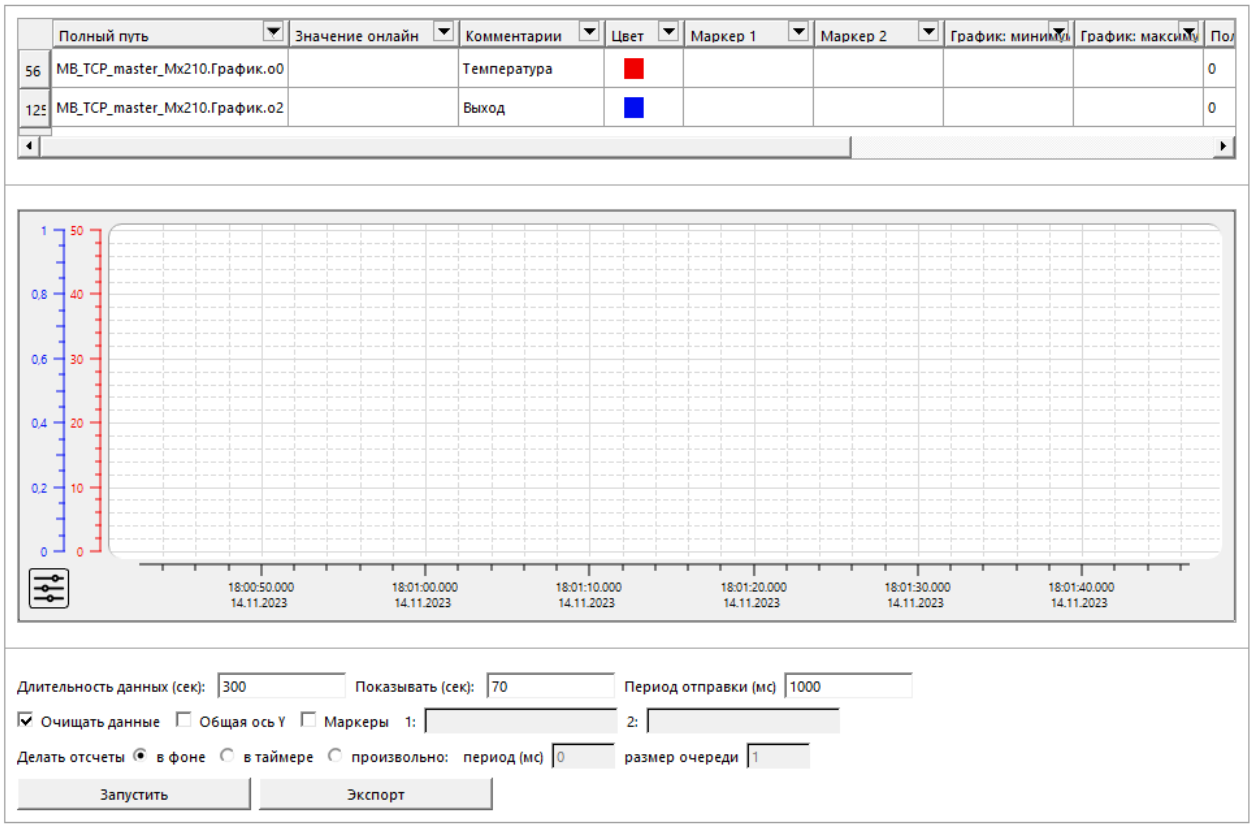


Рисунок 6.69 – Окно представления график

25. Запустить проект на контроллере, запустить отладчик и открыть график. Пронаблюдать корректную работу системы.

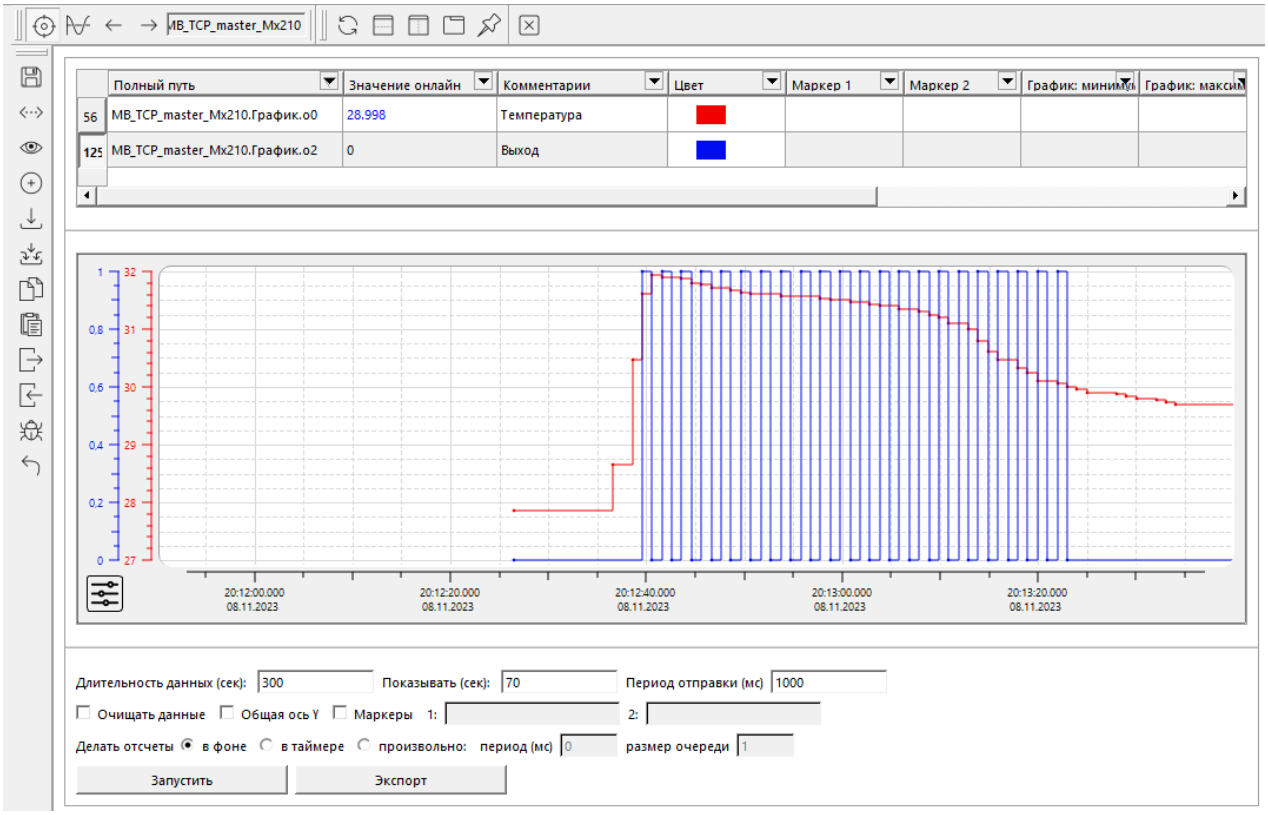


Рисунок 6.70 – Работа программы

6.4 ПЛК210 (Modbus TCP Slave) и Owen OPC Server

В качестве примера будет рассмотрена настройка обмена с Owen OPC Server, который будет использоваться в режиме **Modbus TCP Master**.

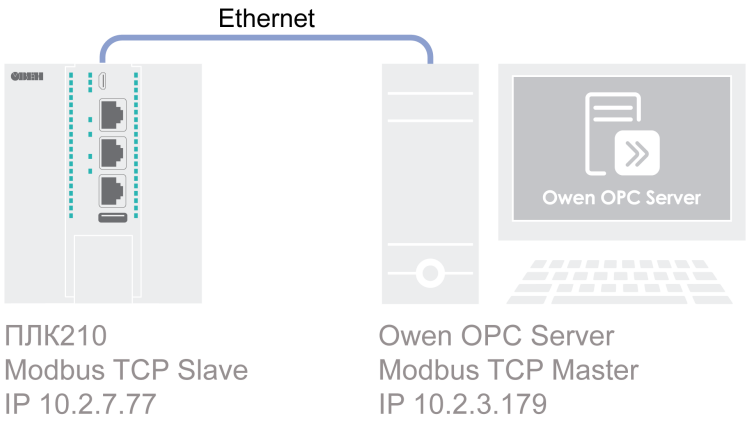


Рисунок 6.71 – Структурная схема примера

Пример создан в среде **Полигон** и подразумевает запуск на **ПЛК210** с прошивкой **3.x**.

Если требуется запустить проект на другом устройстве, следует изменить свойства ***ОС*** и ***Тип процессорной платы*** в окне ***Свойства*** модуля *MB_TCP_slave_Owen_OPC_server* на необходимые.

Пример доступен для скачивания по ссылке. Пароль для доступа к отладчику – **1**.

Таблица 6.7 – Сетевые параметры устройств в примере

Параметр	ПЛК210	Owen OPC Server
Режим работы	Slave	Master
IP адрес	10.2.7.77	10.2.3.179
Порт	502	
Slave ID	1	-

Таблица 6.8 – Регистры/флаги ПЛК в примере

Адрес регистра/флага	Тип в устройстве	Область памяти
0	WORD	Holding Registers
0	BOOL	Coils
1, 2	REAL	Holding Registers

Для настройки обмена следует:

1. Подключить контроллер и ПК к общей локальной сети (сетевые настройки ПЛК и ПК в примере см. таблицу 6.7).
2. Создать новый проект **Полигон** (в примере с именем *MB_TCP_slave_Owen_OPC_server*). Добавить в проект библиотеку ***paModbus***.
3. Добавить в место работы ***Фон*** программу с именем *Modbus_TCP_Server*.
4. Внутри программы добавить две ***Страницы***, в свойстве ***Комментарии*** которых указать соответственно *Modbus TCP Server* и *Регистры Modbus*.

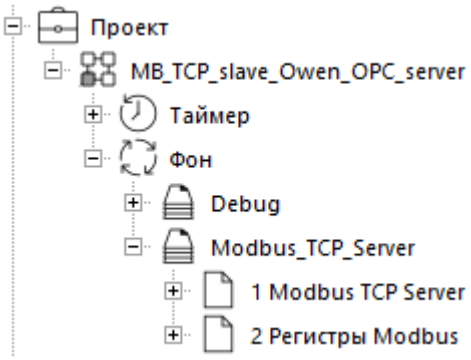


Рисунок 6.72 – Дерево проекта

5. Внутри страницы *Modbus TCP Server* создать блок TcpIpSrA из библиотеки ***paCore***. На входах блока задать значения в соответствии с таблицей 6.7.

   В примере локальный IP-адрес контроллера взят по SQL-запросу – обращение к соответствующему свойству модуля *MB_TCP_slave_Owen_OPC_server*.

   Запрос IP-адреса:

   ```
   "<sql>SELECT value FROM blocks_prop WHERE indx=:module AND type="prop_ip"</sql>"
   ```

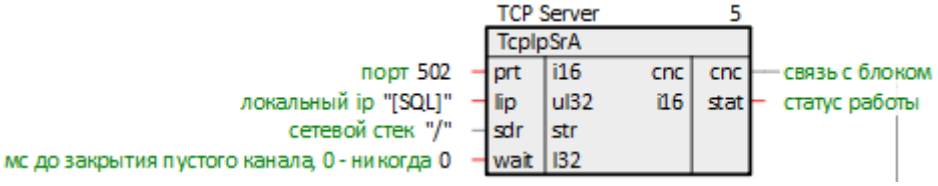


Рисунок 6.73 – Настройка блока TCP/IP-сервера

6. Далее создать блок Modbus TCP Slave. Соединить вход **cnc** с соответствующим выходом блока ***TcpIpSrA***.

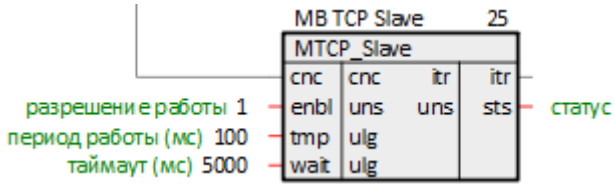


Рисунок 6.74 – Настройка блока Modbus TCP Slave

7. Затем создать блоки записи регистров в ПЛК (в соответствии с таблицей 6.8). Соединить входы блоков **itr** с сооветствующим выходом блока ***Modbus TCP Slave***.

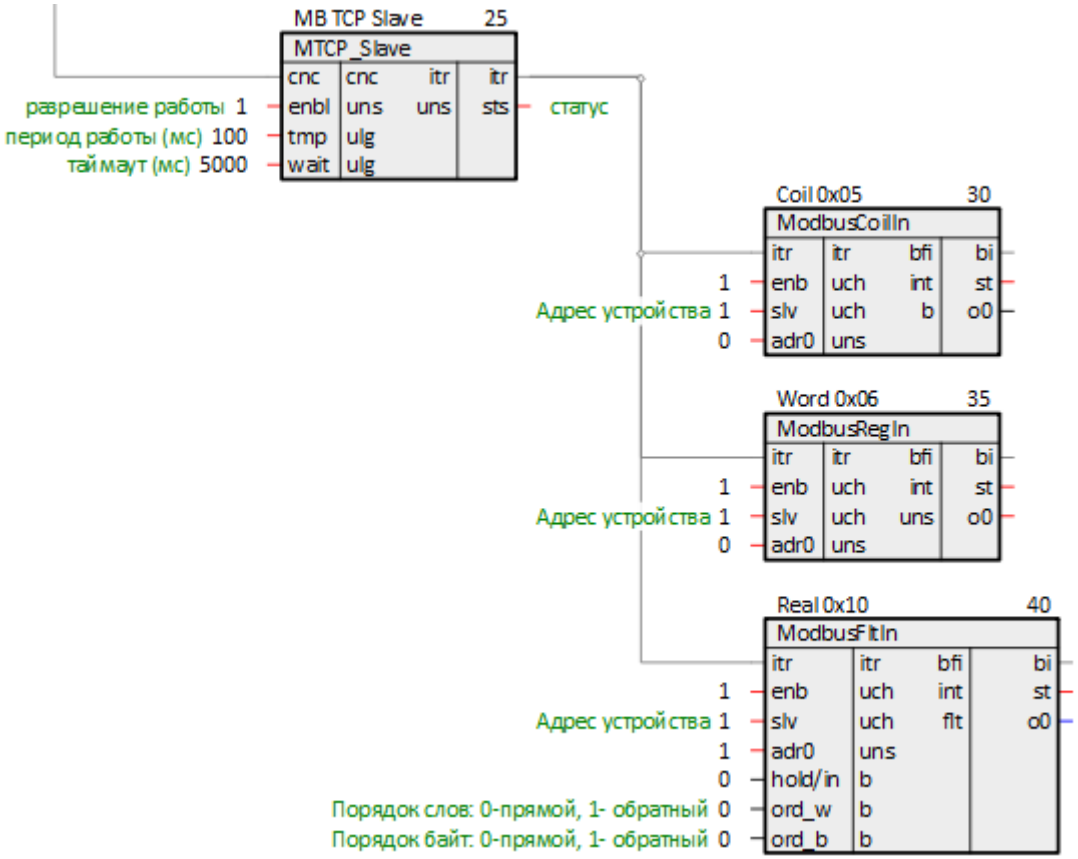


Рисунок 6.75 – Настройка блоков записи

8. Создать блоки чтения регистров из ПЛК (в соответствии с таблицей 6.8). Создать три входа **bo** у блока ***Modbus TCP Slave***. Соединить их с соответствующими выходами блоков чтения.

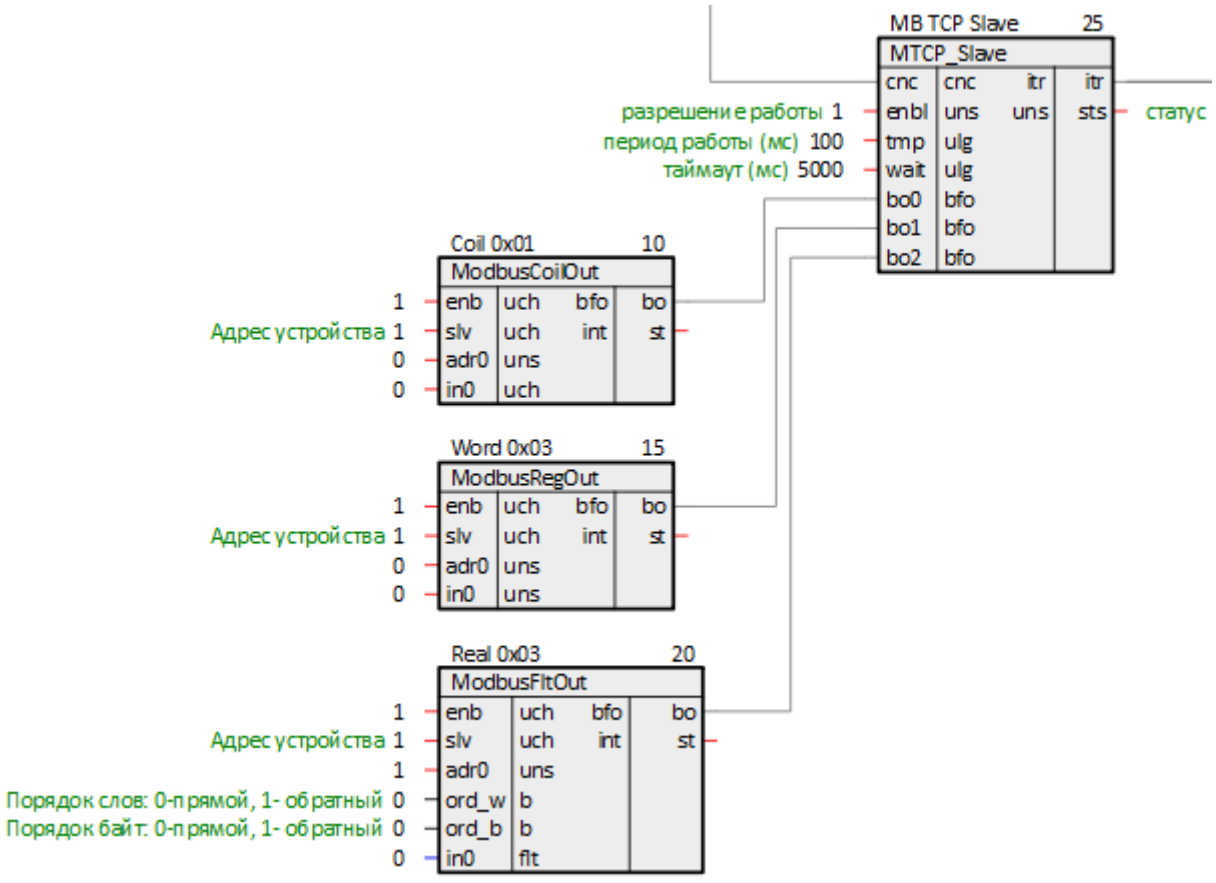


Рисунок 6.76 – Настройка блоков чтения

9. Для того, чтобы одновременно читать и записывать одни и те же значения мастером сети, следует соединить выходы блоков записи **o** со входами блоков чтения **in**.

   Для создания скрытой связи следует в свойствах выхода **o** добавить свойства ***Полный алиас*** и ***Глобальная константа***. В свойстве ***Полный алиас*** задать имя новой константы. Эти действия следует повторить для всех блоков записи на странице.

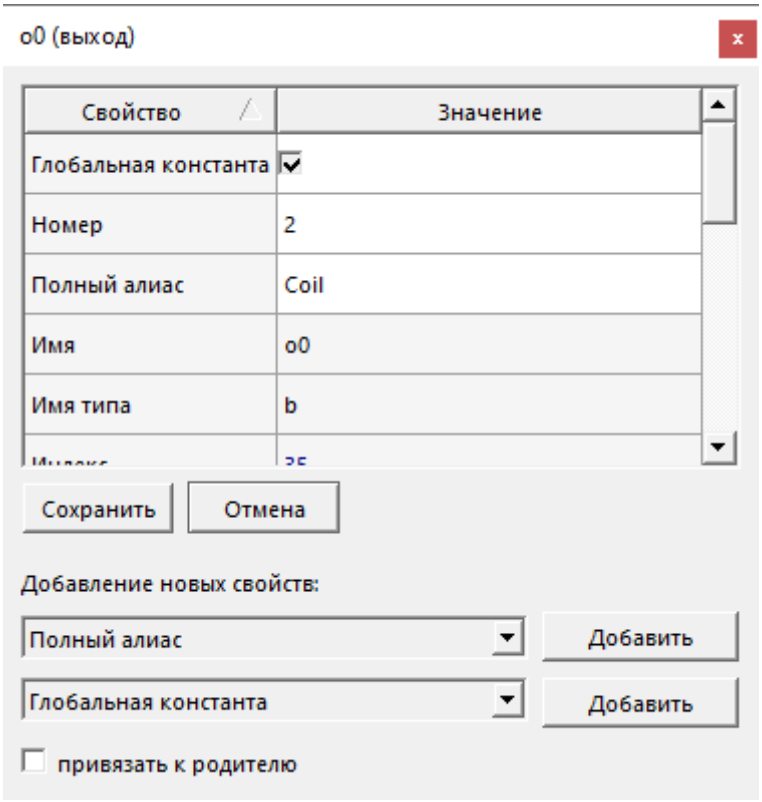


Рисунок 6.77 – Создание скрытой связи

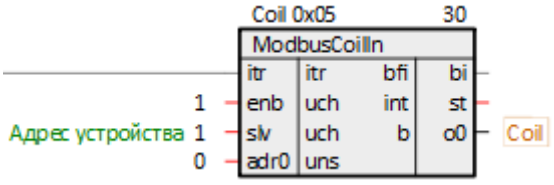


Рисунок 6.78 – Блок записи с константным выходом

10. У соответствующих входов блоков чтения **in** правой кнопкой мыши задать созданные глобальные константы.

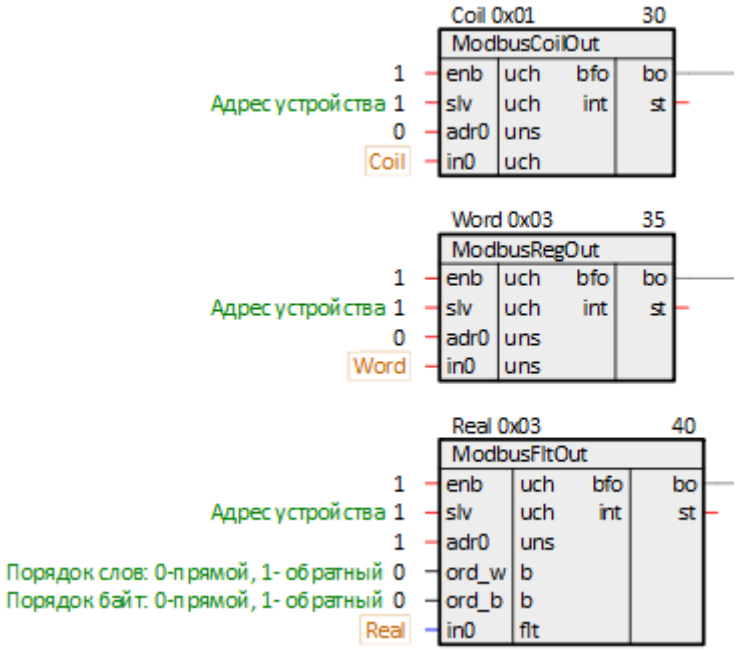


Рисунок 6.79 – Создание скрытой связи

11. Поставить на странице порядки ***По потоку данных***.

Итоговый вид страницы *Modbus TCP Server* показан на рисунке ниже.

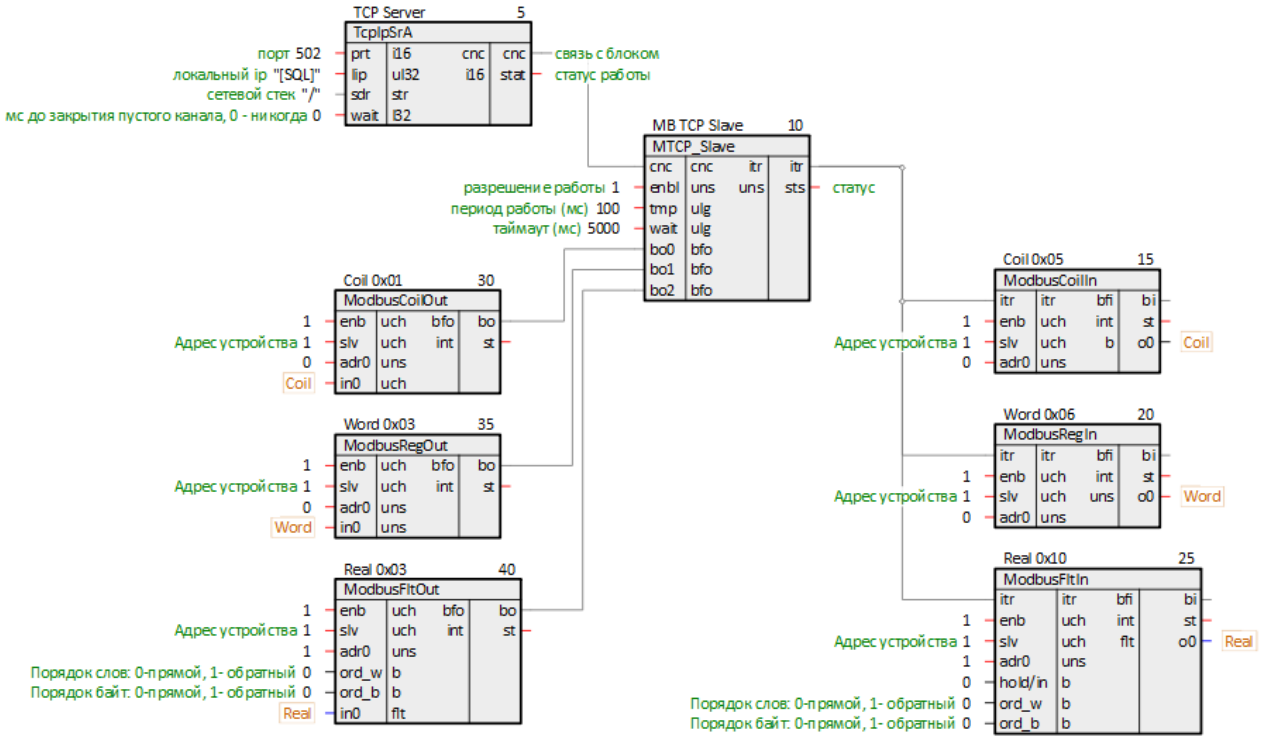


Рисунок 6.80 – Вид страницы Modbus TCP Server

12. Для удобства можно вынести значения созданных регистров ПЛК на отдельную страницу *Регистры Modbus*.

Для этого на странице *Регистры Modbus* следует создать блоки ***TransBit***, ***TransInt*** и ***TransFlt*** из библиотеки ***paCore***.

На входы созданных блоков **I** задать созданные ранее константы. Выходы блоков **o** при необходимости соединить с другими блоками в проекте.

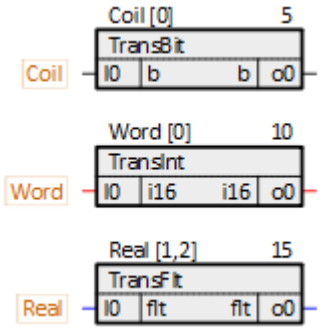


Рисунок 6.81 – Терминальные блоки

13. Установить и запустить Owen OPC Server.
14. Нажать правой кнопкой мыши на компонент **Сервер** и добавить узел.

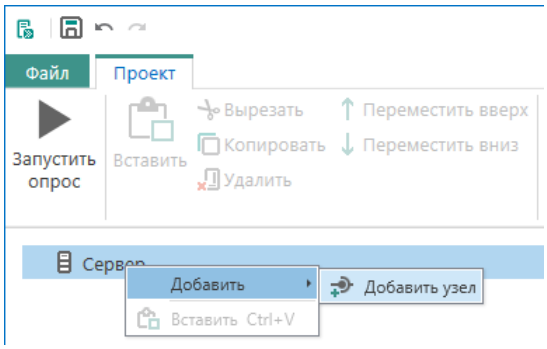


Рисунок 6.82 – Добавление узла

15. В свойствах добавленного узла задать протокол ***Modbus TCP/IP***.

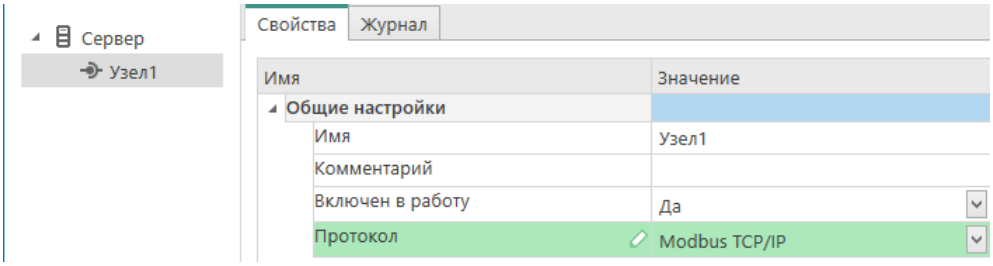


Рисунок 6.83 – Свойства узла

16. Добавить в узел ***Устройство***.

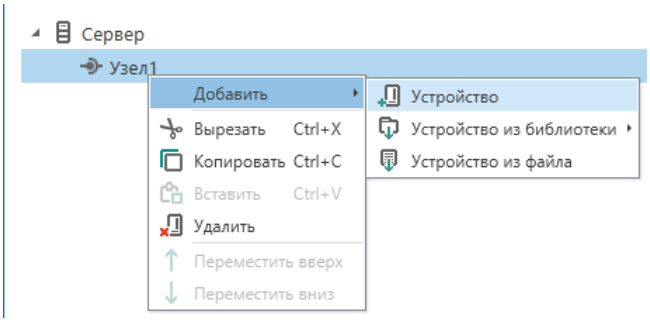


Рисунок 6.84 – Добавление устройства

17. Задать в устройстве свойства в соответствии с таблицей 6.7.

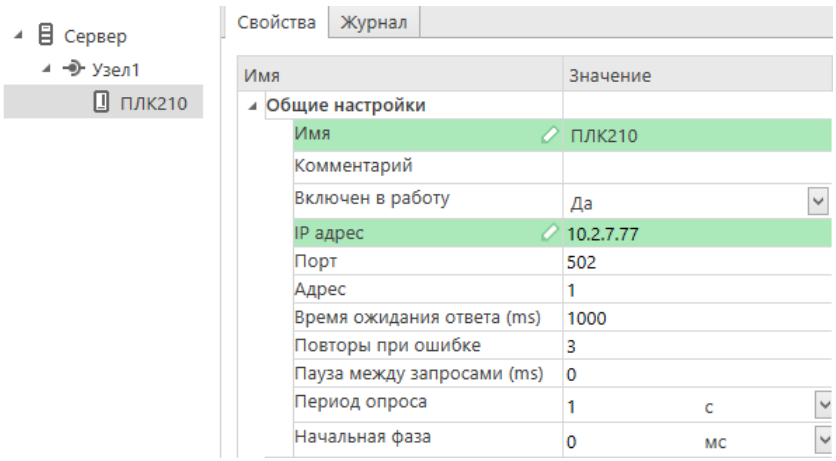


Рисунок 6.85 – Свойства устройства

18. Добавить в устройстве три ***Тега***.

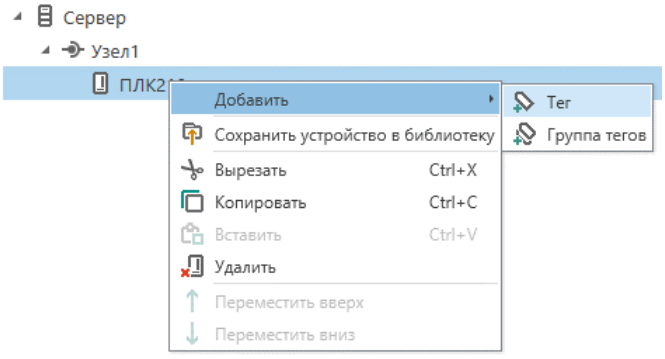


Рисунок 6.86 – Добавление тега

19. Задать созданным тегам свойства в соответствии с таблицей 6.8.

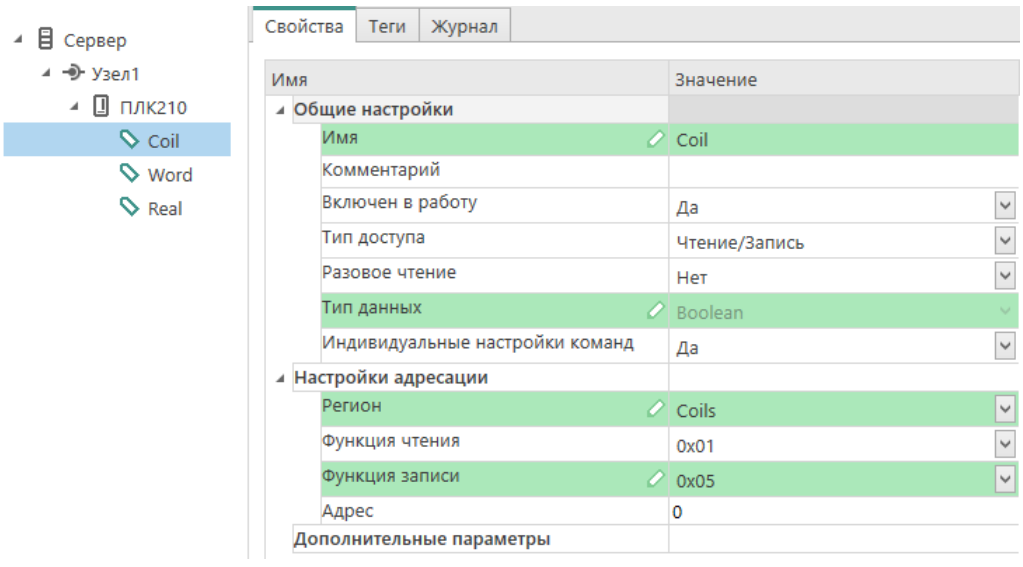


Рисунок 6.87 – Тег Coil

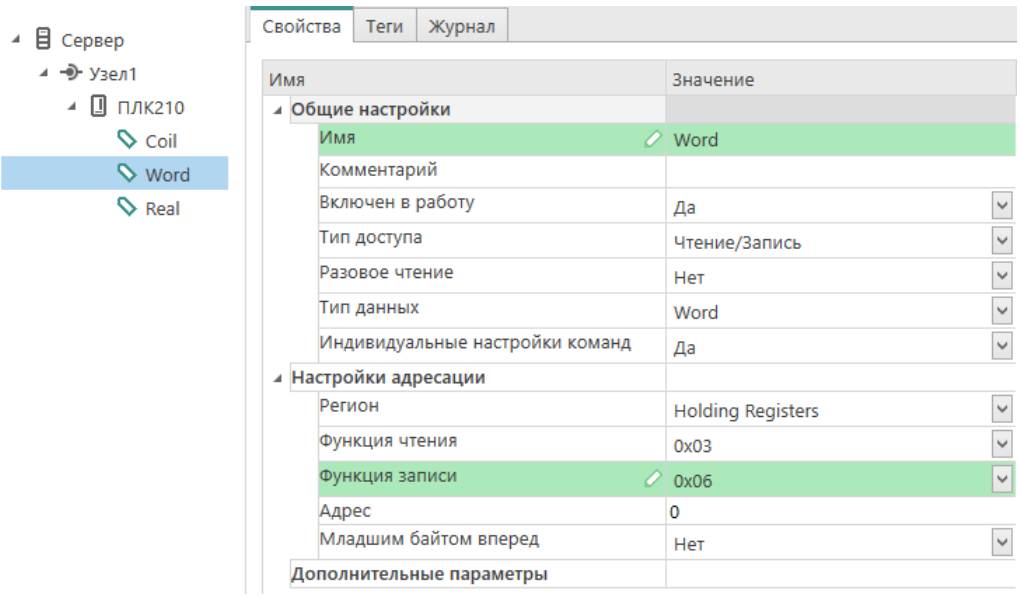


Рисунок 6.88 – Тег Word

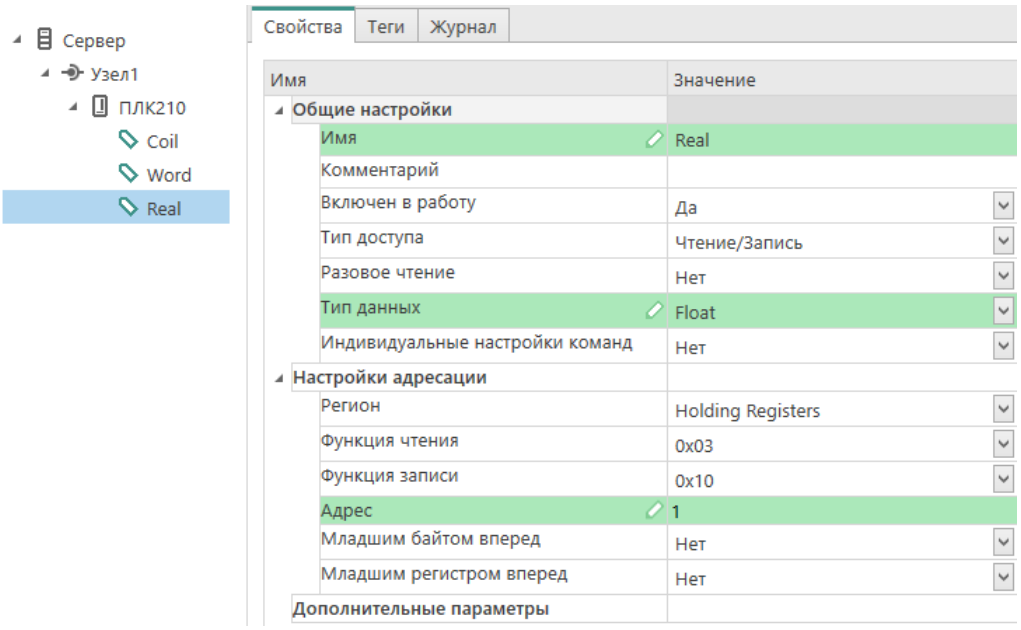


Рисунок 6.89 – Тег Real

20. Загрузить и открыть программу в ПЛК, запустить отладчик. Запустить опрос Owen OPC Server.

21. Корректный обмен данными показан на рисунках ниже.

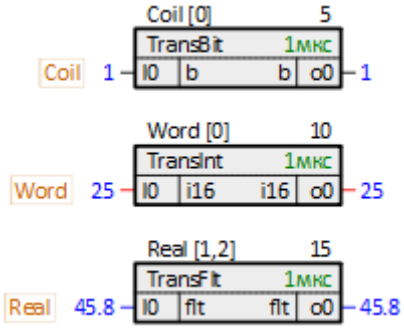


Рисунок 6.90 – Работа программы ПЛК

Теги | Устройства

Имя	Адрес	Значение	Тип данных	Качество	Комментарий
ПЛК210.Coil	Coils [0]	True	Boolean	GOOD	
ПЛК210.Word	Holding Registers [0]	25	Word	GOOD	
ПЛК210.Real	Holding Registers [1]	45,8	Float	GOOD	

Рисунок 6.91 – Работа Owen OPC Server

Приложение А. Коды ошибок Modbus (Modbus Exception Codes)

Таблица А.1 – Коды ошибок Modbus (Modbus Exception Codes)

Код	Имя	Описание
1 (0x01)	ILLEGAL FUNCTION	Код функции в запросе не поддерживается сервером
2 (0x02)	ILLEGAL DATA ADDRESS	Адрес параметра в запросе не поддерживается сервером
3 (0x03)	ILLEGAL DATA VALUE	Недопустимое значение данных для сервера
4 (0x04)	SERVER DEVICE FAILURE	Сбой устройства сервера
5 (0x05)	ACKNOWLEDGE	Сервер принял запрос, но ему требуется много времени для его обработки (предотвращение ошибки тайм-аута)
6 (0x06)	SERVER DEVICE BUSY	Сервер занят обработкой другого запроса. Клиент должен повторить запрос позже, когда сервер освободится
08 (0x08)	MEMORY PARITY ERROR	Произошла ошибка во время использования функции Modbus 20 или 21
10 (0x0A)	GATEWAY PATH UNAVAILABLE	Шлюз перегружен или неправильно настроен – невозможно построить маршрут к серверу
11 (0x0B)	GATEWAY TARGET DEVICE FAILED TO RESPOND	Нет ответа от сервера или он не в сети

Россия, 111024, Москва, 2-я ул. Энтузиастов, д. 5, корп. 5
тел.: +7 (495) 641-11-56, факс: (495) 728-41-45
тех. поддержка 24/7: 8-800-775-63-83, support@owen.ru
отдел продаж: sales@owen.ru
Веб-сайт ООО "ПромАвтоматика-Софт": www.pa.ru
рег.:1-RU-134025-1.1