2016

СПК

Настройка обмена с ПЛК110 [М02]

Руководство для начинающих пользователей

Версия: 1.0
Дата: 02.12.2016

Оглавление

1. Цель и структура документа

Этот документ представляет собой руководство по настройке обмена данными между контроллерами **СПК** и **ПЛК110 [M02]** по протоколу **Modbus**. Основное внимание уделяется практической части вопроса; подробные сведения о работе с **Modbus** приведены в других документах – **СПК. Modbus** и **Руководство пользователя ПЛК**, которые доступных на дисках с ПО из комплекта поставки, а также на сайте компании Овен.

Подразумевается, что пользователь уже имеет базовые навыки работы с соответствующими приборами.

Обратите внимание, что программирование СПК происходит в среде **CODESYS 3.5**, в то время как ПЛК110 [M02] программируется с помощью **CoDeSys 2.3**.

СПК и ПЛК могут работать как в режиме Master, так и в режиме Slave. На вопрос о том, какое из устройств следует делать master'ом, а какое – slave'ом, нет однозначного ответа. Пользователь должен выбрать режим, основываясь на специфике конкретной задачи и используемого оборудования. В качестве примера можно рассмотреть две типовые ситуации:

- Один СПК используется для отображения данных с нескольких ПЛК. В рамках Modbus RTU очевидным является решение: СПК – Master, ПЛК – slave'ы;
- Несколько СПК используются для отображения данных с одного ПЛК. В рамках Modbus RTU очевидным является решение: ПЛК – Master, СПК – slave'ы;

Документ содержит 4 примера:

1. Протокол Modbus RTU. СПК – master, ПЛК – slave;

2. Протокол Modbus RTU. СПК – slave, ПЛК – master;

3. Протокол Modbus TCP. СПК – master, ПЛК – slave;

4. Протокол Modbus TCP. СПК – slave, ПЛК – master.

2. Основные особенности настройки обмена для СПК и ПЛК

Ниже приведены ключевые особенности настройки обмена между СПК и ПЛК, на которые мы будем неоднократно ссылаться в дальнейшем.

2.1. СПК: Соответствие нумерации COM-портов СПК и CODESYS

При настройке интерфейсов RS-232/485 в **CODESYS** необходимо указывать номера портов. Программная нумерация портов отличается от нумерации на корпусе устройств. Соответствие между номерами портов на корпусе СПК и в CODESYS приведено в табл. 2.1:

Табл. 2.1. Соответствие нумерации COM-портов СПК и CODESYS

<table>
<tr><th rowspan="2">Нумерация портов на корпусе прибора</th><th colspan="5">Нумерация портов в CODESYS</th></tr>
<tr><td>СПК105*</td><td colspan="2">СПК107,
СПК107.Д</td><td>СПК110,
СПК110.Д</td><td>СПК2xx.03</td><td>СПК2xx.04</td></tr>
<tr><td>COM1</td><td>2 (RS-485)
3 (RS-232)</td><td colspan="2">2 (RS-232/485)</td><td colspan="2">2 (RS-232)</td></tr>
<tr><td>COM2</td><td>-</td><td colspan="2">3 (RS-232/485)</td><td>3 (RS-232/485)</td><td>3 (RS-232)</td></tr>
<tr><td>COM3</td><td>-</td><td colspan="2">-</td><td colspan="2">4 (RS-232/485)</td></tr>
</table>

* *В СПК105 интерфейсы RS-485 и RS-232 выведены на один порт COM1 и, в отличие от остальных СПК, поддерживается их одновременная работа*

2.2. СПК: использование объединений (UNION)

Стандарт **Modbus** предусматривает только два типа данных, участвующих в обмене – **BOOL** и **WORD**. Достаточно часто возникает потребность передать данные других типов, например, **REAL** и **STRING**. В этом случае на устройстве, которое отравляет данные, необходимо преобразовать их в последовательность **WORD** регистров. Соответственно, на устройстве, получающем данные, должно быть выполнено обратное преобразование. Наиболее простой способ сделать это в **CODESYS 3.5** – использовать тип данных объединение (Union). При настройке ПЛК в **CoDeSys 2.3** это не требуется, т.к. у пользователя уже есть готовые модули для каждого типа данных.

Объединение (UNION) представляет собой пользовательский тип данных, все переменные которого расположены в одной области памяти. Таким образом, переменные различных типов будут представлять различную интерпретацию одних и тех же данных. Для конвертации достаточно записать значение в одну из переменных объединения и считать его из другой.

Рассмотрим конвертацию значения с плавающей точкой, хранящегося в двух **WORD**, в переменную типа **REAL**:

1. Нажмите **ПКМ** на приложение **Application** и добавьте объект **DUT** типа **объединение** с названием **Real_Word**:

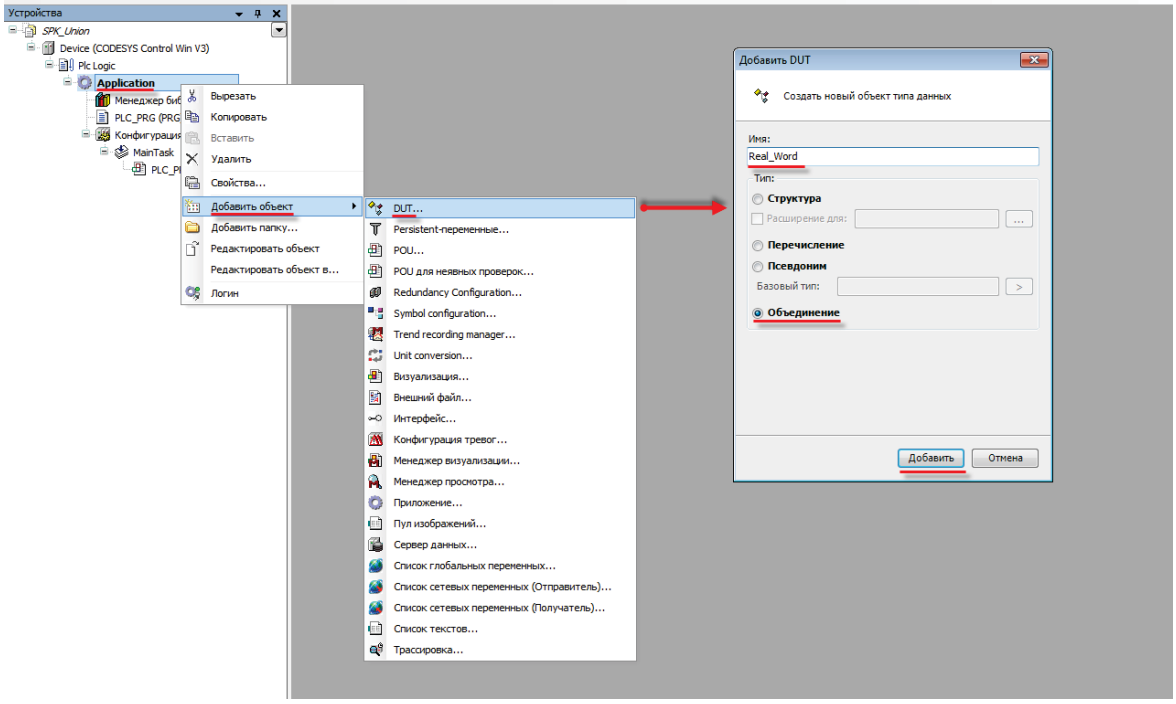


Рис. 2.1. Добавление в проект объединения

2. В объединении объявите переменную **rRealValue** типа **REAL** и массив **awModbusReal** типа **WORD**, содержащий два элемента:

```
Real_Word
1  TYPE Real_Word :
2  UNION
3      rRealValue      :REAL;
4      awModbusReal    :ARRAY [0..1] OF WORD;
5  END_UNION
6  END_TYPE
```

Рис. 2.2. Объявление переменных объединения

3. В программе объявим экземпляр объединения **Real_Word** с названием **_2WORD_TO_REAL**:

```
PLC_PRG
1  PROGRAM PLC_PRG
2  VAR
3      _2WORD_TO_REAL: Real_Word;
4  END_VAR
5
```

Рис. 2.3. Объявление экземпляра объединения в программе

Для использования переменных объединения в нужном месте программы введите имя экземпляра объединения и нажмите точку, после чего выберите из списка нужную переменную:

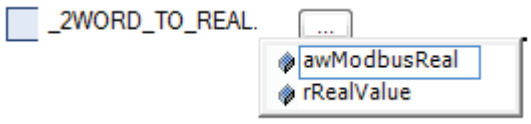


Рис. 2.4. Работа с переменными объединения в программе

4. Переменные массива **awModbusReal** будут привязаны к регистрам при настройке **Modbus**, а переменная **rRealValue** будет использоваться в программе для работы со значением с плавающей точкой.

На рис. 2.5 приведен скриншот значения переменных объединения в процессе работы программы. При записи в элементы массива **awModbusReal** значений в HEX, в переменную **rReaLValue** будет записано соответствующее им значение с плавающей точкой.

PLC_PRG

Device.Application.PLC_PRG

Выражение	Тип	Значение
_2WORD_TO_REAL	REAL_WORD	
rRealValue	REAL	1.1
awModbusReal	ARRAY [0..1] OF WORD	
awModbusReal[0]	WORD	16#CCCD
awModbusReal[1]	WORD	16#3F8C

Рис. 2.5. Отображение значений переменных объединения

2.3. CODESYS: области памяти slave-устройства

При настройке СПК в режиме **Modbus Slave** необходимо обращать внимание на следующие особенности CODESYS 3.5:

1 CODESYS 3.5 в текущих версиях (SP7 Patch4)не поддерживает функции работы с битами для Modbus RTU;

2. **Input registers** (регистры ввода) доступны только для чтения, **holding registers (**регистры временного хранения) – для чтения и записи.

3. Значения **holding** регистров не могут быть изменены из программы СПК.

2.4. ПЛК: выравнивание данных в Modbus Slave

При настройке ПЛК в режиме **Modbus Slave** необходимо выравнивать данные по регистрам (напомним, что регистр занимает 2 байта, т.е. 16 бит). При этом надо учитывать, что адреса регистров размещаемых данных должны быть кратны количеству байт этих данных.

Поясним вышесказанное на примере. Предположим, нам необходимо использовать в slave'е элемент **8 bits**, два элемента **2 byte** и элемент **float**. Размещать их последовательно **нельзя**. Выравненные данные в этом случае будет выглядеть следующим образом:

Элемент	**Комментарий**	**Адреса регистров**	**Адреса бит (в абсолютной адресации)**
8 bits		0	0..7
8 bits	добавлен для выравнивания		8..15
2 byte		1	16…31
2 byte		2	32…47
2 byte	добавлен для выравнивания	3	48…63
float		4	64…79

Более подробная информация о выравнивании содержится в документе **Руководство пользователя ПЛК**, доступном на диске с ПО из комплекта поставки, а также сайте компании Овен.

3. Modbus RTU. СПК – master, ПЛК – slave

3.1. Описание примера

Этот пример посвящен настройке обмена данными между сенсорным панельным контроллером **СПК207** и контроллером **ПЛК110 [M02]** по протоколу **Modbus RTU**. В этом примере СПК выполняет функцию **Master**, а ПЛК – **Slave**.

Основные характеристики используемых устройств приведены в табл. 3.1. Используемые в примере переменные описаны в табл. 3.2.

Табл. 3.1. Характеристики устройств

Устройство	**СПК207**	**ПЛК110 [M02]**
Функция	Master	Slave
Используемый порт (нумерация на корпусе)	RS-485 (COM2)	RS-485 (1)
Настройки обмена	115200, 8 бит, 1 стоп бит, без контроля четности	
Slave ID	-	1
Таргет	3.5.4.20 (023)	PLC110.30-M v2 (версия 3.08)
Среда разработки проекта	CODESYS 3.5 SP7 Patch4	CoDeSys 2.3.9.41
Название файла проекта	ModbusRTUmaster.projectarchive	ModbusRTUslave.pro

Табл. 3.2. Список переменных

СПК207 (Master)			ПЛК110 [M02] (Slave)	
Переменные, в которые считываются значения из Slave	**Переменные, значения которых записываются в Slave**	**Тип данных**	**Переменные ПЛК**	**Адрес регистра/бита**
xVarRead	xVarWrite	BOOL	xVar	0/0
wVarRead	wVarWrite	WORD	wVar	1
rVarRead	rVarWrite	REAL	rVar	2-3
sVarRead	sVarWrite	STRING(6)	sVar	4-6

Проекты примера доступны для скачивания: Example_SpkModbusRtuMaster.zip

3.2. Настройка ПЛК (slave)

1. Создайте новый проект **CoDeSys 2.3** для **ПЛК110** с программой **PLC_PRG** на языке **CFC**.

2. В компоненте **Конфигурация ПЛК** (вкладка **Ресурсы**) пользователь производит настройку регистров Modbus и привязывает к ним переменные.

Нажмем **ПКМ** на название контроллера (в нашем примере - **PLC110_30**) и добавим подэлемент **Modbus (Slave)**:

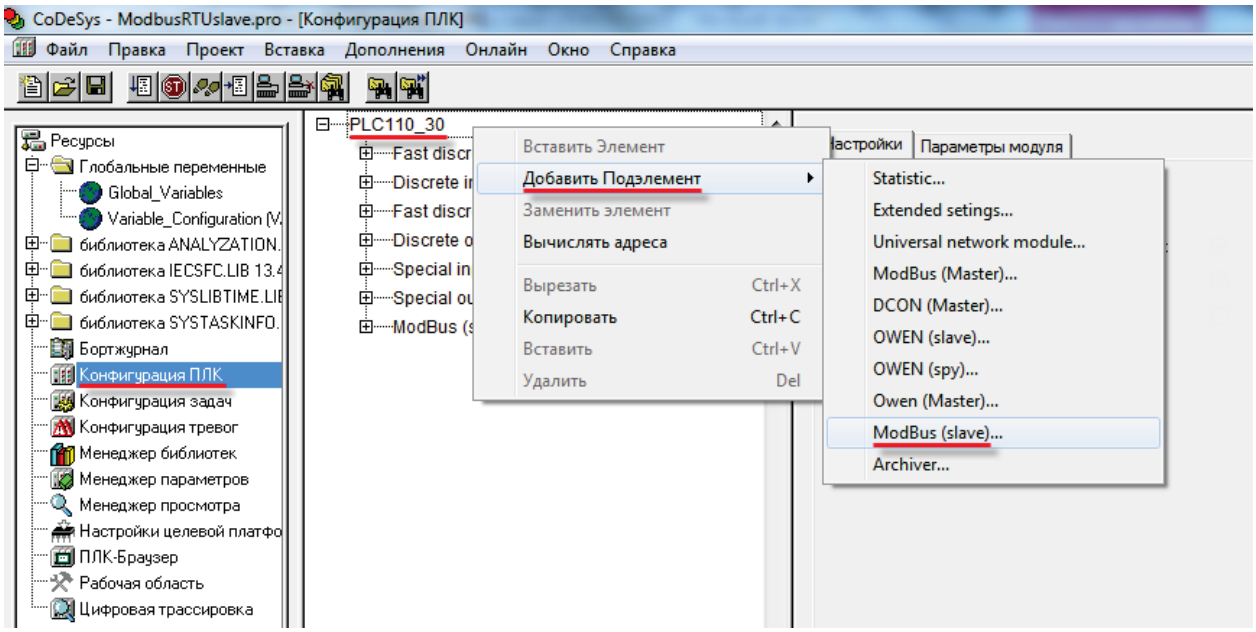


Рис. 3.1. Конфигурация ПЛК. Добавление **Modbus (Slave)**

В его настройках выберем адрес (**Slave ID**), равный **1** (в соответствии с табл. 3.1):

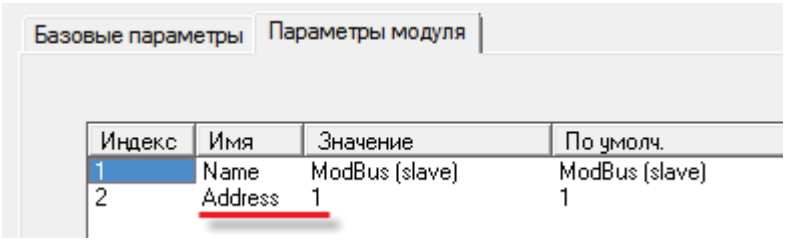


Рис. 3.2. Конфигурация ПЛК. Настройка **Modbus (Slave)**

3. Выберем порт ПЛК, который будет использоваться для связи с СПК. Для этого нажмем **ПКМ** на элемент **Modbus (FIX)** и добавим подэлемент **RS-485-1** (согласно табл. 3.1).

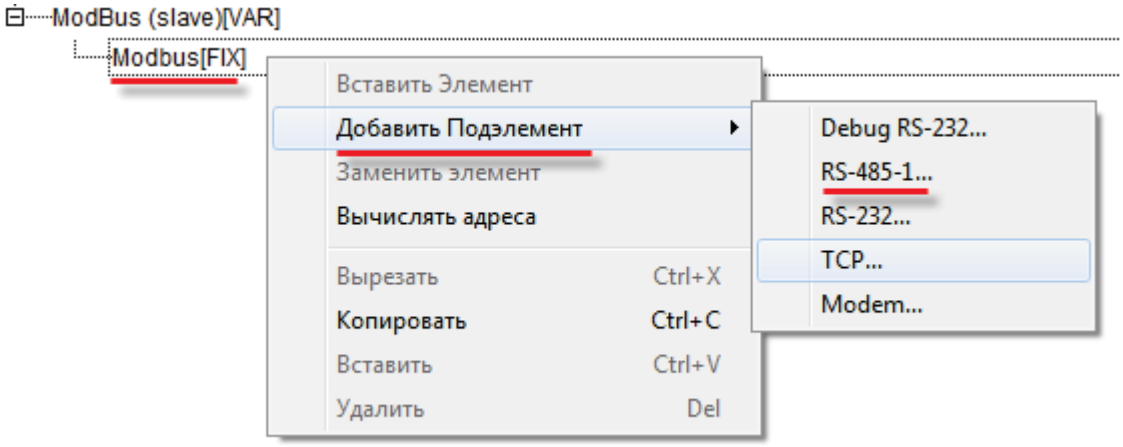


Рис. 3.3. Конфигурация ПЛК. Добавление подэлемента **RS-485-1**

В параметрах подэлемента укажите сетевые настройки в соответствии с табл. 3.1, а также используемый протокол – **RTU**.

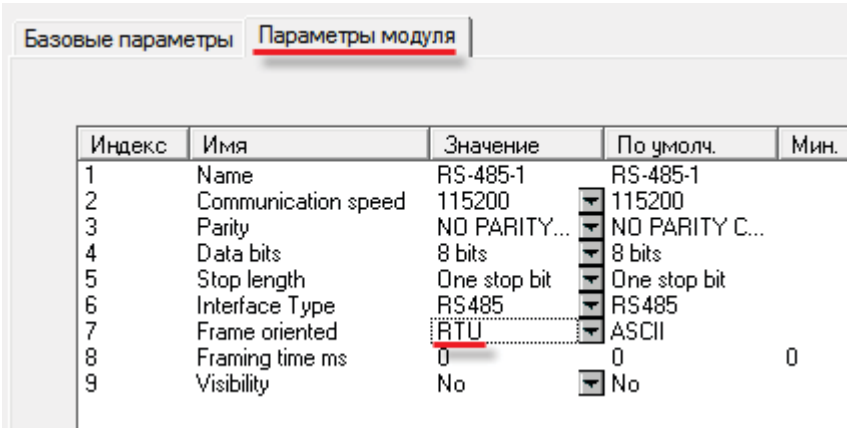


Рис. 3.4. Конфигурация ПЛК. Настройки подэлемента **RS-485-1**

4. Нажмем **ПКМ** на элемент **Modbus (Slave)** и добавим следующие подэлементы:

- **8 bits** (для **BOOL**);
- **8 bits** (для обеспечения выравнивания памяти);
- **2 byte** (для **WORD**);
- **Float** (для **REAL**);
- 3 элемента **2 byte** (для **STRING** из 6 символов).

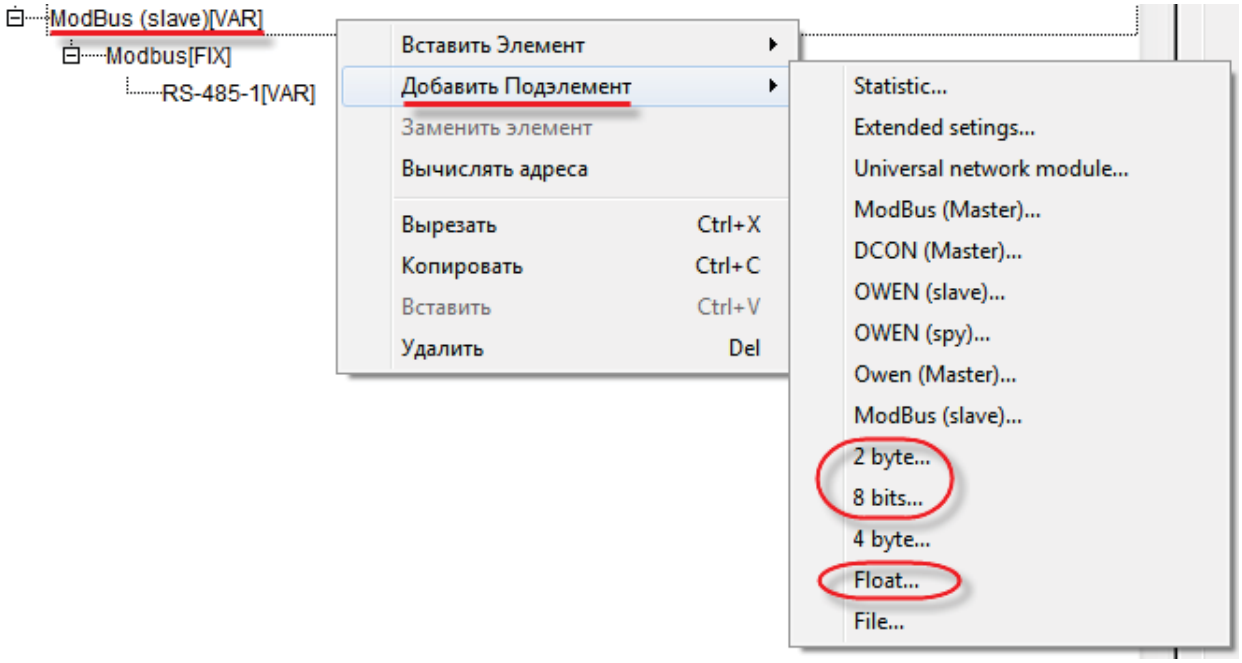


Рис. 3.5. Конфигурация ПЛК. Добавление подэлементов

В результате **Конфигурация ПЛК** будет выглядеть следующим образом (см. рис. 3.6). Объявим переменные (после ввода их имен они автоматически будут добавлены в список глобальных переменных проекта). Для ввода имени переменной два раза нажмите на **AT**. ***Обратите внимание***, что к регистрам 4-6 не привязывается никаких переменных. Соответствующая им переменная **sVar** будет объявлена в программе **PLC_PRG** (в пп. 5).

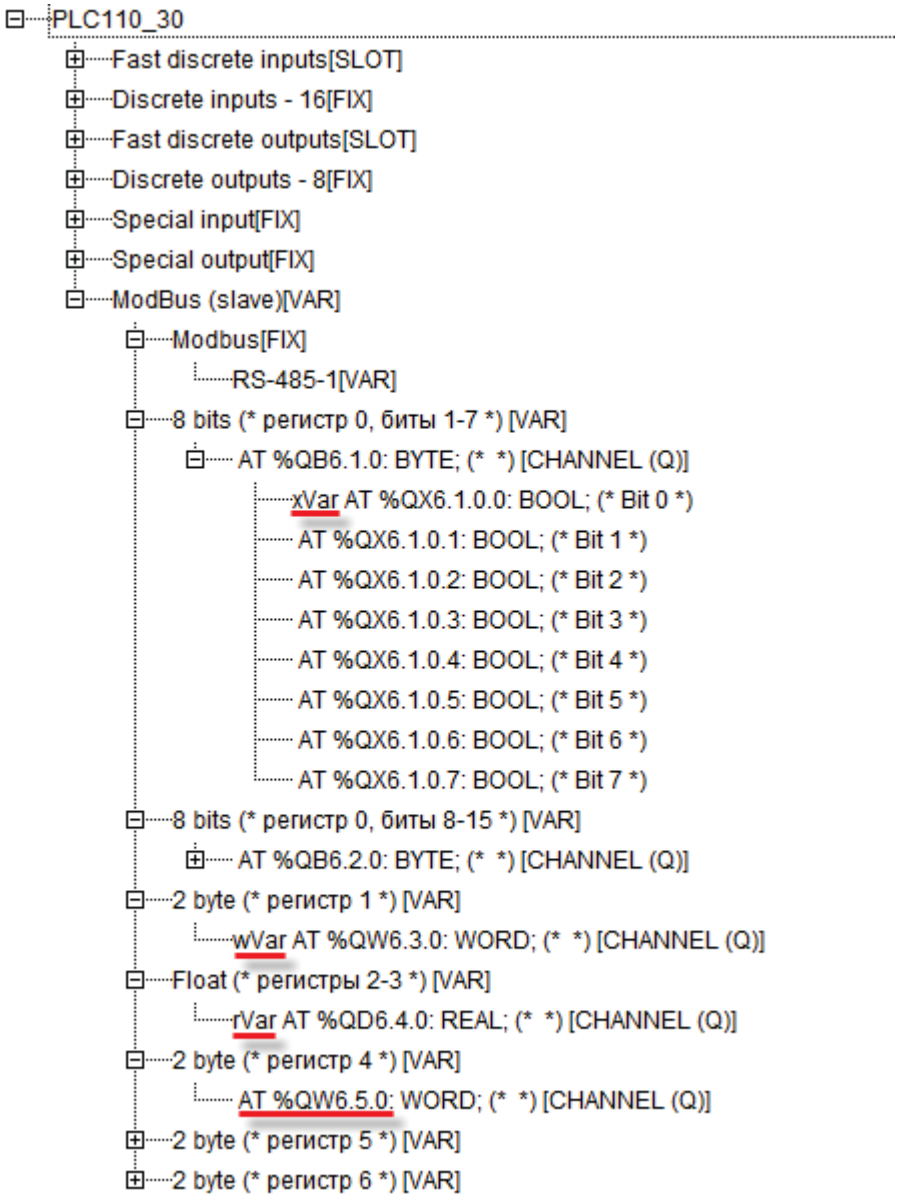


Рис. 3.6. Внешний вид **Modbus (Slave)** с добавленными подэлементами

Обратите внимание, что нумерация регистров в среде CoDeSys всегда начинается с нуля, при этом каждый регистр физически занимает два байта (16 бит). В связи с этим, переменная типа **REAL** займет два регистра (с адресами 2 и 3). Переменная типа **STRING**, которой соответствует три **2 byte** элемента, займет регистры с адресами 4-6. Это необходимо учитывать при настройке master-устройства.

Подробнее вопросы адресации рассмотрены в **Руководстве пользователя ПЛК**.

5. Программа **PLC_PRG** будет выглядеть следующим образом:

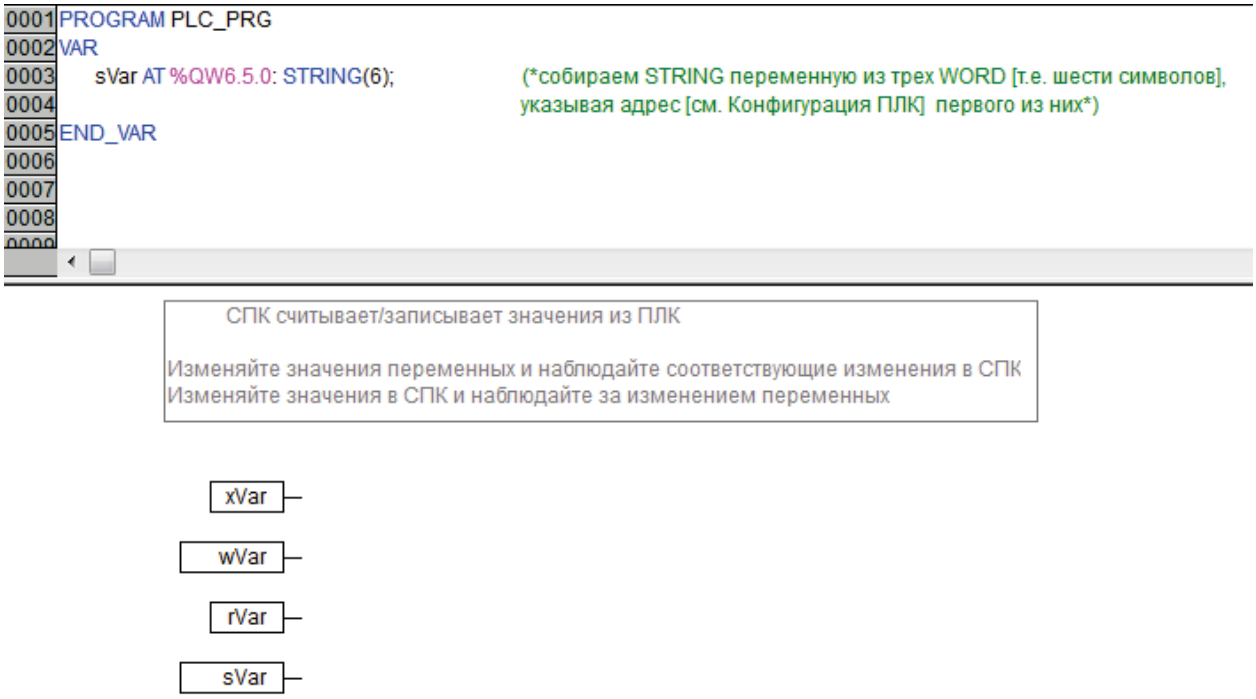


Рис. 3.7 Код программы **PLC_PRG**

На этом настройка **ПЛК (slave)** завершена.

Обратите внимание, что проект не содержит каких-либо операций и используется только для отображения и ввода значений. Пользователь должен создать программу для реализации необходимым алгоритмов.

3.3. Настройка СПК (master)

1. Создайте новый проект **CODESYS 3.5** для **СПК207** с программой **PLC_PRG** на языке **CFC**.

2. Добавьте в проект объединение с именем **Real_Word**:

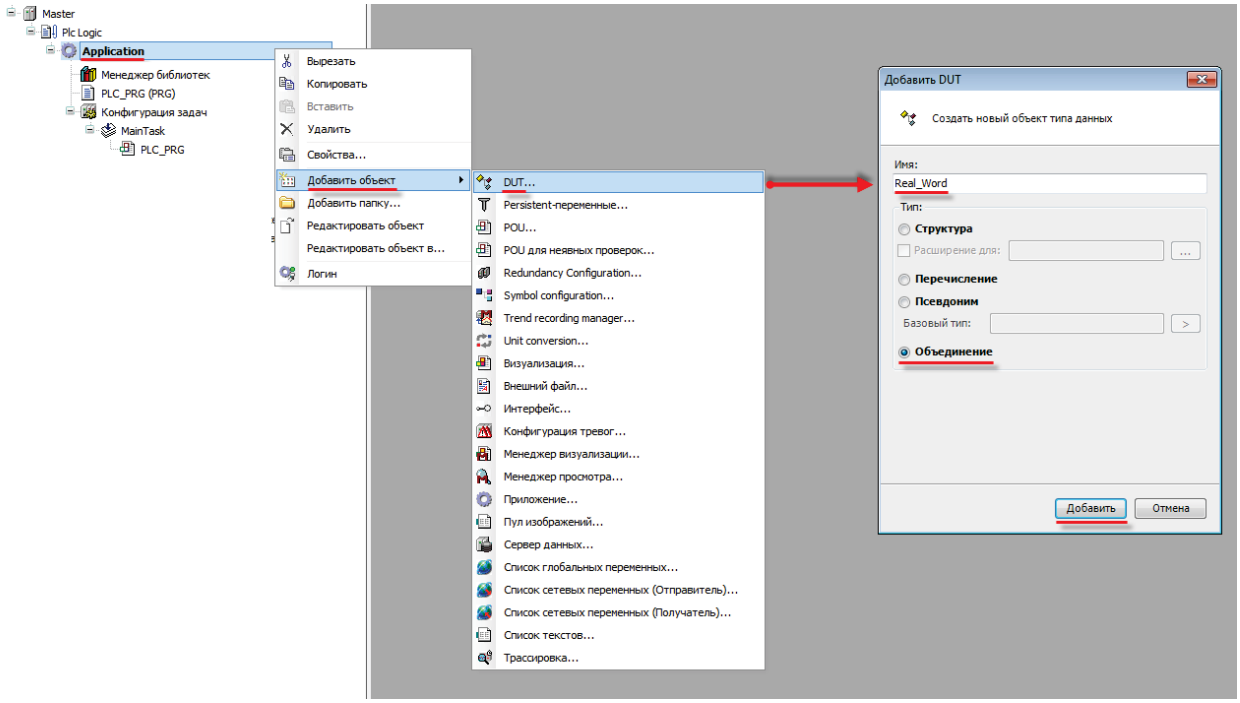


Рис. 3.8. Добавление в проект объединения

В объединении объявите переменную **rRealValue** типа **REAL** и массив **awModbusReal** типа **WORD**, содержащий два элемента:

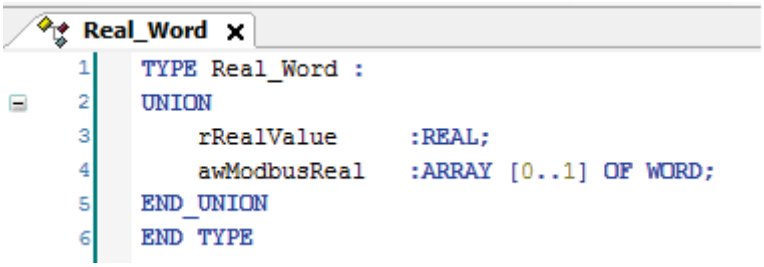


Рис. 3.9. Объявление переменных объединения

3. Добавьте в проект объединение с именем **String_Word**.

В объединении объявите переменную **sStringValue** типа **STRING** и массив **awModbusString** типа **WORD**, содержащий три элемента (**STRING** сможет содержать до 6 символов, поскольку каждый **WORD** может содержать два символа):

```
TYPE String_Word :
UNION
    awModbusString      :ARRAY [0..2] OF WORD;
    sStringValue        :STRING;
END_UNION
END_TYPE
```

Рис. 3.10. Объявление переменных объединения

4. Объявите в программе **PLC_PRG** девять переменных – 4 из них будут использоваться для отображения данных, считанных из ПЛК, еще 4 – для ввода данных, которые будут записаны в ПЛК. Последняя переменная будет являться триггером записи.

PLC_PRG

```
PROGRAM PLC_PRG
VAR
    (*данные, из считываемые ПЛК*)
    xVarRead:       BOOL;
    wVarRead:       WORD;
    rVarRead:       Real_Word;
    sVarRead:       String_Word;

    (*данные, записываемые в ПЛК*)
    xVarWrite:      BOOL;
    wVarWrite:      WORD;
    rVarWrite:      Real_Word;
    sVarWrite:      String_Word;

    xTrigger:       BOOL; // триггер записи
END_VAR
```

Рис. 3.11. Объявление переменных программы

5. Код программы будет выглядеть следующим образом:

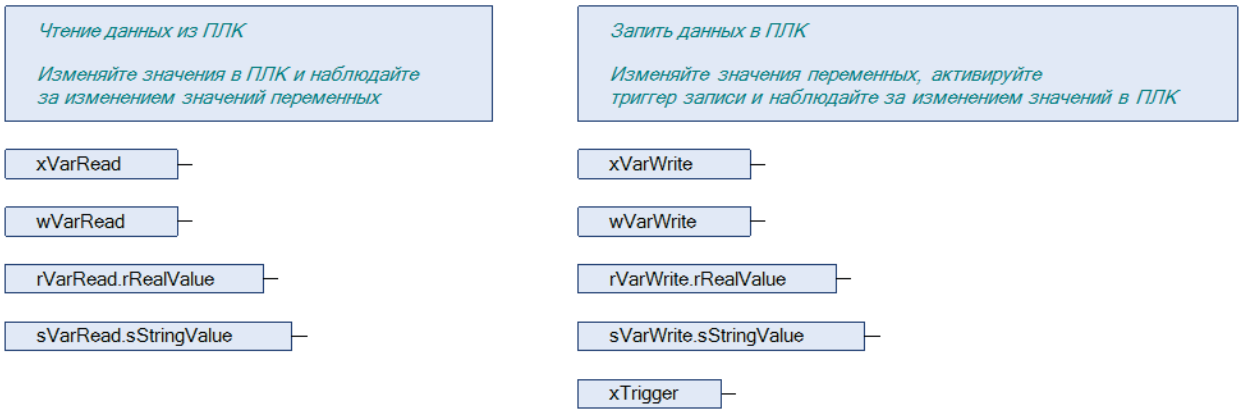


Рис. 3.12. Код программы на языке **CFC**

6. Добавьте в проект устройство **Modbus COM**. ***Обратите внимание***, что версия компонента не должна превышать версию таргет-файла СПК. Подробнее см. в документе **СПК. Modbus**.

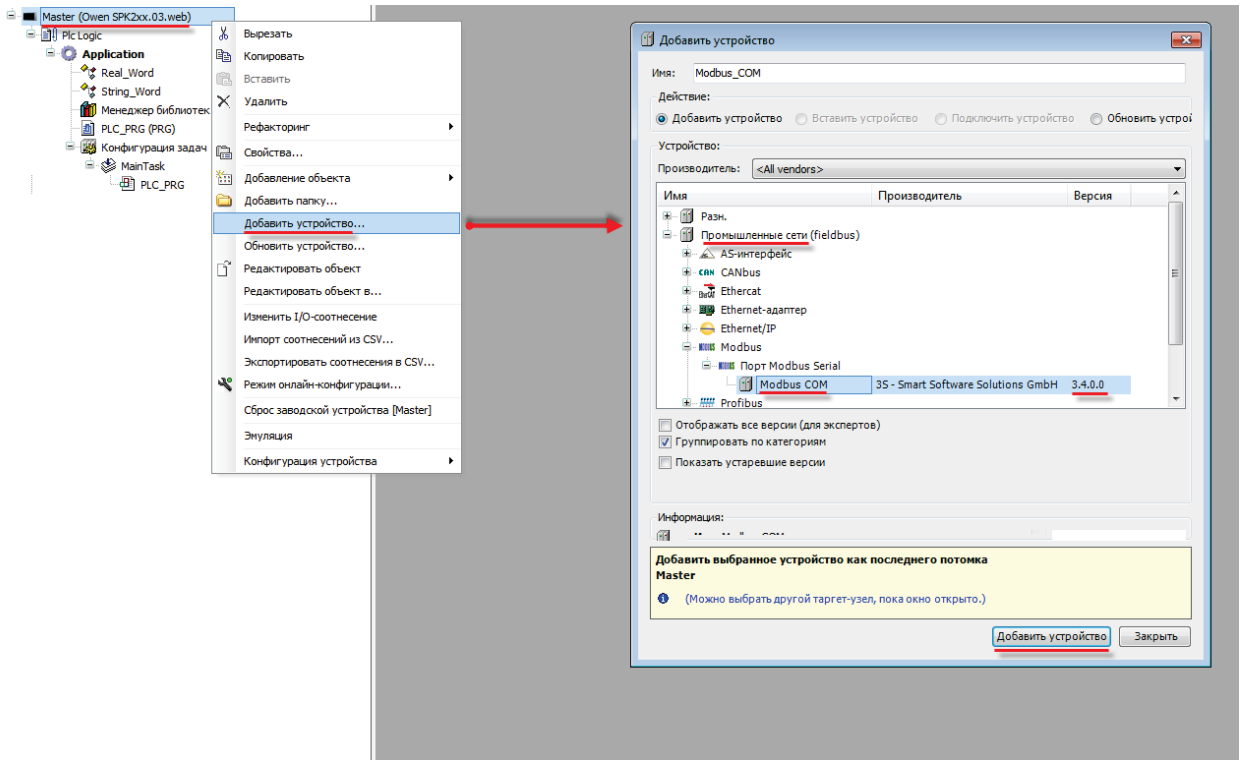


Рис. 3.13. Добавление устройства **Modbus COM**

В конфигурации COM-порта укажите сетевые настройки в соответствии с табл. 3.1, а также номер порта. COM-порту **2** (нумерация на корпусе СПК) будет соответствовать номер **3** в среде CODESYS 3.5 (см. п. 2.1).

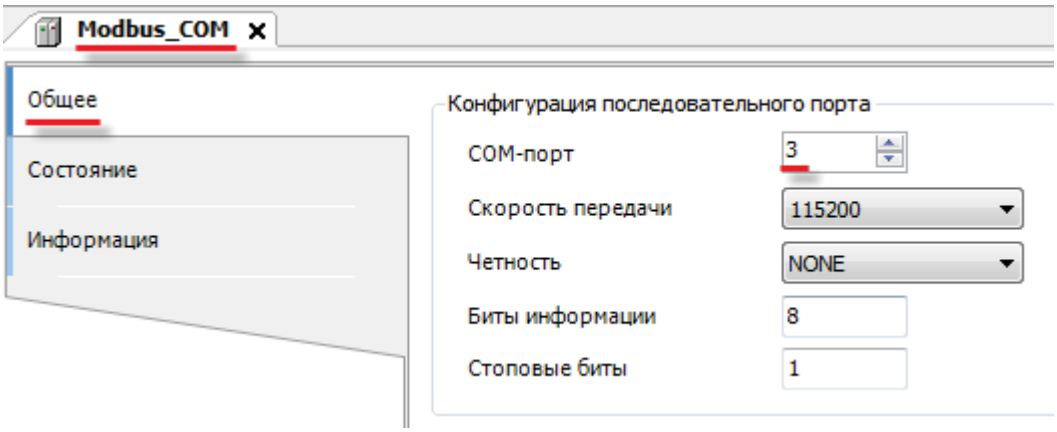


Рис. 3.14. Настройки COM-порта **COM2**

7. В COM-порт добавьте компонент **Modbus Master**. ***Обратите внимание***, что версия компонента не должна превышать версию таргет-файла СПК.

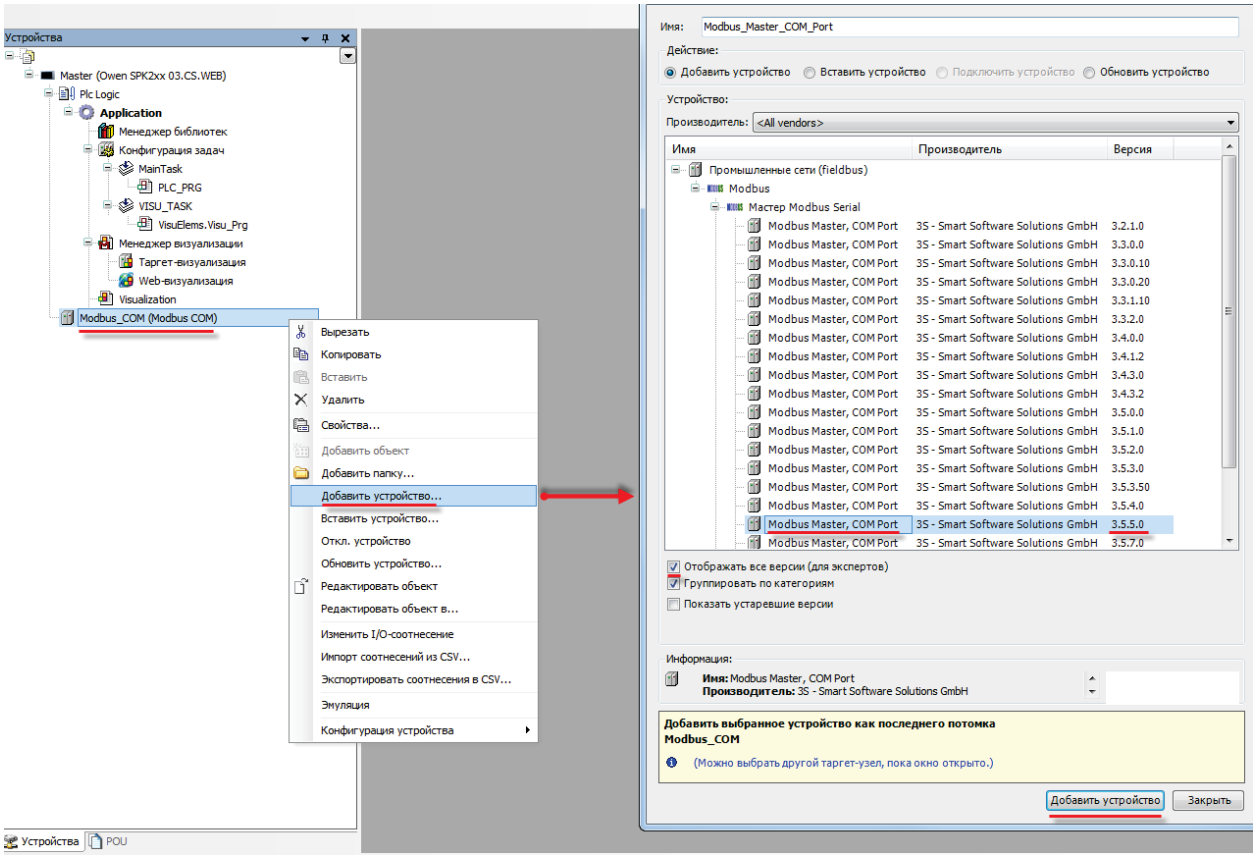


Рис. 3.15. Добавление компонента **Modbus Master**

В настройках компонента поставьте галочку **Автоперезапуск соединения**.

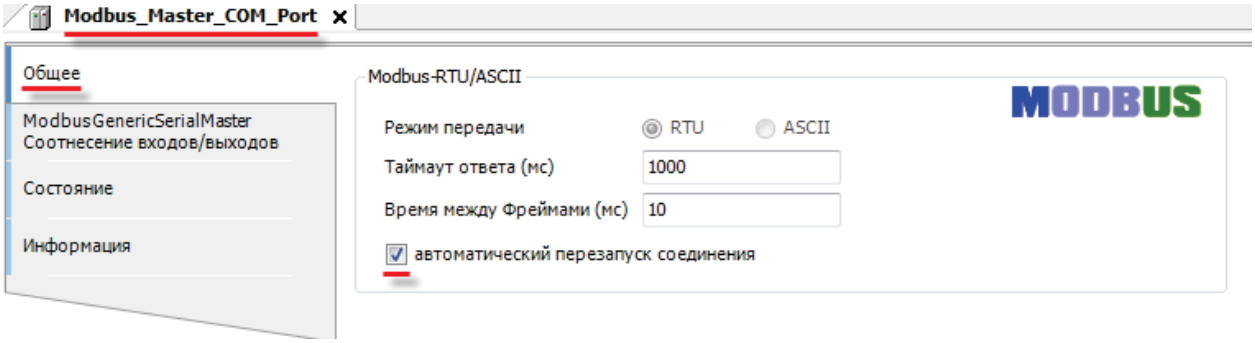


Рис. 3.16. Настройка компонентов **Modbus Master**

8. В **Modbus Master** добавьте компонент **Modbus Slave**. ***Обратите внимание***, что версия компонента не должна превышать версию таргет-файла СПК. Подробнее см. в документе **СПК. Modbus**.

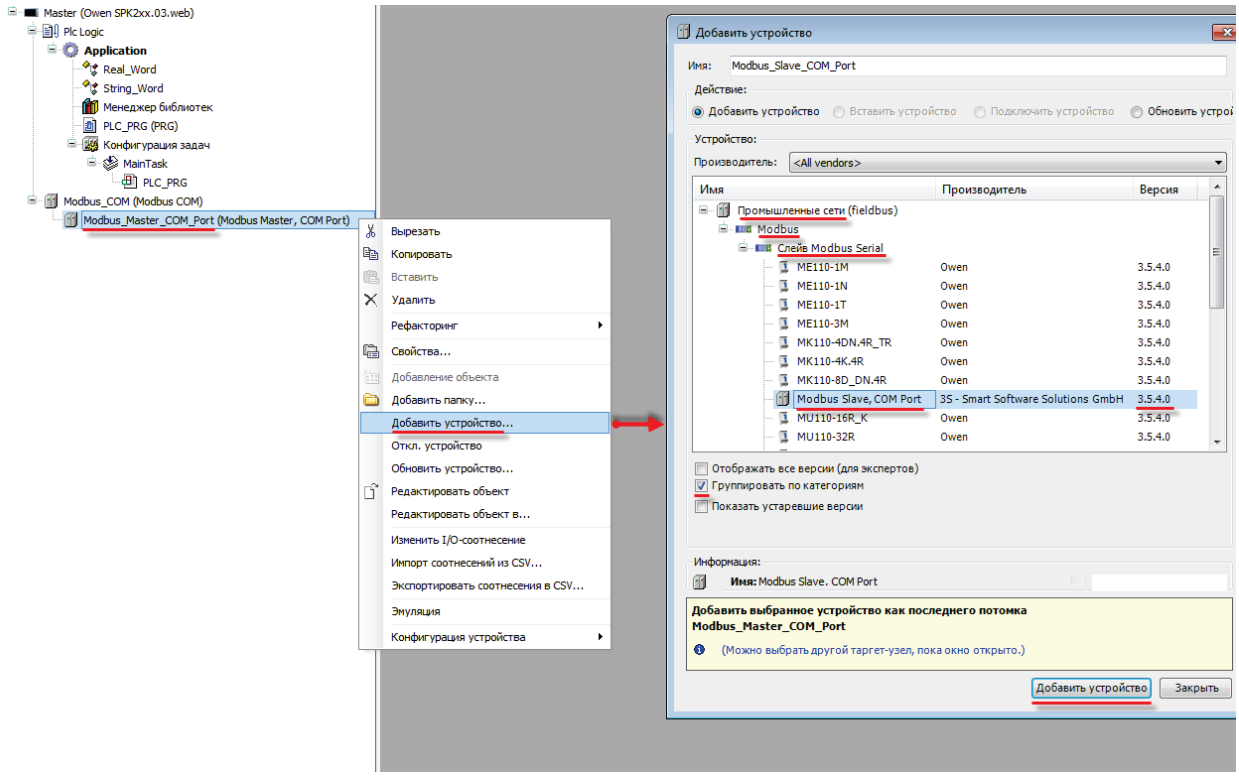


Рис. 3.17. Добавление компонента **Modbus Slave** в проект

В настройках компонента на вкладке **Общее** укажите адрес slave-устройства в соответствии с табл. 3.1.

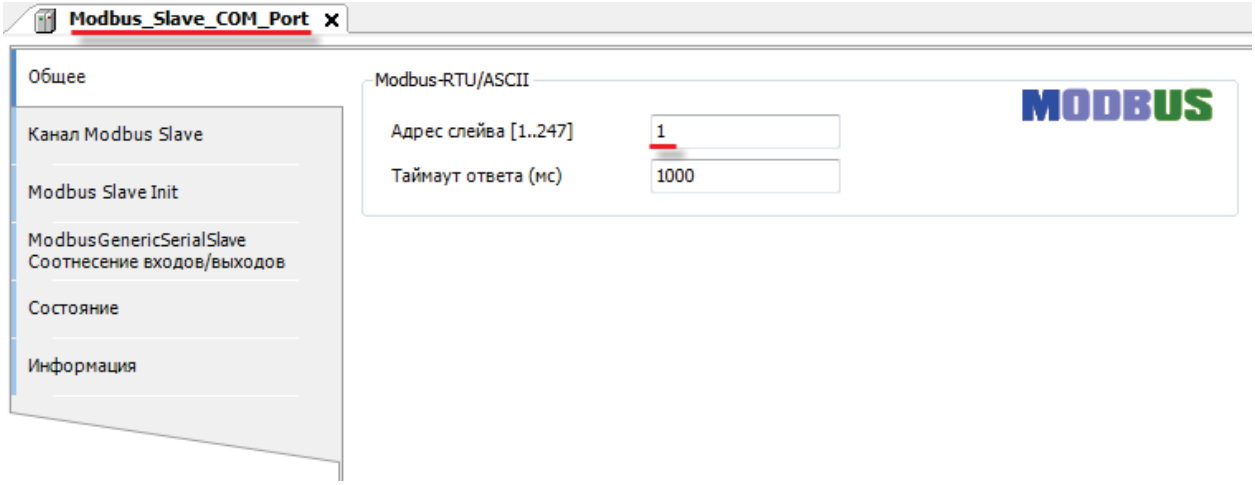
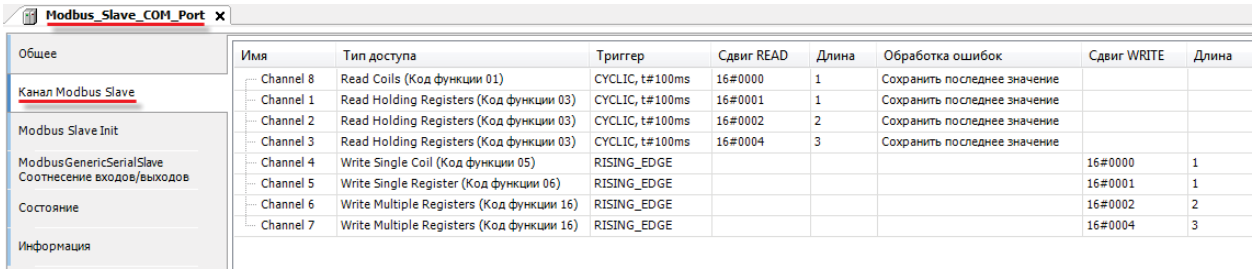


Рис. 3.18. Настройки компонента **Modbus Slave** в проект

На вкладке **Канал Modbus Slave** создайте 8 каналов – 4 из них будут использоваться для чтения переменных, 4 – для записи. Чтение будет осуществляться циклически, запись – по переднему фронту триггера (**RISING_EDGE**). Используемые функции соответствуют типам данных, адреса регистров настроены согласно табл. 3.2.



Имя	Тип доступа	Триггер	Сдвиг READ	Длина	Обработка ошибок	Сдвиг WRITE	Длина
Channel 8	Read Coils (Код функции 01)	CYCLIC, t#100ms	16#0000	1	Сохранить последнее значение		
Channel 1	Read Holding Registers (Код функции 03)	CYCLIC, t#100ms	16#0001	1	Сохранить последнее значение		
Channel 2	Read Holding Registers (Код функции 03)	CYCLIC, t#100ms	16#0002	2	Сохранить последнее значение		
Channel 3	Read Holding Registers (Код функции 03)	CYCLIC, t#100ms	16#0004	3	Сохранить последнее значение		
Channel 4	Write Single Coil (Код функции 05)	RISING_EDGE				16#0000	1
Channel 5	Write Single Register (Код функции 06)	RISING_EDGE				16#0001	1
Channel 6	Write Multiple Registers (Код функции 16)	RISING_EDGE				16#0002	2
Channel 7	Write Multiple Registers (Код функции 16)	RISING_EDGE				16#0004	3

Рис. 3.19. Настройка каналов **Modbus Slave**

На вкладке **ModbusGenericSerialSlave Соотнесение входов/выходов** привяжите к каналам переменные программы в соответствии с табл. 3.2. Не забудьте у параметра **Всегда обновлять переменные** выставить значение **Включено 2**.

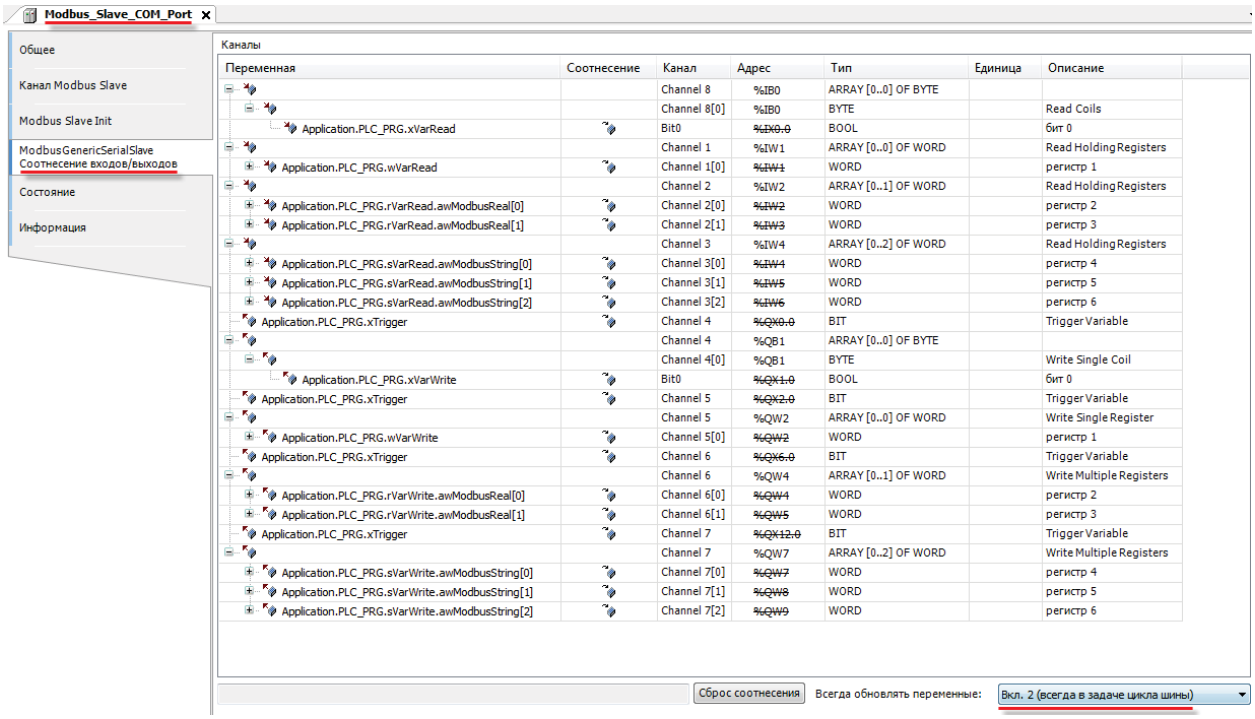


Рис. 3.20. Привязка переменных к каналу

На этом настройка **СПК (master)** завершена.

Обратите внимание, что проект не содержит каких-либо операций и используется только для отображения и ввода значений. Пользователь должен создать программу для реализации необходимым алгоритмов.

3.4. Работа с примером

Загрузите проекты в оба устройства и запустите их.

Изменяйте значения переменных в ПЛК и наблюдайте соответствующие изменения в программе СПК:

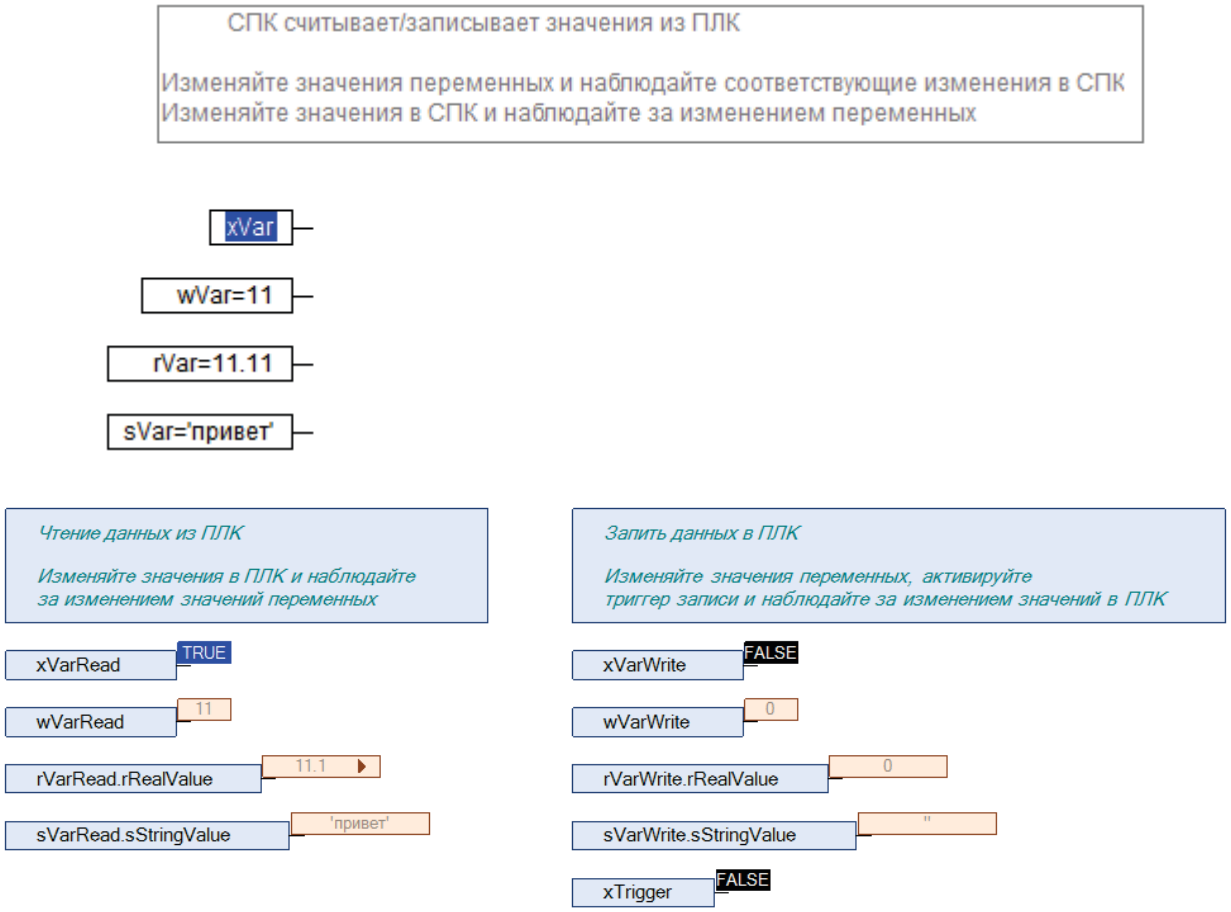


Рис. 3.21. СПК считывает данные из ПЛК

Измените значения **write** переменных СПК и активируйте триггер записи. Наблюдайте соответствующие изменения в программе ПЛК. Также новые значения будут считаны в **read** переменные программы СПК.

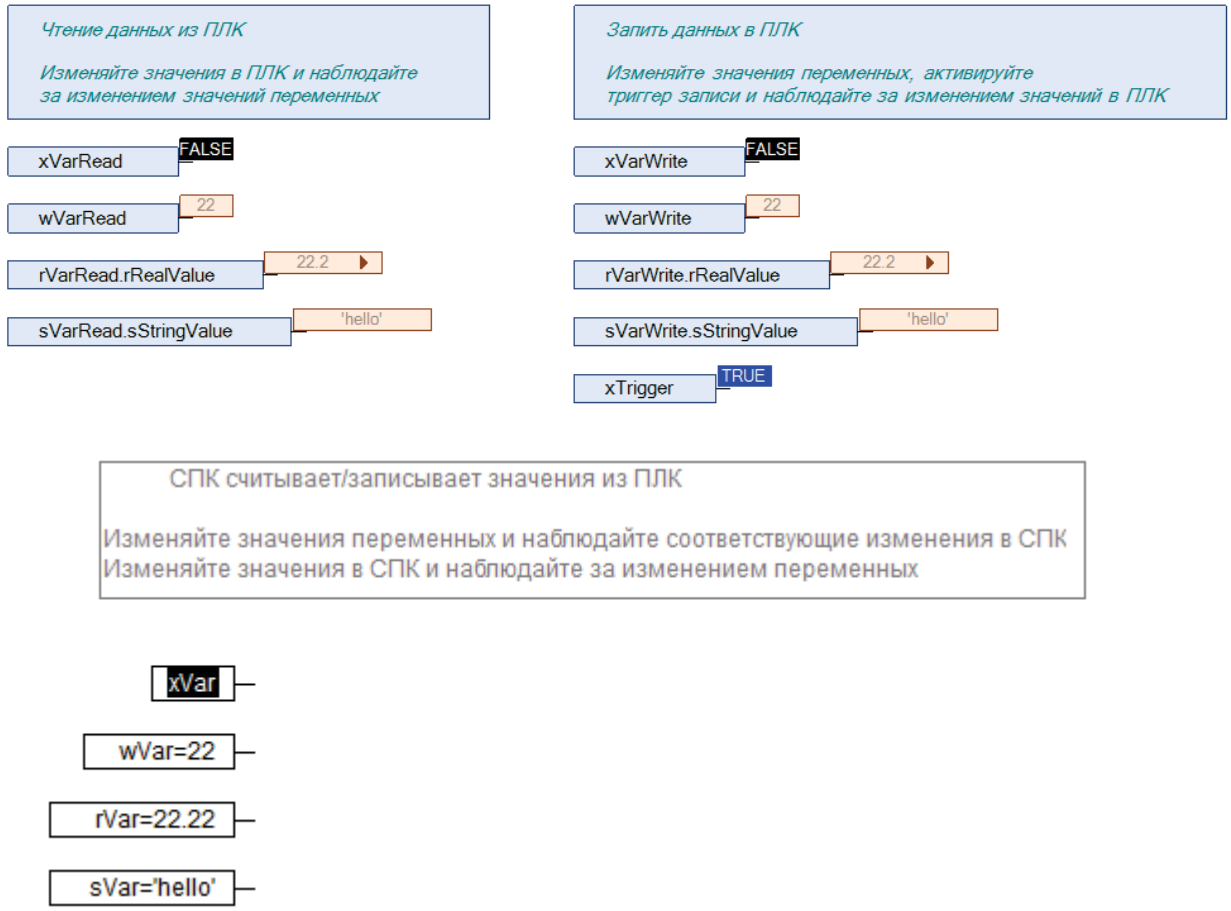


Рис. 3.22. СПК записывает данные в ПЛК

4. Modbus RTU. СПК – slave, ПЛК – master

4.1. Описание примера

Этот пример посвящен настройке обмена данными между сенсорным панельным контроллером **СПК207** и контроллером **ПЛК110 [M02]** по протоколу **Modbus RTU**. В этом примере СПК выполняет функцию **Slave**, а ПЛК – **Master**.

Основные характеристики используемых устройств приведены в табл. 4.1. Используемые в примере переменные описаны в табл. 4.2.

Табл. 4.1. Характеристики устройств

Устройство	**СПК207**	**ПЛК110 [M02]**
Функция	Slave	Master
Используемый порт (нумерация на корпусе)	RS-485 (COM2)	RS-485 (1)
Настройки обмена	115200, 8 бит, 1 стоп бит, без контроля четности	
Slave ID	1	-
Таргет	3.5.4.20 (023)	PLC110.30-M v2 (версия 3.08)
Среда разработки проекта	CODESYS 3.5 SP7 Patch4	CoDeSys 2.3.9.41
Название файла проекта	ModbusRTUslave.projectarchive	ModbusRTUmaster.pro

Табл. 4.2. Список переменных

СПК207 (Slave)			ПЛК110 [M02] (Master)
Переменные, которые считывает Master	**Адрес input регистра**	**Тип данных**	**Переменные ПЛК**
wVarFromSPK	0	WORD	wVarFromSPK
rVarFromSPK	1-2	REAL	rVarFromSPK
sVarFromSPK	3-5	STRING(6)	sVarFromSPK
Переменные, которые записывает Master	**Адрес holding регистра**	**Тип данных**	**Переменные ПЛК**
wVarToSPK	0	WORD	wVarToSPK
rVarToSPK	1-2	REAL	rVarToSPK
sVarToSPK	3-5	STRING(6)	sVarToSPK

Проекты примера доступны для скачивания: Example_SpkModbusRtuSlave.zip

4.2. Настройка СПК (slave)

1. Создайте новый проект **CODESYS 3.5** для **СПК207** с программой **PLC_PRG** на языке **CFC**.

2. Добавьте в проект объединение с именем **Real_Word**:

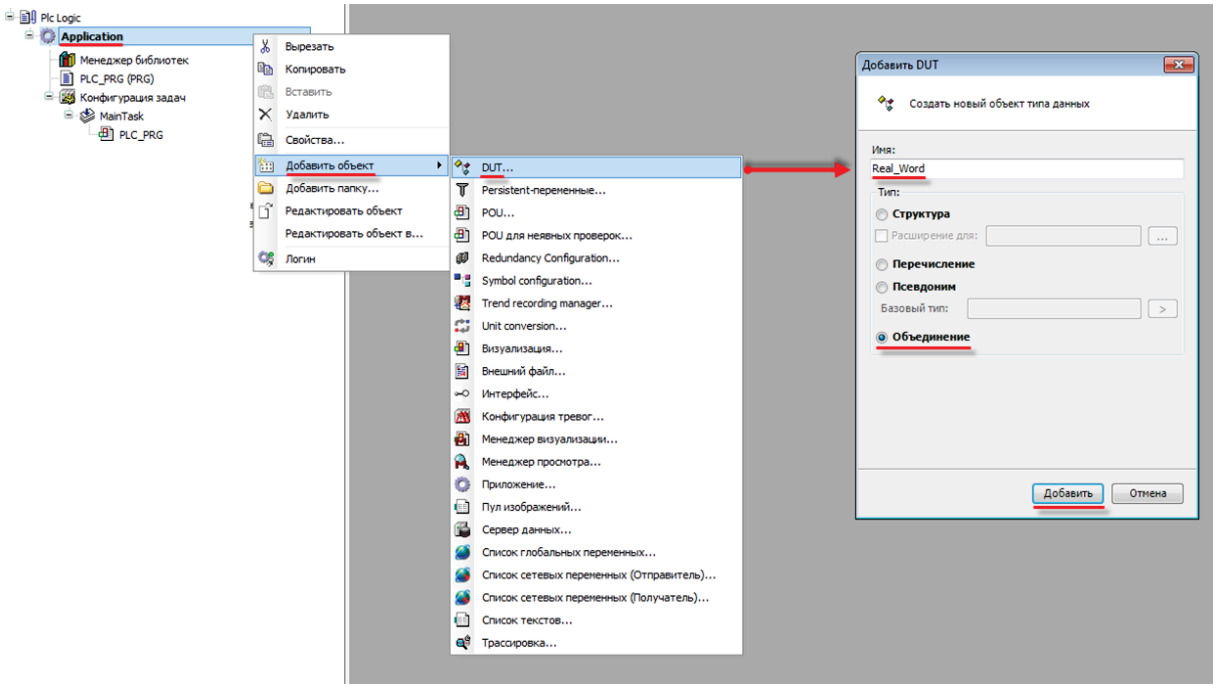


Рис. 4.1. Добавление в проект объединения

В объединении объявите переменную **rRealValue** типа **REAL** и массив **awModbusReal** типа **WORD**, содержащий два элемента:

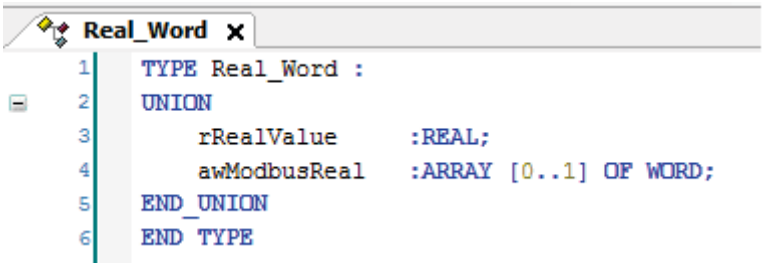


Рис. 4.2. Объявление переменных объединения

3. Добавьте в проект объединение с именем **String_Word**.

В объединении объявите переменную **sStringValue** типа **STRING** и массив **awModbusString** типа **WORD**, содержащий три элемента (**STRING** сможет содержать до 6 символов, поскольку каждый **WORD** может содержать два символа):

```
TYPE String_Word :
UNION
    awModbusString      :ARRAY [0..2] OF WORD;
    sStringValue        :STRING;
END_UNION
END_TYPE
```

Рис. 4.3. Объявление переменных объединения

4. Объявите в программе **PLC_PRG** 6 переменных – 3 из них будут считываться ПЛК, 3 – записываться ПЛК. ***Обратите внимание*** на п. 2.3.

PLC_PRG

```
PROGRAM PLC_PRG
VAR
    (*данные, считываемые ПЛК*)
    wVarFromSPK:    WORD;
    rVarFromSPK:    Real_Word;
    sVarFromSPK:    String_Word;

    (*данные, записываемые ПЛК*)
    wVarToSPK:      WORD;
    rVarToSPK:      Real_Word;
    sVarToSPK:      String_Word;
END_VAR
```

Рис. 4.4. Объявление переменных программы

5. Код программы будет выглядеть следующим образом:

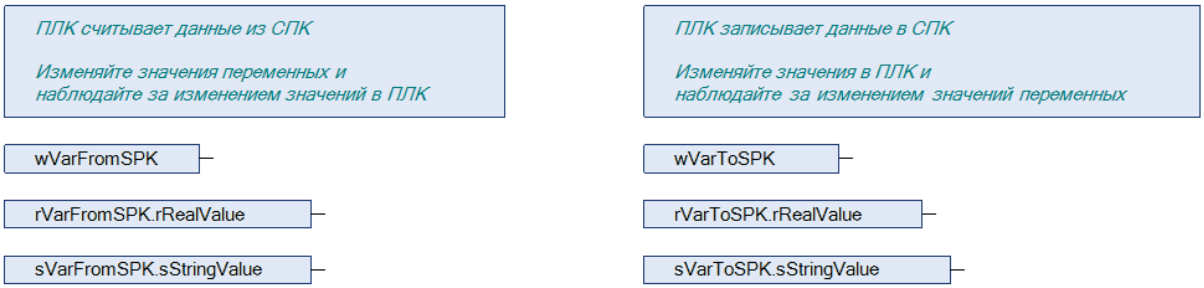


Рис. 4.5. Код программы на языке **CFC**

6. Добавьте в проект устройство **Modbus COM**. ***Обратите внимание***, что версия компонента не должна превышать версию таргет-файла СПК. Подробнее см. в документе **СПК. Modbus**.

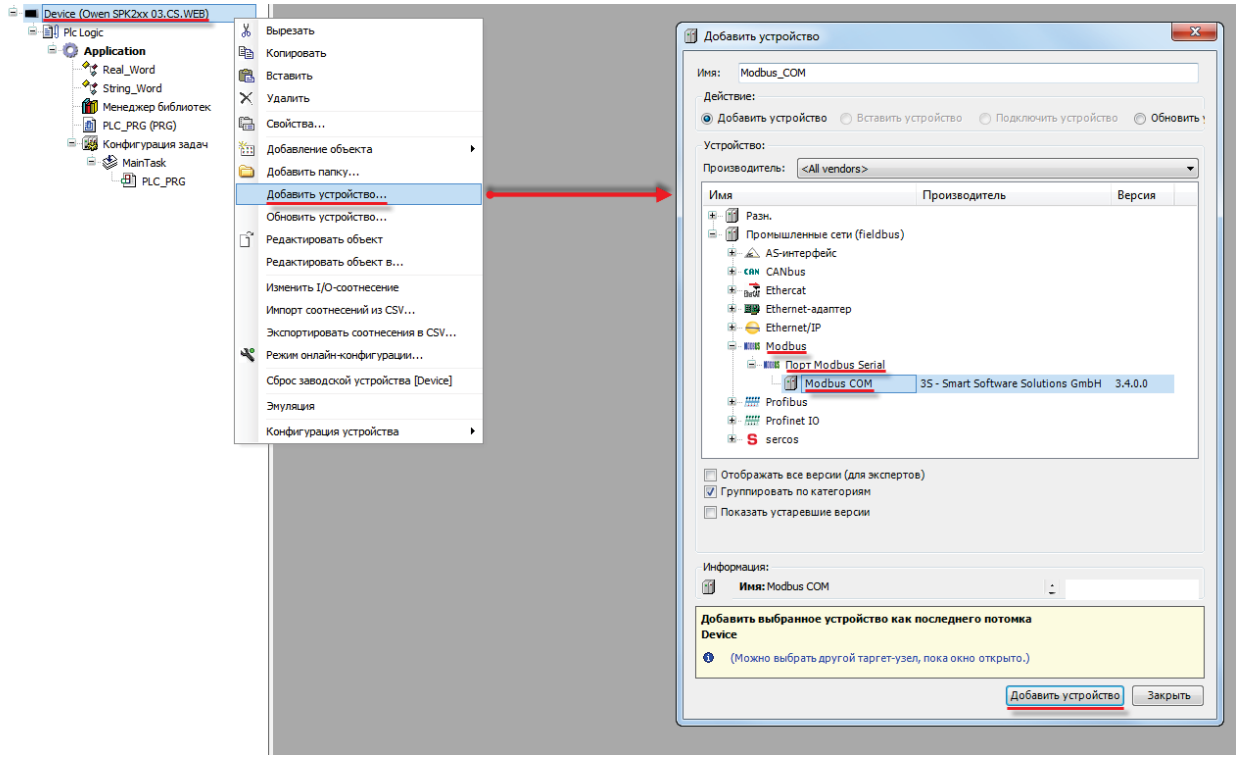


Рис. 4.6. Добавление устройства **Modbus COM**

В конфигурации COM-порта укажите сетевые настройки в соответствии с табл. 4.1, а также номер порта. COM-порту **2** будет соответствовать номер **3** (см. п. 2.1).

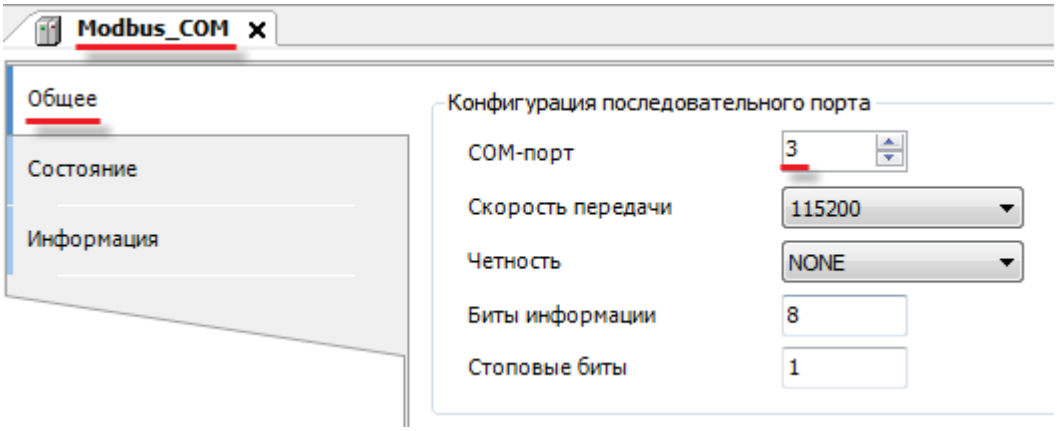


Рис. 4.7. Настройки COM-порта **COM2**

7. В COM-порт добавьте компонент **Modbus Serial Device**. ***Обратите внимание***, что версия компонента не должна превышать версию таргет-файла СПК.

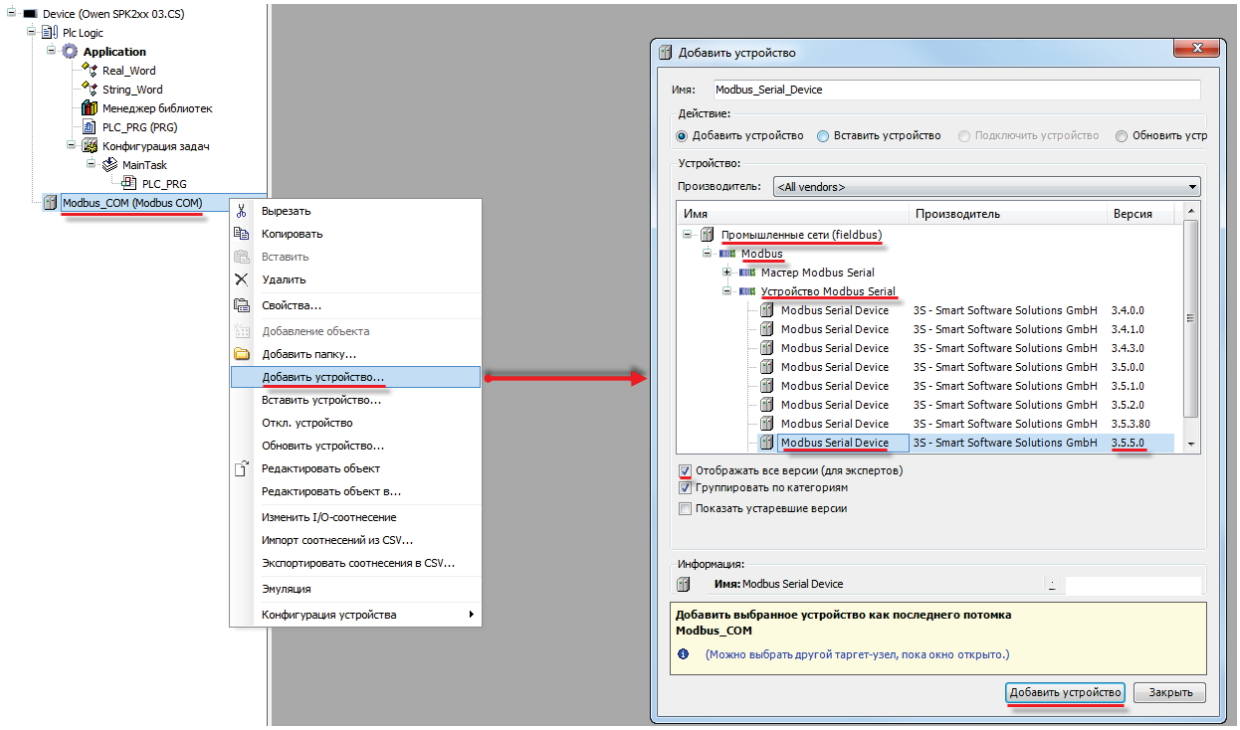


Рис. 4.8. Добавление компонента **Modbus Serial Device**

В настройках компонента на вкладке **Modbus Serial Device** укажите адрес slave-устройства (**1** в соответствии с табл. 4.1).

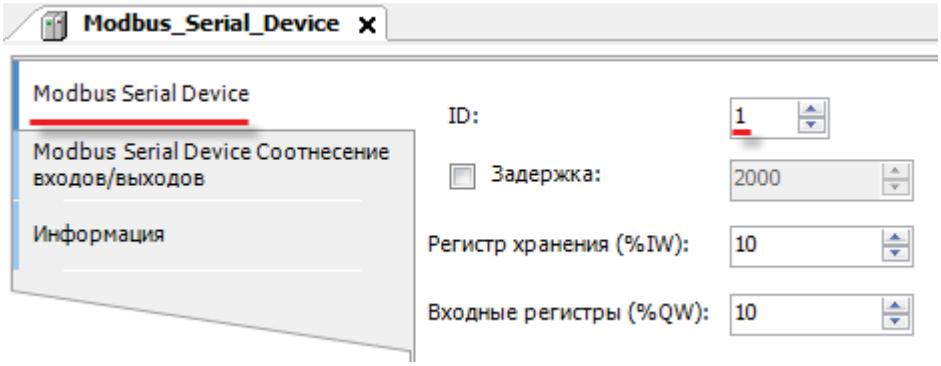


Рис. 4.9. Настройки компонента **Modbus Serial Device**

На вкладке **Modbus Serial Device Соотнесение входов/выходов** привяжите к регистрам переменные программы в соответствии с табл. 4.2.

Обратите внимание, что канал **Inputs** содержит **Holding регистры**, а канал **Outputs** – **Input регистры**. ***Обратите внимание*** на порядок **WORD** для переменных типа **REAL**.

Не забудьте у параметра **Всегда обновлять переменные** выставить значение **Включено 2**.

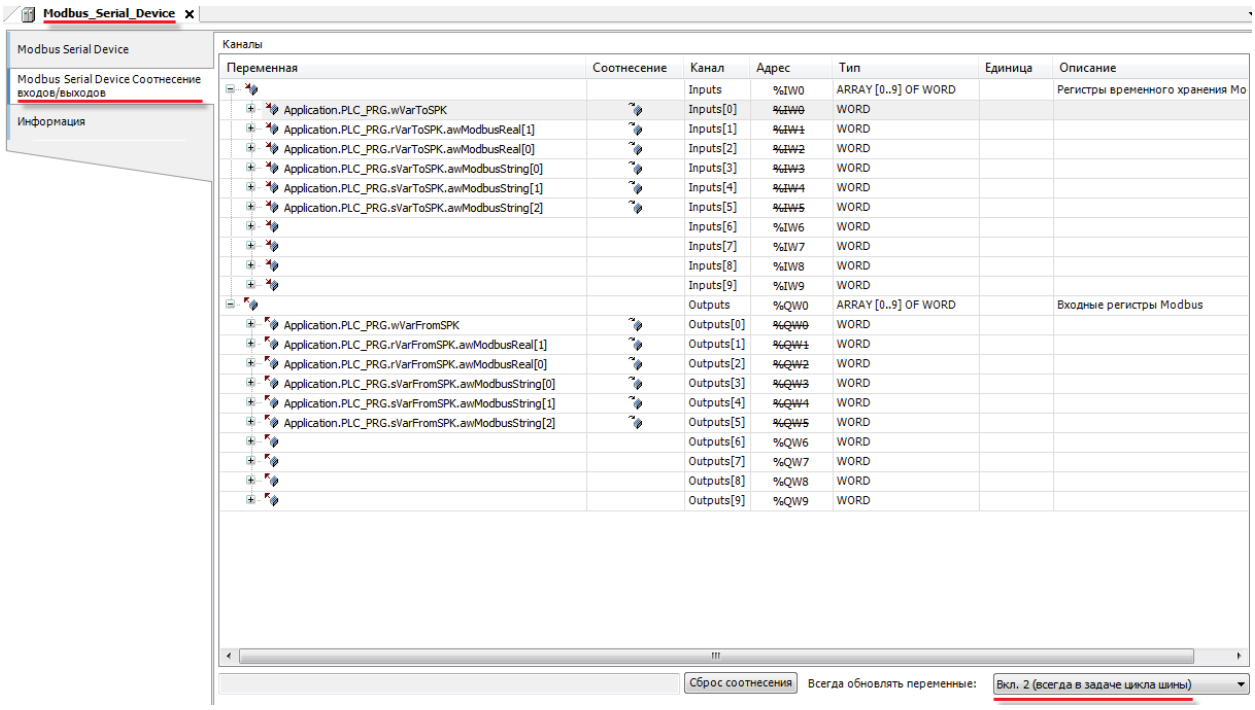


Рис. 4.10. Привязка переменных к регистрам slave-устройства

На этом настройка **СПК (slave)** завершена.

4.3. Настройка ПЛК (master)

1. Создайте новый проект **CoDeSys 2.3** для **ПЛК110** с программой **PLC_PRG** на языке **CFC**.

Нажмем **ПКМ** на название контроллера (в нашем примере - **PLC110_30**) и добавим подэлемент **Modbus (Master)**:

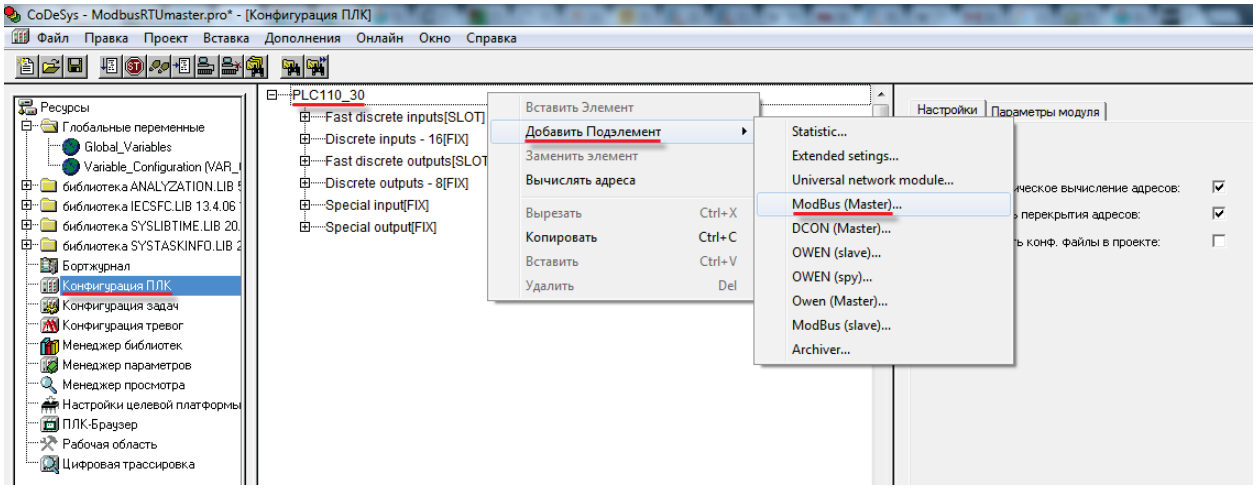


Рис. 4.11. Конфигурация ПЛК. Добавление **Modbus (Master)**

Этот элемент не нуждается в настройках.

2. Выберем порт ПЛК, который будет использоваться для связи с СПК. Для этого в элементе **Modbus (Master)** нажмем **ПКМ** на порт **Debug RS-232** и в контекстном меню выберем команду **Заменить элемент**. В нашем примере мы используем порт **RS-485-1**.

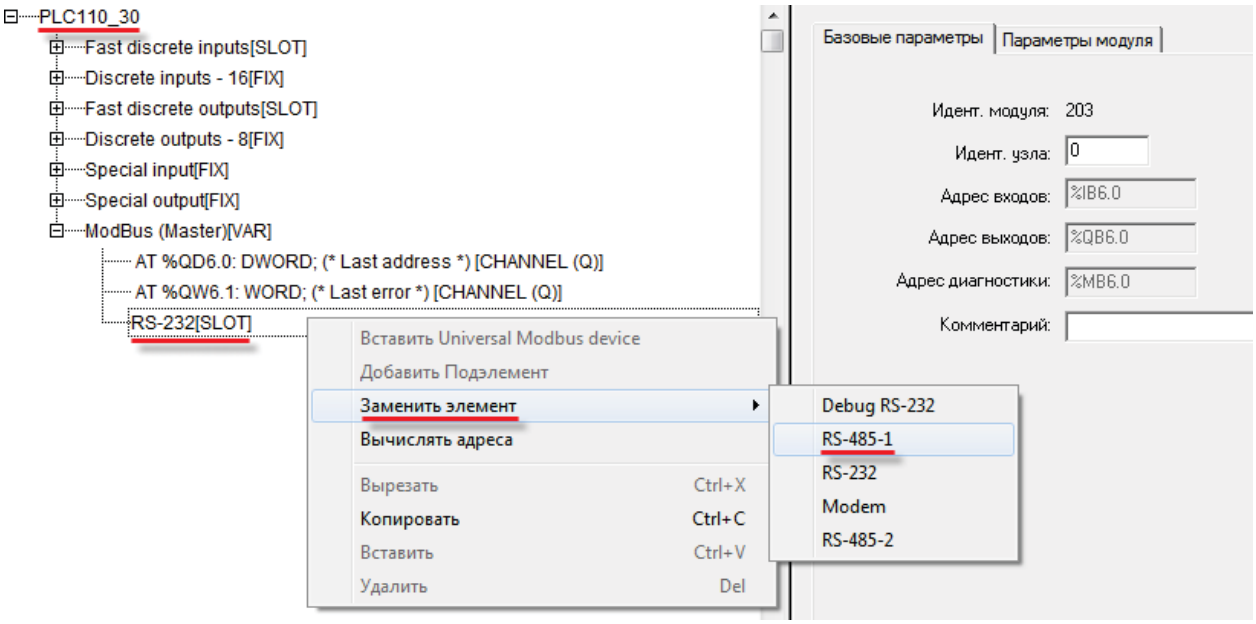


Рис. 4.12. Конфигурация ПЛК. Выбор порта

Настройки порта по умолчанию соответствуют тем настройкам, которые мы задали СПК (согласно табл. 4.1): скорость – **115200**, бит данных – **8**, стоп бит – **1**, контроль четности – **нет**. Для параметра **Frame Oriented** выберем значение **RTU**.

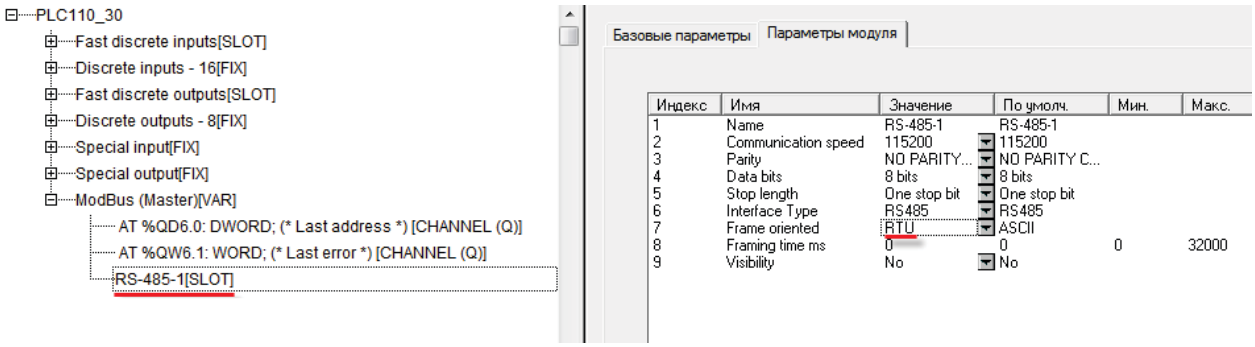


Рис. 4.13. Конфигурация ПЛК. Настройки порта

3. Нажмем **ПКМ** на элемент **Modbus (Master)** и добавим два подэлемента **Universal Modbus Device**:

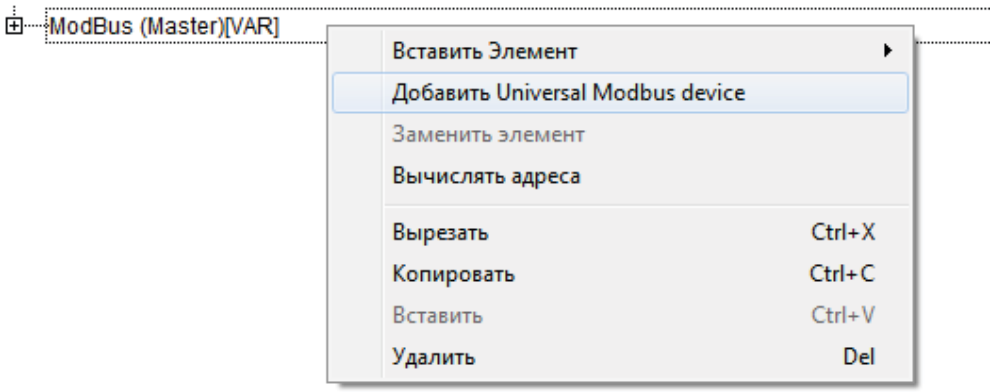


Рис. 4.14. Конфигурация ПЛК. Добавление **Universal Modbus Device**

Один из них будет использоваться для чтения значений из СПК, второй – для записи.

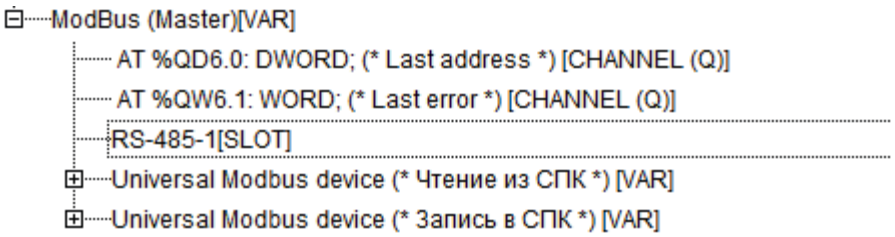


Рис. 4.15. Внешний вид **Конфигурации ПЛК** после добавления двух **Universal Modbus Device**

В настройках элемента **Чтение из СПК** укажем тип связи (**Serial**), **Slave ID** СПК (в соответствии с табл. 4.1 он равен **1**) и режим опроса (**By poll time**, т.е. циклически).

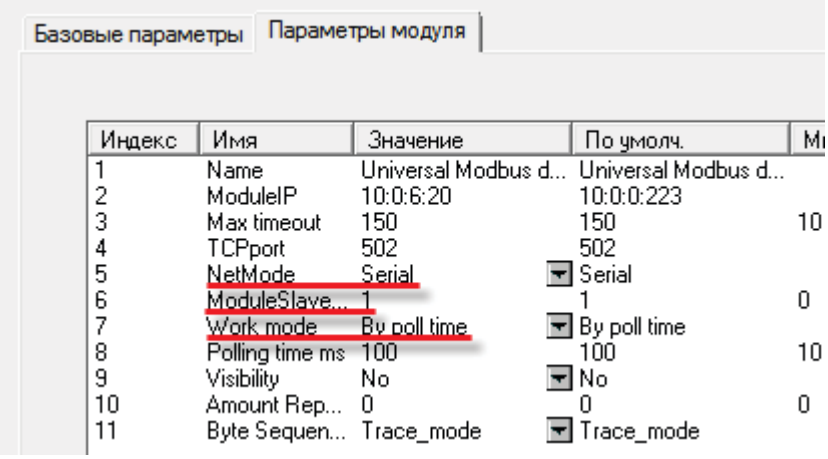


Рис. 4.16. Настройки **Universal Modbus Device (Чтение из СПК)**

В настройках элемента **Запись в СПК** укажем тип связи (**Serial**), **Slave ID** СПК (в соответствии с табл. 4.1 он равен **1**) и режим опроса (**By value change**, т.е. спорадически).

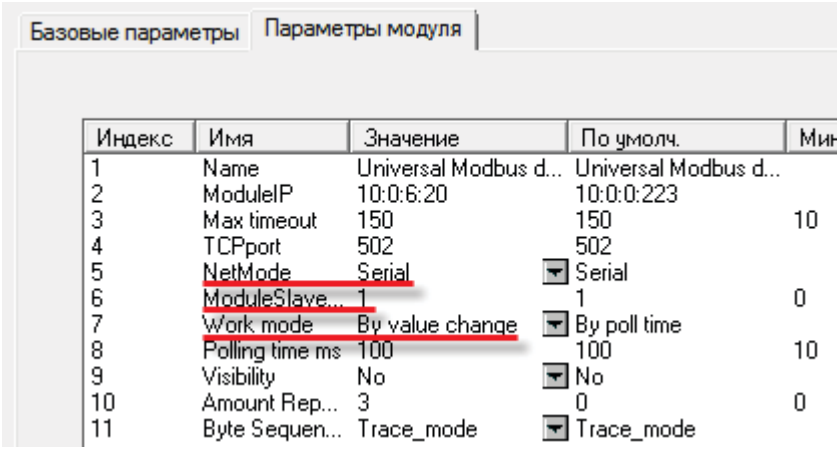


Рис. 4.17. Настройки **Universal Modbus Device (Запись в СПК)**

Нажмем **ПКМ** на элемент **Universal Modbus Device (Чтение из СПК**) и добавим в него подэлементы **Register Input Module**, **Real Input Module** и **String Input Module**.

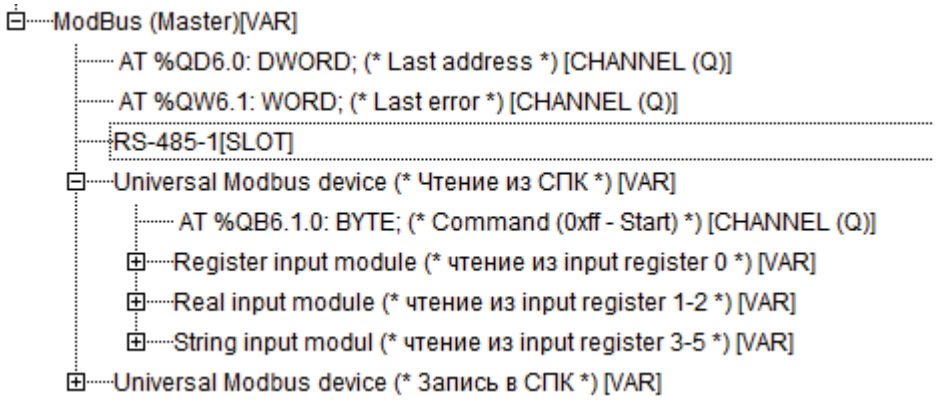


Рис. 4.18. **Universal Modbus Device (Чтение из СПК)** с добавленными **Input** модулями

Привяжем к каждому из каналов переменную (после ввода ее имени она автоматически будет добавлена в список глобальных переменных проекта). Для ввода имени переменной два раза нажмите на **AT**.

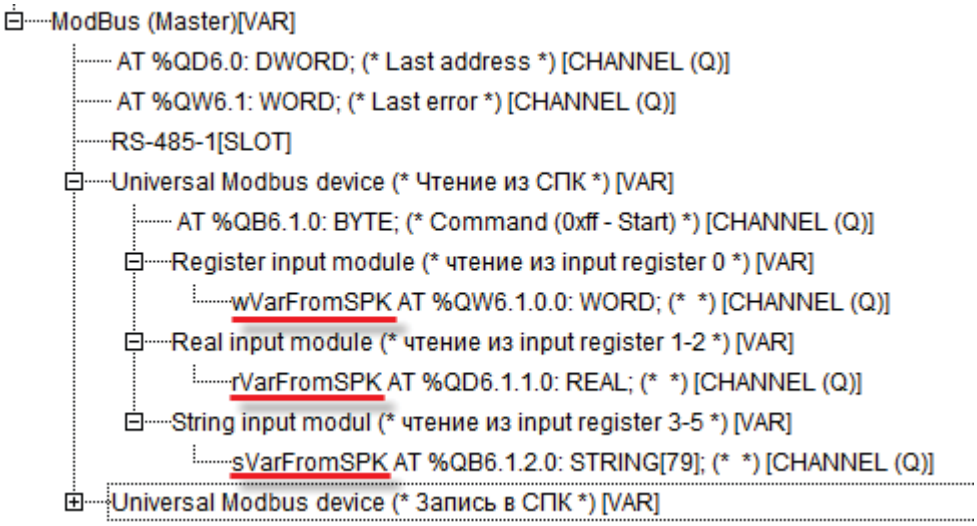


Рис. 4.19. Привязка переменных к каналам

Настройки модулей (используемые регистры СПК в соответствии с табл. 4.1 и функции Modbus) приведены ниже.

Обратите внимание, что при работе с переменными, занимающими несколько регистров СПК (тип **REAL** и **STRING**), указывается только первый из группы регистров.

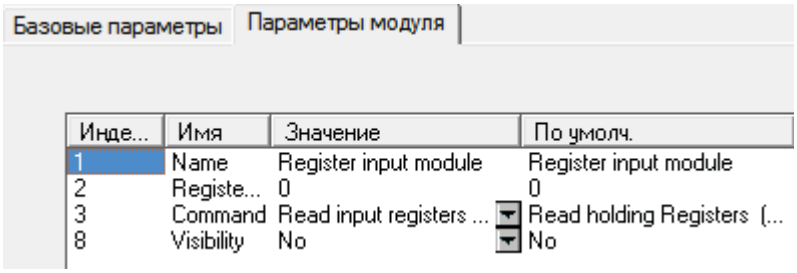


Рис. 4.20. Параметры **Register Input Module**

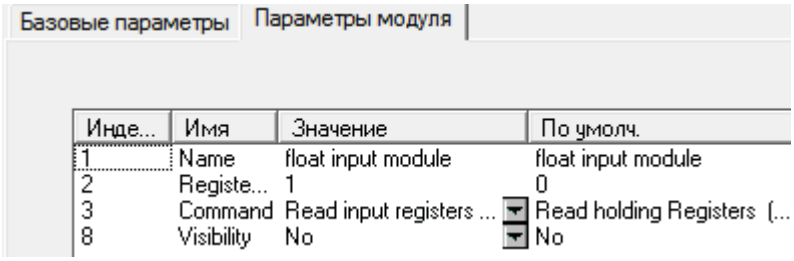


Рис. 4.21. Параметры **Real Input Module**

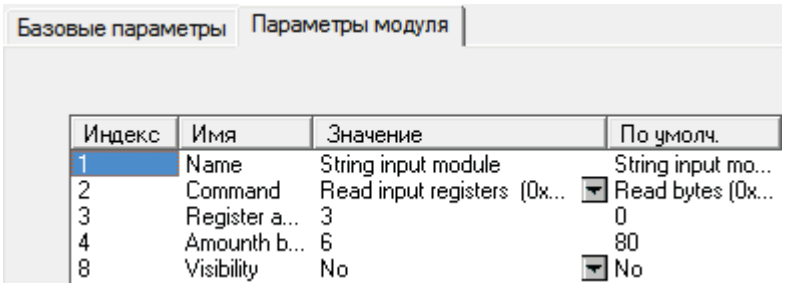


Рис. 4.22. Параметры **String Input Module**

Нажмем **ПКМ** на элемент **Запись в СПК** и добавим в него подэлементы **Register Output Module**, **Real Output Module** и **String Output Module**.

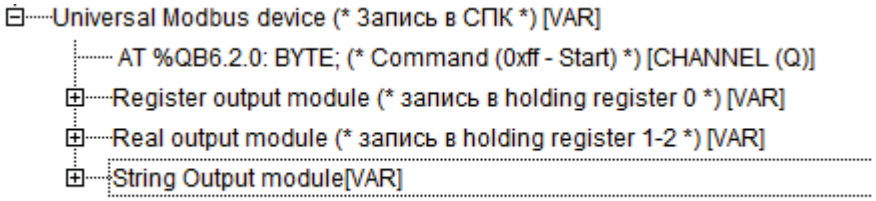


Рис. 4.23. **Universal Modbus Device (Запись в СПК)** с добавленными **Output** модулями

Привяжем к каждому из каналов переменную (после ввода ее имени она автоматически будет создана в проекте как глобальная). Для ввода имени переменной два раза нажмите на **AT**.

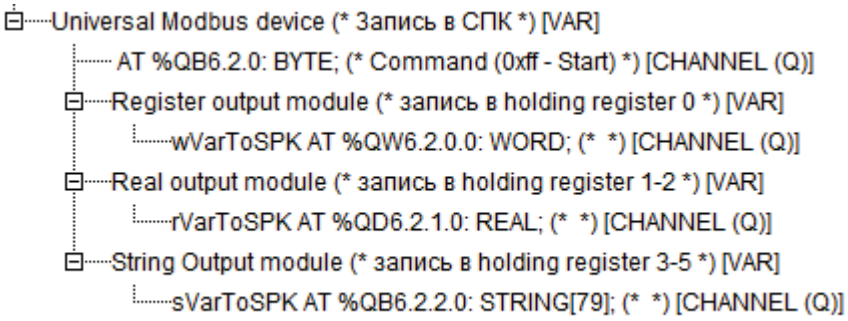


Рис. 4.24. Привязка переменных к каналам

Настройки модулей (используемые регистры СПК в соответствии с табл. 4.1 и функции Modbus) приведены ниже.

Обратите внимание, что при работе с переменными, занимающими несколько регистров СПК (тип **REAL** и **STRING**), указывается только первый из группы регистров

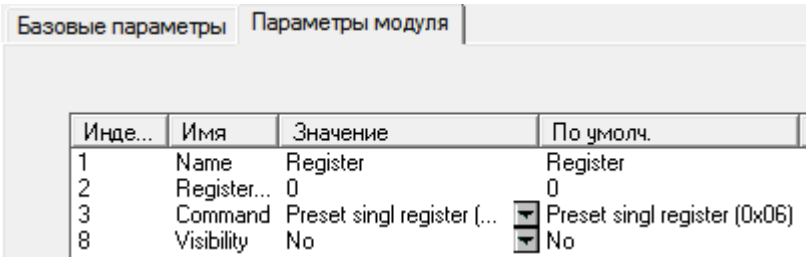


Рис. 4.25. Параметры **Register Output Module**

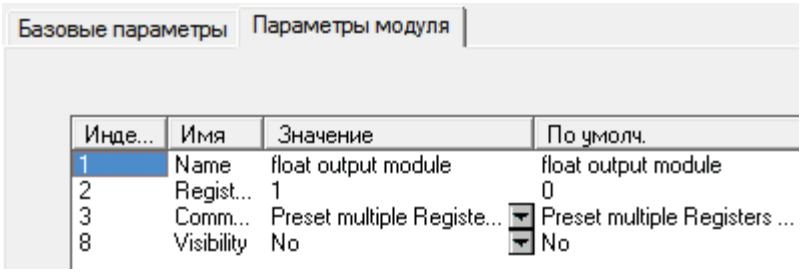


Рис. 4.26. Параметры **Real Output Module**

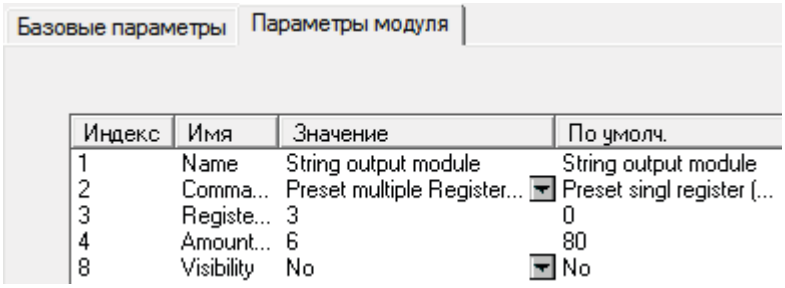


Рис. 4.27. Параметры **String Output Module**

4. Программа **PLC_PRG** будет выглядеть следующим образом:

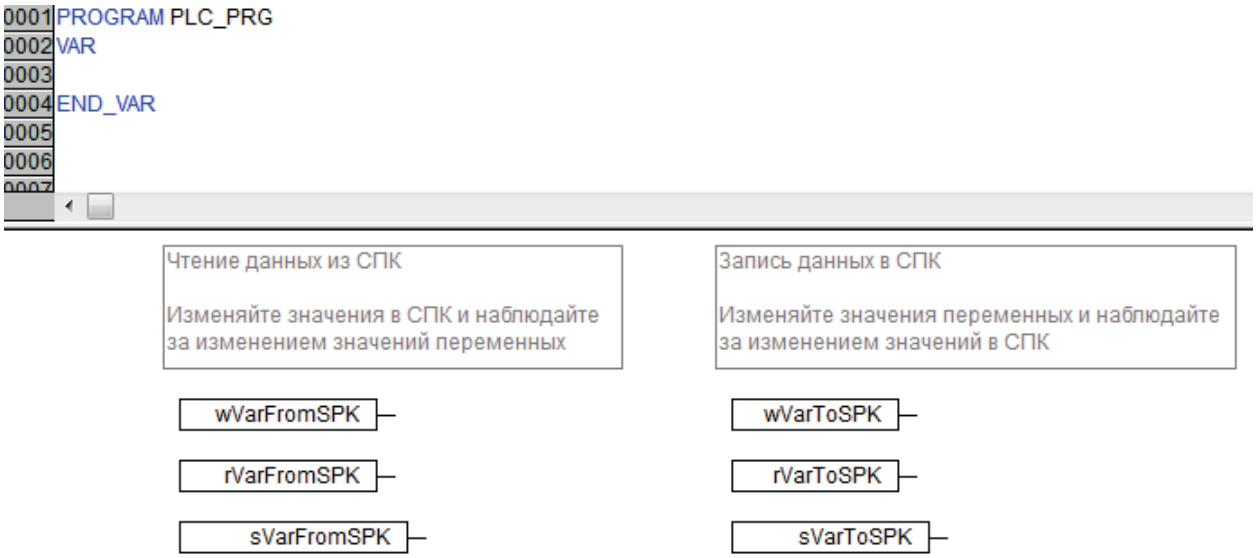


Рис. 4.28 Код программы **PLC_PRG**

На этом настройка **ПЛК (master)** завершена.

Обратите внимание, что проект не содержит каких-либо операций и используется только для отображения и ввода значений. Пользователь должен создать программу для реализации необходимым алгоритмов.

4.4. Работа с примером

Загрузите проекты в оба устройства и запустите их.

Изменяйте значения **ToSPK** переменных в ПЛК и наблюдайте соответствующие изменения в программе СПК:

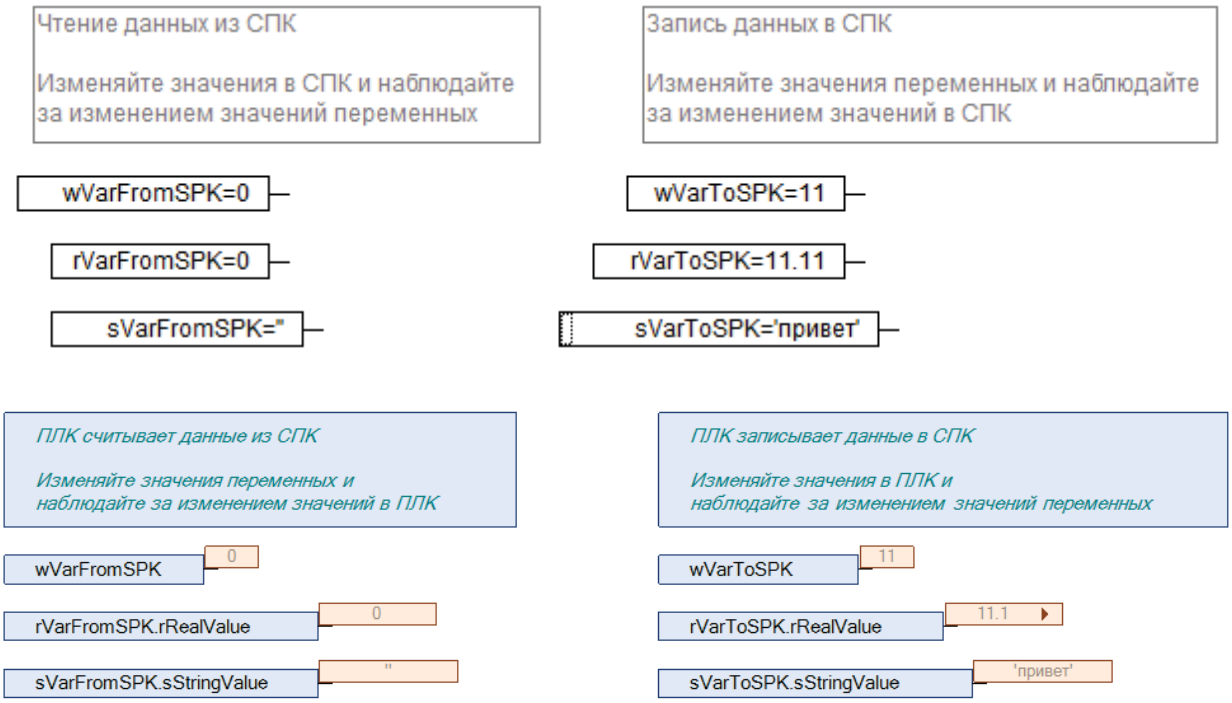


Рис. 4.29. ПЛК записывает данные в СПК

Изменяйте значения **FromSPK** переменных СПК и наблюдайте соответствующие изменения в программе ПЛК.

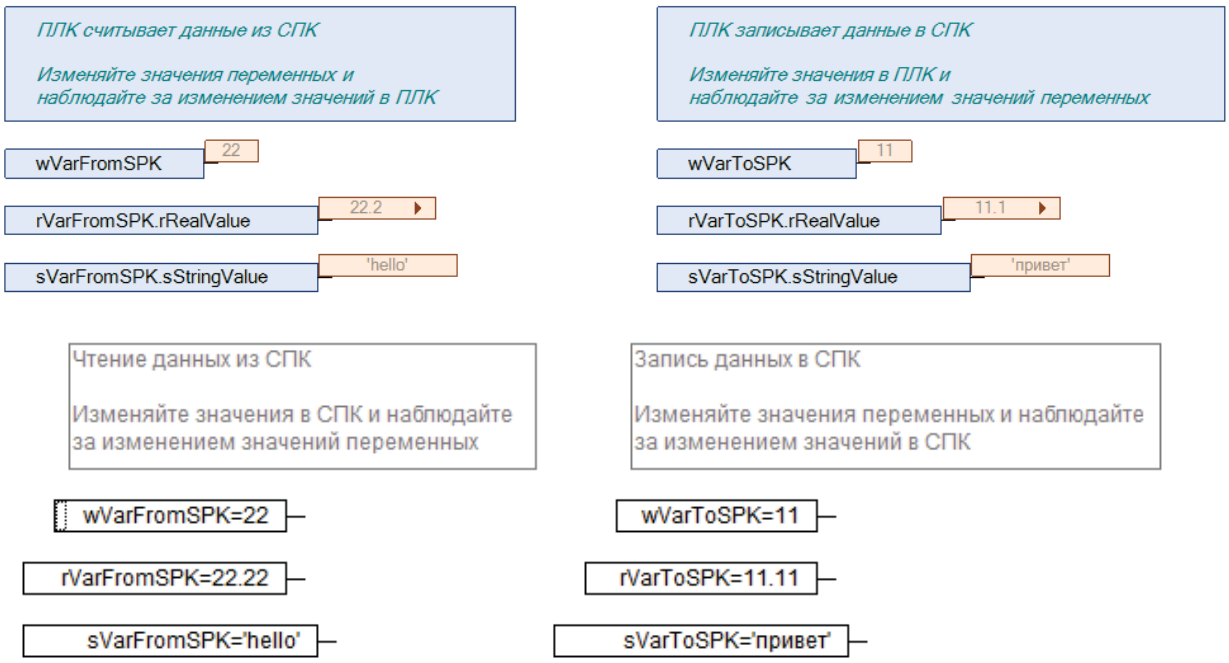


Рис. 4.30. ПЛК считывает данные из СПК

5. Modbus TCP. СПК – master, ПЛК – slave

5.1. Описание примера

Этот пример посвящен настройке обмена данными между сенсорным панельным контроллером **СПК207** и контроллером **ПЛК110 [M02]** по протоколу **Modbus TCP**. В этом примере СПК выполняет функцию **Master**, а ПЛК – **Slave**.

Основные характеристики используемых устройств приведены в табл. 5.1. Используемые в примере переменные описаны в табл. 5.2.

Табл. 5.1. Характеристики устройств

Устройство	**СПК207**	**ПЛК110 [M02]**
Функция	Master	Slave
IP адрес	10.2.11.20	10.2.11.10
Маска подсети	255.255.0.0	
Порт TCP	502	
Slave ID	-	1
Таргет	3.5.4.20 (023)	PLC110.30-M v2 (версия 3.08)
Среда разработки проекта	CODESYS 3.5 SP7 Patch4	CoDeSys 2.3.9.41
Название файла проекта	ModbusTCPmaster.projectarchive	ModbusTCPslave.pro

Табл. 5.2. Список переменных

СПК207 (Master)			ПЛК110 [M02] (Slave)	
Переменные, в которые считываются значения из Slave	**Переменные, значения которых записываются в Slave**	**Тип данных**	**Переменные ПЛК**	**Адрес регистра/бита**
xVarRead	xVarWrite	BOOL	xVar	0/0
wVarRead	wVarWrite	WORD	wVar	1
rVarRead	rVarWrite	REAL	rVar	2-3
sVarRead	sVarWrite	STRING(6)	sVar	4-6

Проекты примера доступны для скачивания: Example_SpkModbusTcpMaster.zip

5.2. Настройка ПЛК (slave)

1. Создайте новый проект **CoDeSys 2.3** для **ПЛК110** с программой **PLC_PRG** на языке **CFC**.

2. В компоненте **Конфигурация ПЛК** (вкладка **Ресурсы**) пользователь производит настройку регистров Modbus и привязывает к ним переменные.

Нажмем **ПКМ** на название контроллера (в нашем примере - **PLC110_30**) и добавим подэлемент **Modbus (Slave)**:

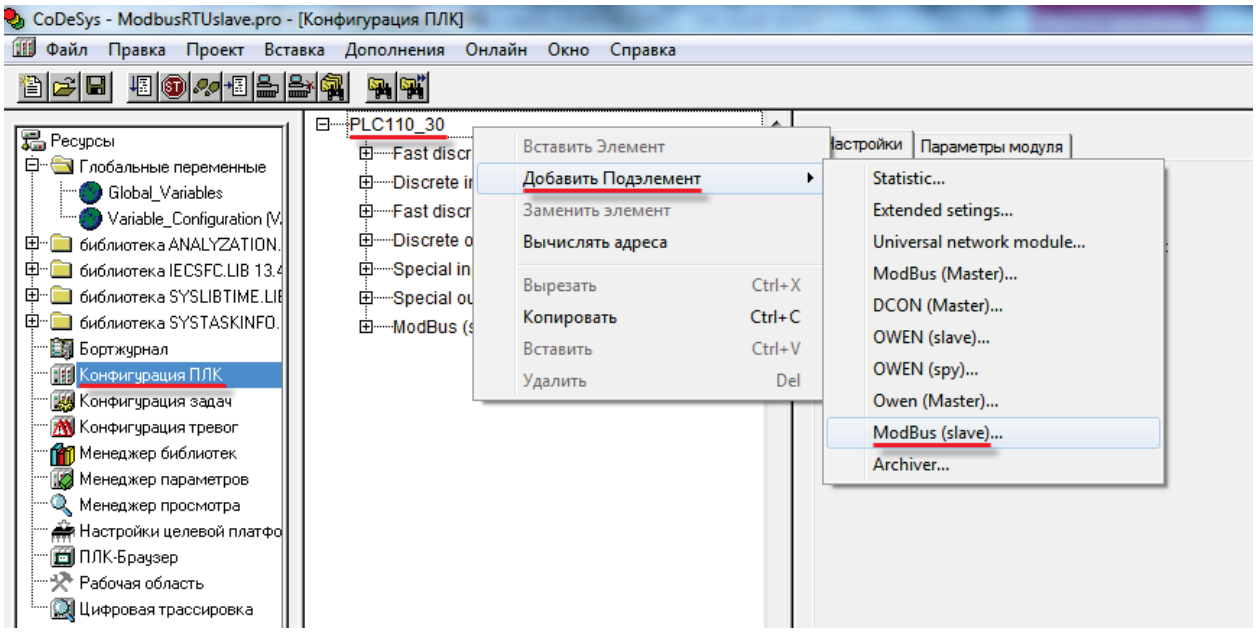


Рис. 5.1. Конфигурация ПЛК. Добавление **Modbus (Slave)**

В его настройках выберем адрес (**Slave ID**), равный **1** (в соответствии с табл. 5.1):

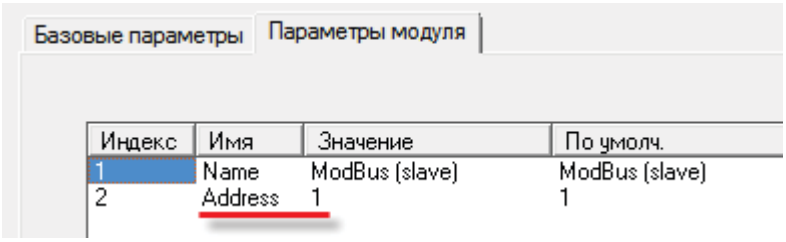


Рис. 5.2. Конфигурация ПЛК. Настройка **Modbus (Slave)**

3. Выберем порт ПЛК, который будет использоваться для связи с СПК. Для этого нажмем **ПКМ** на элемент **Modbus (FIX)** и добавим подэлемент **TCP**.

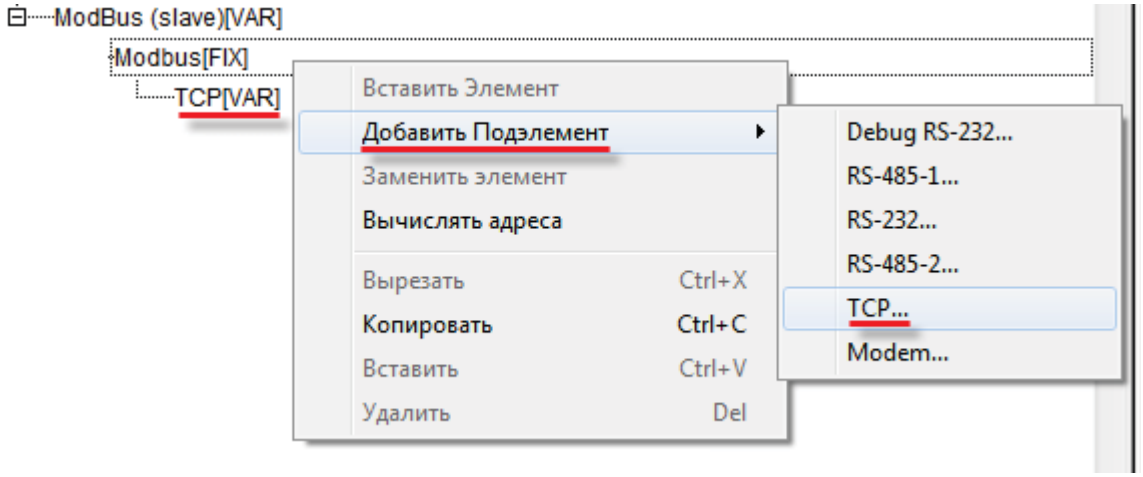


Рис. 5.3. Конфигурация ПЛК. Добавление подэлемента **TCP**

В параметрах подэлемента укажите используемый TCP порт - в соответствии с табл. 5.1 это порт **502**.

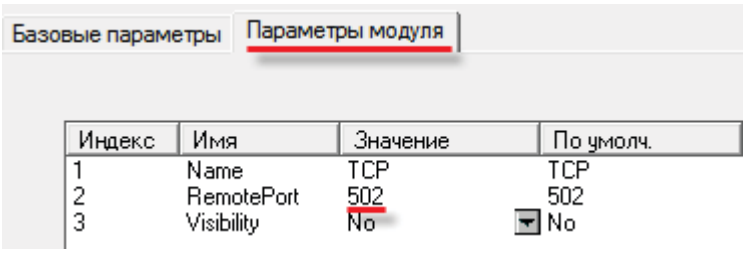


Рис. 5.4. Конфигурация ПЛК. Настройки подэлемента **TCP**

4. Нажмем **ПКМ** на элемент **Modbus (Slave)** и добавим следующие подэлементы:

- **8 bits** (для **BOOL**);
- **8 bits** (для обеспечения выравнивания памяти);
- **2 byte** (для **WORD**);
- **Float** (для **REAL**);
- 3 элемента **2 byte** (для **STRING** из 6 символов).

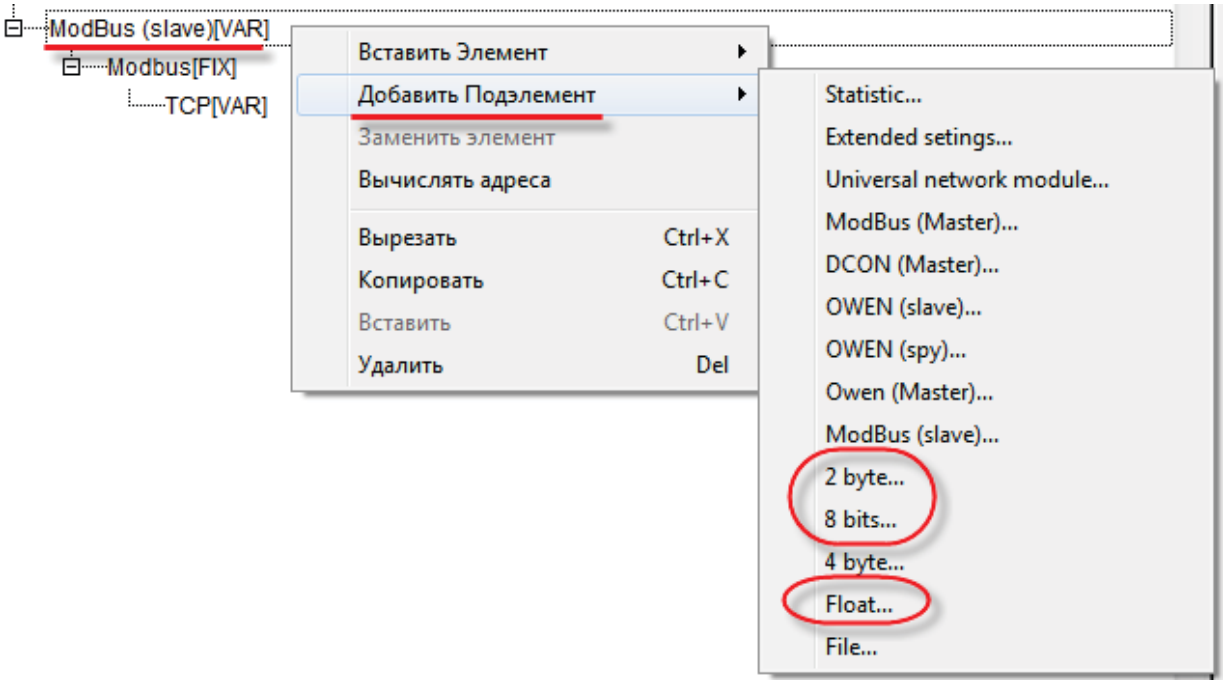


Рис. 5.5. Конфигурация ПЛК. Добавление подэлементов

В результате **Конфигурация ПЛК** будет выглядеть следующим образом (см. рис. 5.6). Объявим переменные (после ввода их имен они автоматически будут добавлены в список глобальных переменных проекта). Для ввода имени переменной два раза нажмите на **AT**. ***Обратите внимание***, что к регистрам 4-6 не привязывается никаких переменных. Соответствующая им переменная **sVar** будет объявлена в программе **PLC_PRG** (в пп. 5).

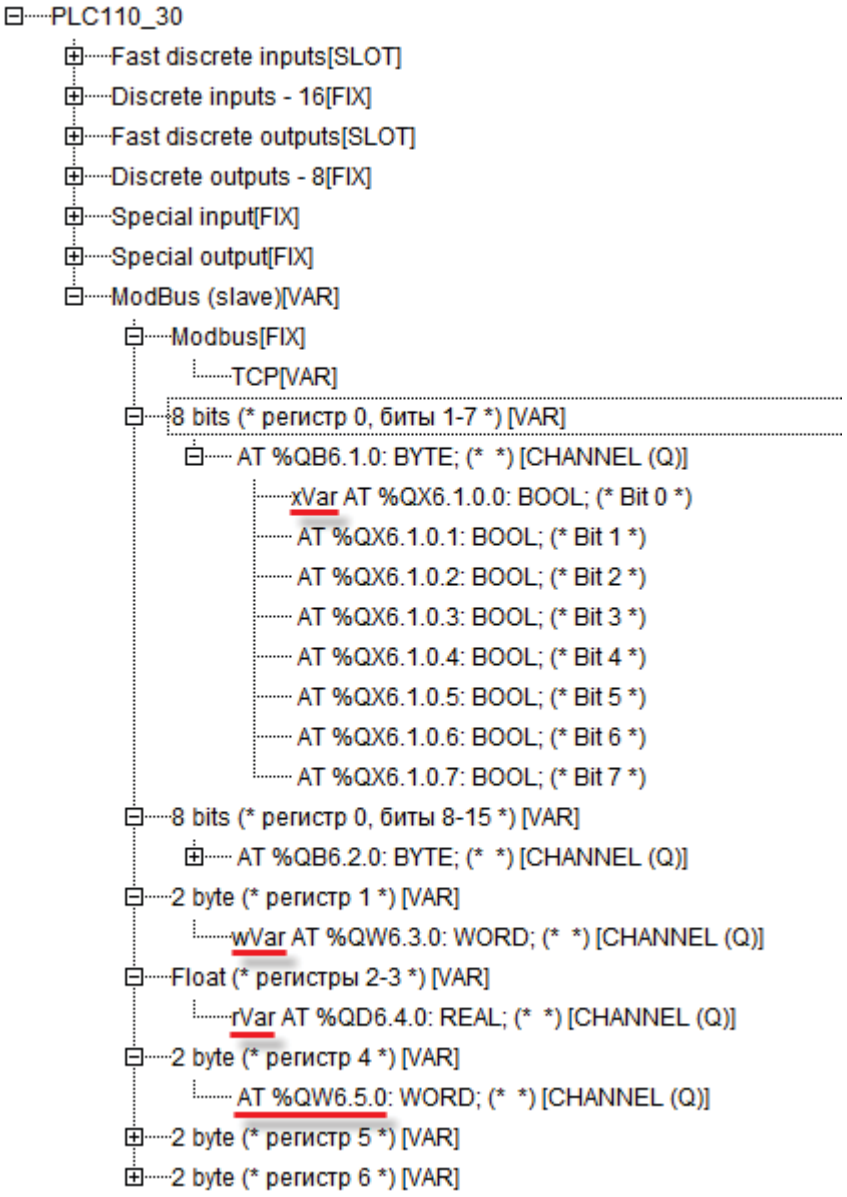


Рис. 5.6. Внешний вид **Modbus (Slave)** с добавленными подэлементами

Обратите внимание, что нумерация регистров в среде CoDeSys всегда начинается с нуля, при этом каждый регистр физически занимает два байта (16 бит). В связи с этим, переменная типа **REAL** займет два регистра (с адресами 2 и 3). Переменная типа **STRING**, которой соответствует три **2 byte** элемента, займет регистры с адресами 4-6. Это необходимо учитывать при настройке master-устройства.

Подробнее вопросы адресации рассмотрены в **Руководстве пользователя ПЛК**.

5. Программа **PLC_PRG** будет выглядеть следующим образом:

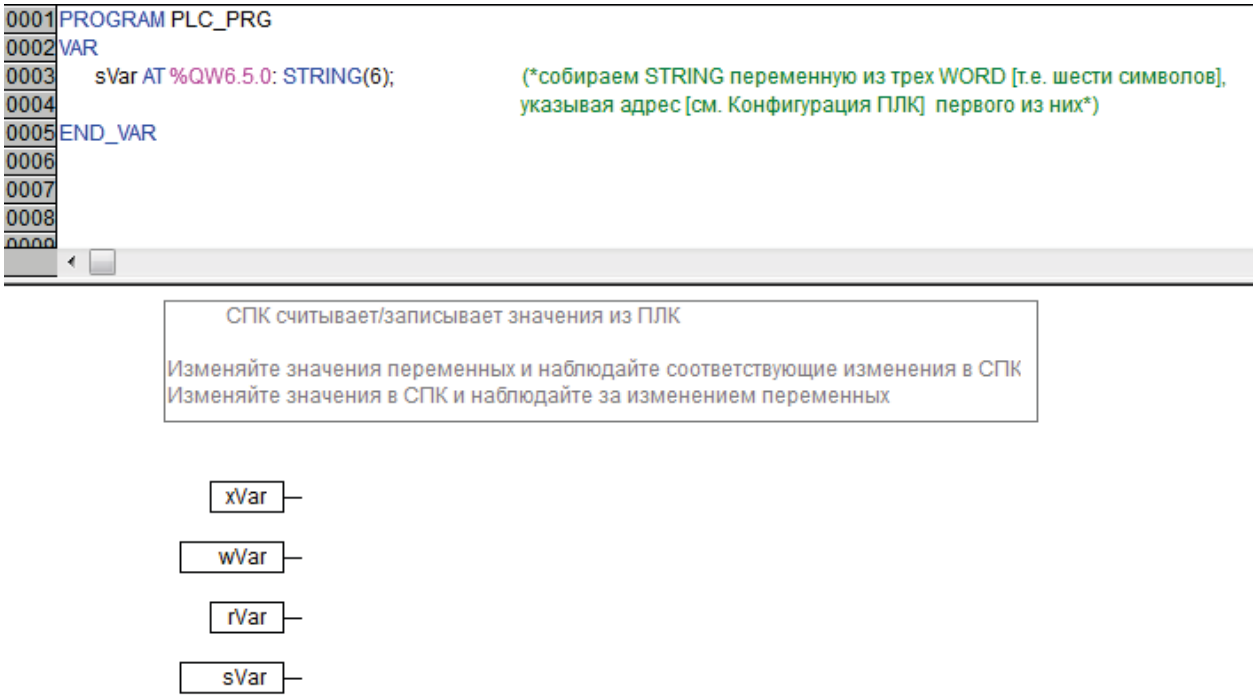


Рис. 5.7 Код программы **PLC_PRG**

На этом настройка **ПЛК (slave)** завершена

Обратите внимание, что проект не содержит каких-либо операций и используется только для отображения и ввода значений. Пользователь должен создать программу для реализации необходимым алгоритмов.

5.3. Настройка СПК (master)

1. Создайте новый проект **CODESYS 3.5** для **СПК207** с программой **PLC_PRG** на языке **CFC**.

2. Добавьте в проект объединение с именем **Real_Word**:

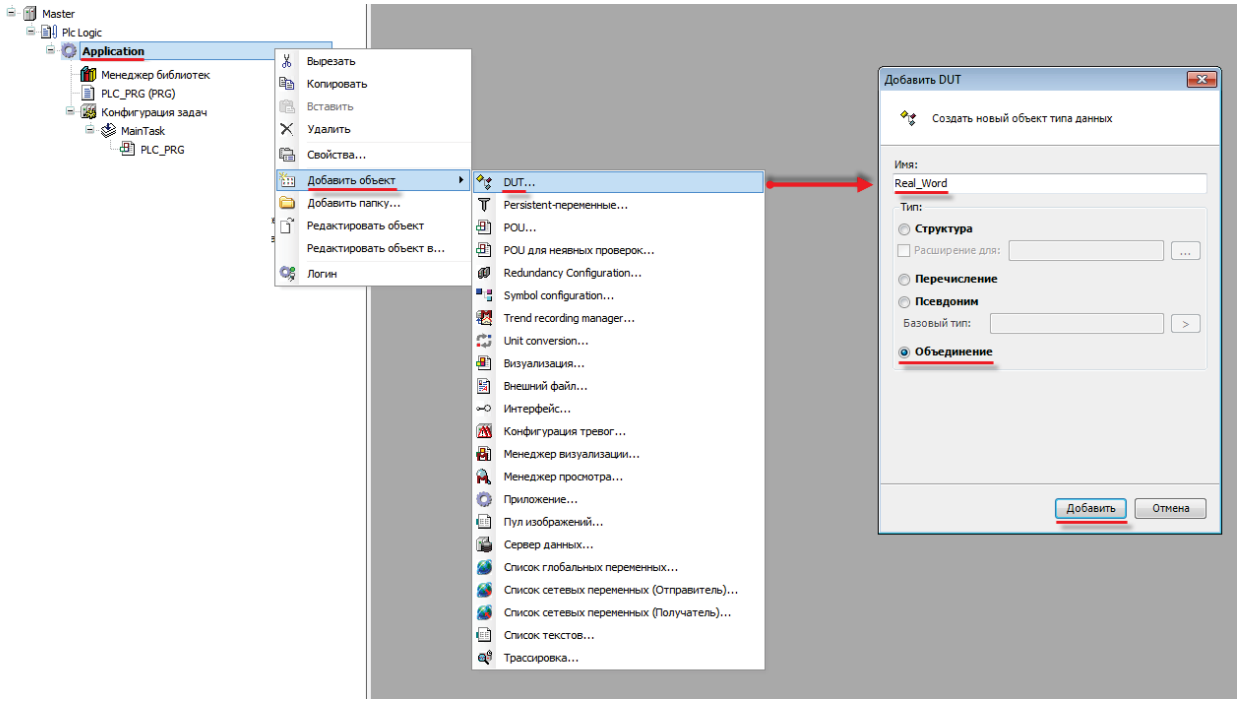


Рис. 5.8. Добавление в проект объединения

В объединении объявите переменную **rRealValue** типа **REAL** и массив **awModbusReal** типа **WORD**, содержащий два элемента:

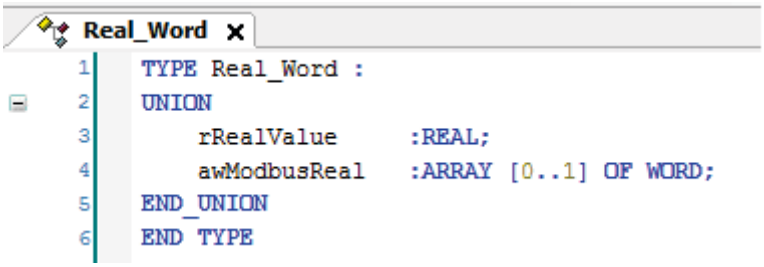


Рис. 5.9. Объявление переменных объединения

3. Добавьте в проект объединение с именем **String_Word**.

В объединении объявите переменную **sStringValue** типа **STRING** и массив **awModbusString** типа **WORD**, содержащий три элемента (**STRING** сможет содержать до 6 символов, поскольку каждый **WORD** может содержать два символа):

```
TYPE String_Word :
UNION
    awModbusString      :ARRAY [0..2] OF WORD;
    sStringValue        :STRING;
END_UNION
END_TYPE
```

Рис. 5.10. Объявление переменных объединения

4. Объявите в программе **PLC_PRG** девять переменных – 4 из них будут использоваться для отображения данных, считанных из ПЛК, еще 4 – для ввода данных, которые будут записаны в ПЛК. Последняя переменная будет являться триггером записи.

PLC_PRG

```
PROGRAM PLC_PRG
VAR
    (*данные, из считываемые ПЛК*)
    xVarRead:       BOOL;
    wVarRead:       WORD;
    rVarRead:       Real_Word;
    sVarRead:       String_Word;

    (*данные, записываемые в ПЛК*)
    xVarWrite:      BOOL;
    wVarWrite:      WORD;
    rVarWrite:      Real_Word;
    sVarWrite:      String_Word;

    xTrigger:       BOOL; // триггер записи
END_VAR
```

Рис. 5.11. Объявление переменных программы

5. Код программы будет выглядеть следующим образом:

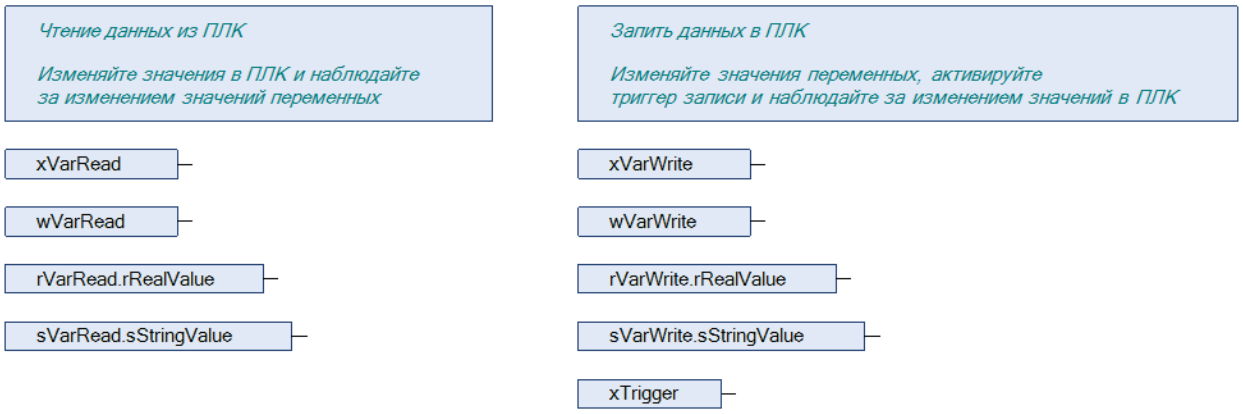


Рис. 5.12. Код программы на языке **CFC**

6. Добавьте в проект компонент **Ethernet**. ***Обратите внимание***, что версия компонента не должна превышать версию таргет-файла СПК. Подробнее см. в документе **СПК. Modbus**.

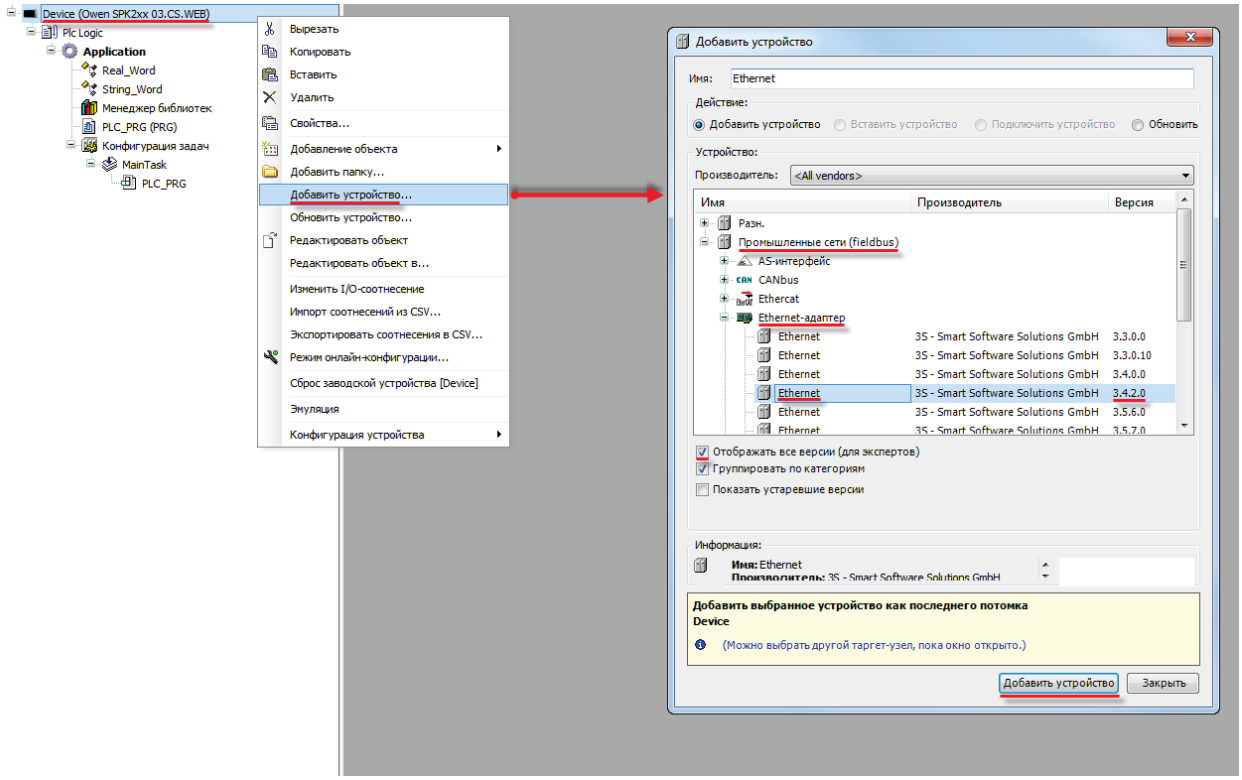


Рис. 5.13. Добавление компонента **Ethernet**

В конфигурации Ethernet укажите сетевые настройки в соответствии с табл. 5.1.

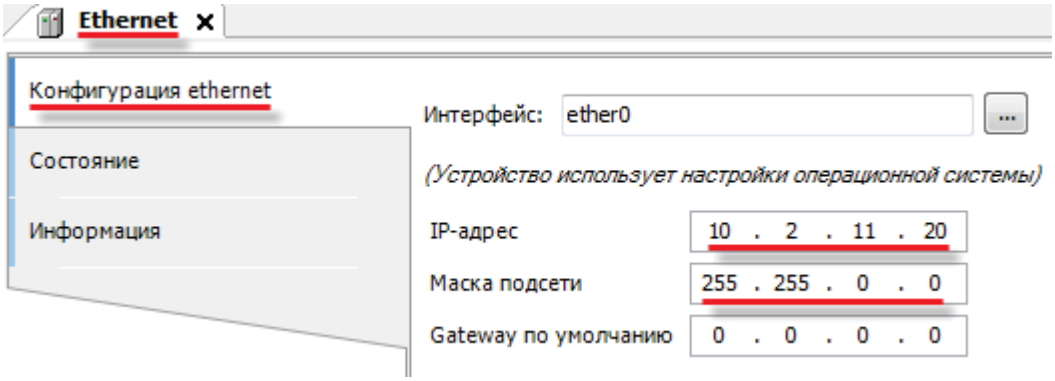


Рис. 5.14. Настройки компонента **Ethernet**

7. В компонент **Ethernet** добавьте компонент **Modbus TCP Master**. ***Обратите внимание***, что версия компонента не должна превышать версию таргет-файла СПК. Подробнее см. в документе **СПК. Modbus**.

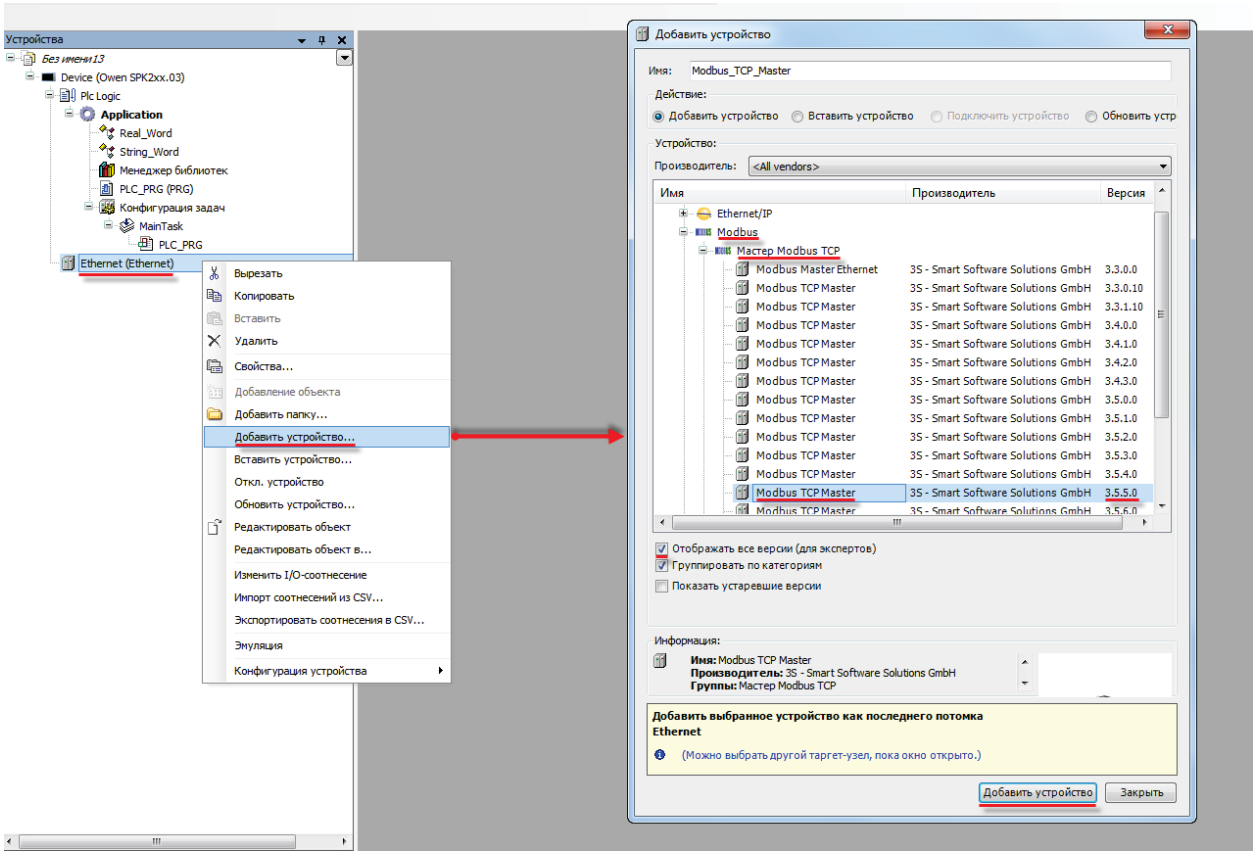


Рис. 5.15. Добавление компонента **Modbus TCP Master**

В настройках компонента поставьте галочку **Автоподключение**.

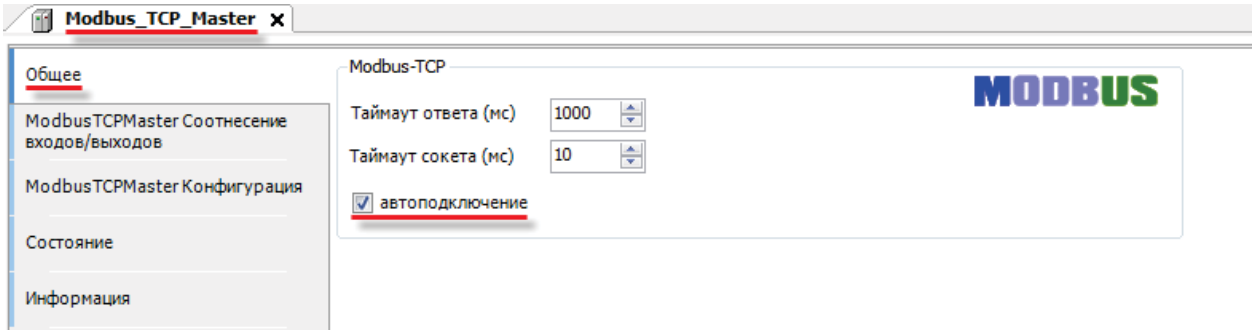


Рис. 5.16. Настройка компонентов **Modbus TCP Master**

8. В **Modbus TCP Master** добавьте компонент **Modbus TCP Slave**. ***Обратите внимание***, что версия компонента не должна превышать версию таргет-файла СПК. Подробнее см. в документе **СПК. Modbus**.

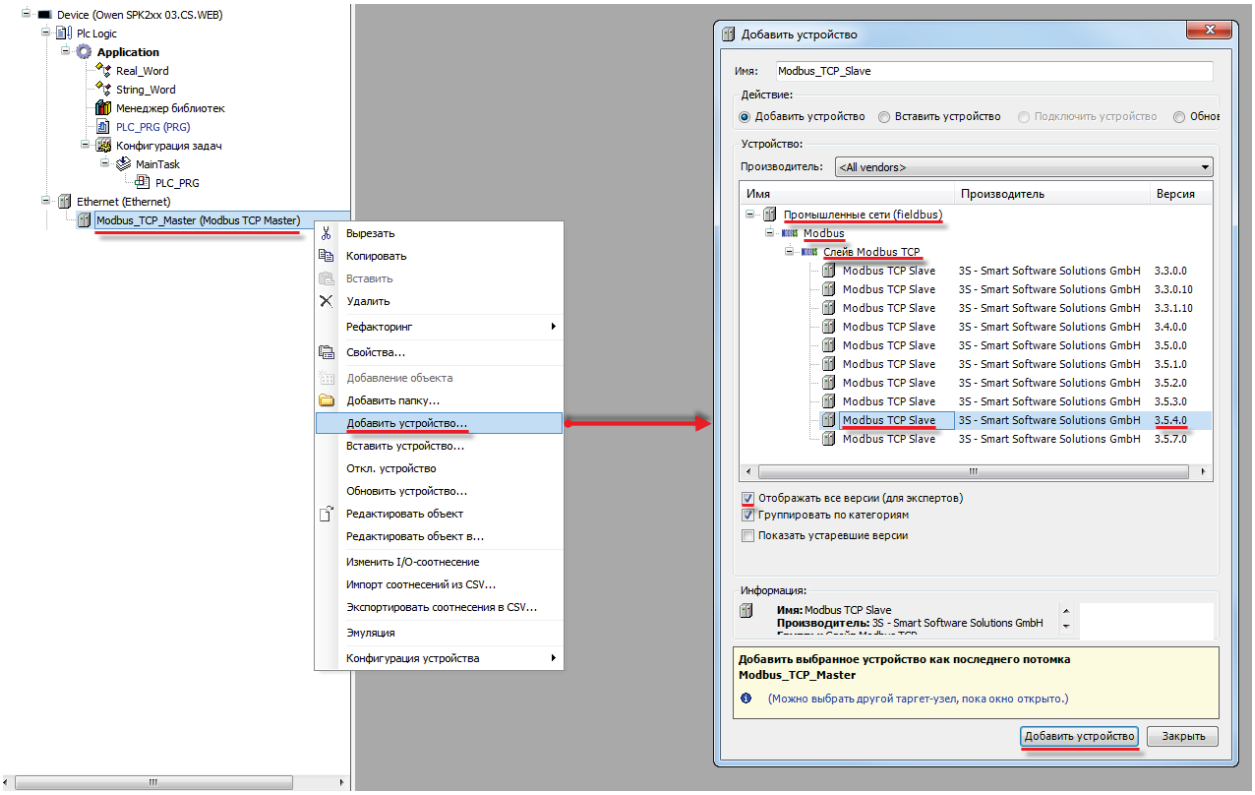


Рис. 5.17. Добавление компонента **Modbus TCP Slave** в проект

В настройках компонента на вкладке **Общее** укажите IP-адрес, Unit ID и порт slave-устройства в соответствии с табл. 5.1.

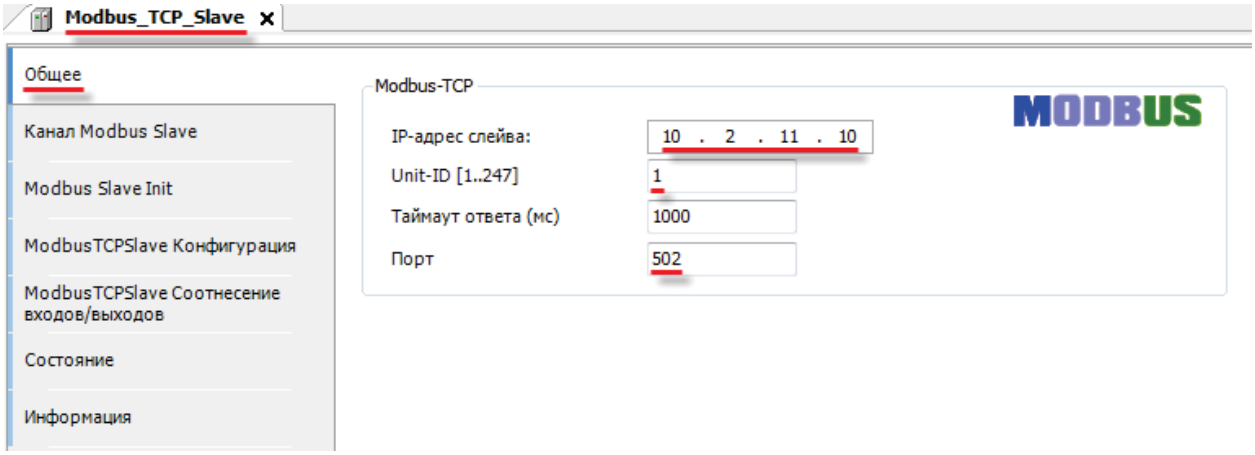
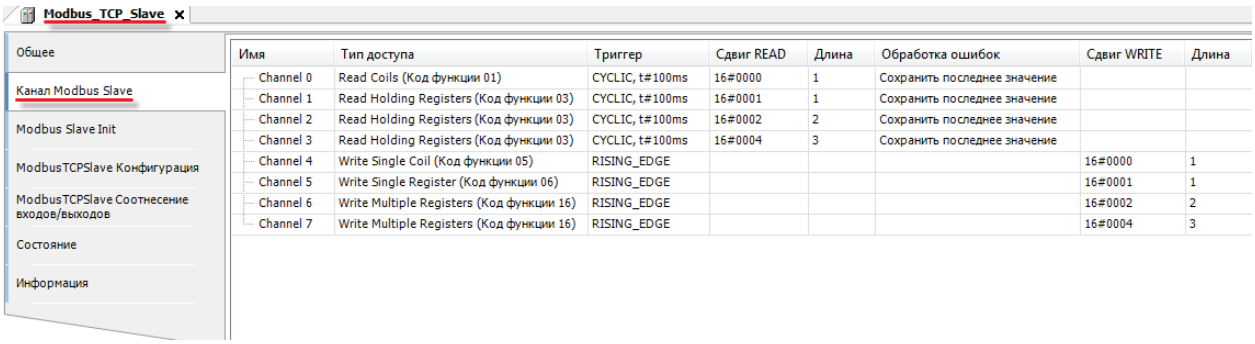


Рис. 5.18. Настройки компонента **Modbus Slave** в проект

На вкладке **Канал Modbus Slave** создайте 8 каналов – 4 из них будут использоваться для чтения переменных, 4 – для записи. Чтение будет осуществляться циклически, запись – по переднему фронту триггера (**RISING_EDGE**). Используемые функции соответствуют типам данных, адреса регистров настроены согласно табл. 5.2.



Имя	Тип доступа	Триггер	Сдвиг READ	Длина	Обработка ошибок	Сдвиг WRITE	Длина
Channel 0	Read Coils (Код функции 01)	CYCLIC, t#100ms	16#0000	1	Сохранить последнее значение		
Channel 1	Read Holding Registers (Код функции 03)	CYCLIC, t#100ms	16#0001	1	Сохранить последнее значение		
Channel 2	Read Holding Registers (Код функции 03)	CYCLIC, t#100ms	16#0002	2	Сохранить последнее значение		
Channel 3	Read Holding Registers (Код функции 03)	CYCLIC, t#100ms	16#0004	3	Сохранить последнее значение		
Channel 4	Write Single Coil (Код функции 05)	RISING_EDGE				16#0000	1
Channel 5	Write Single Register (Код функции 06)	RISING_EDGE				16#0001	1
Channel 6	Write Multiple Registers (Код функции 16)	RISING_EDGE				16#0002	2
Channel 7	Write Multiple Registers (Код функции 16)	RISING_EDGE				16#0004	3

Рис. 5.19. Настройка каналов **Modbus Slave**

На вкладке **ModbusTCPSlave Соотнесение входов/выходов** привяжите к каналам переменные программы в соответствии с табл. 5.2. Не забудьте у параметра **Всегда обновлять переменные** выставить значение **Включено 2**.

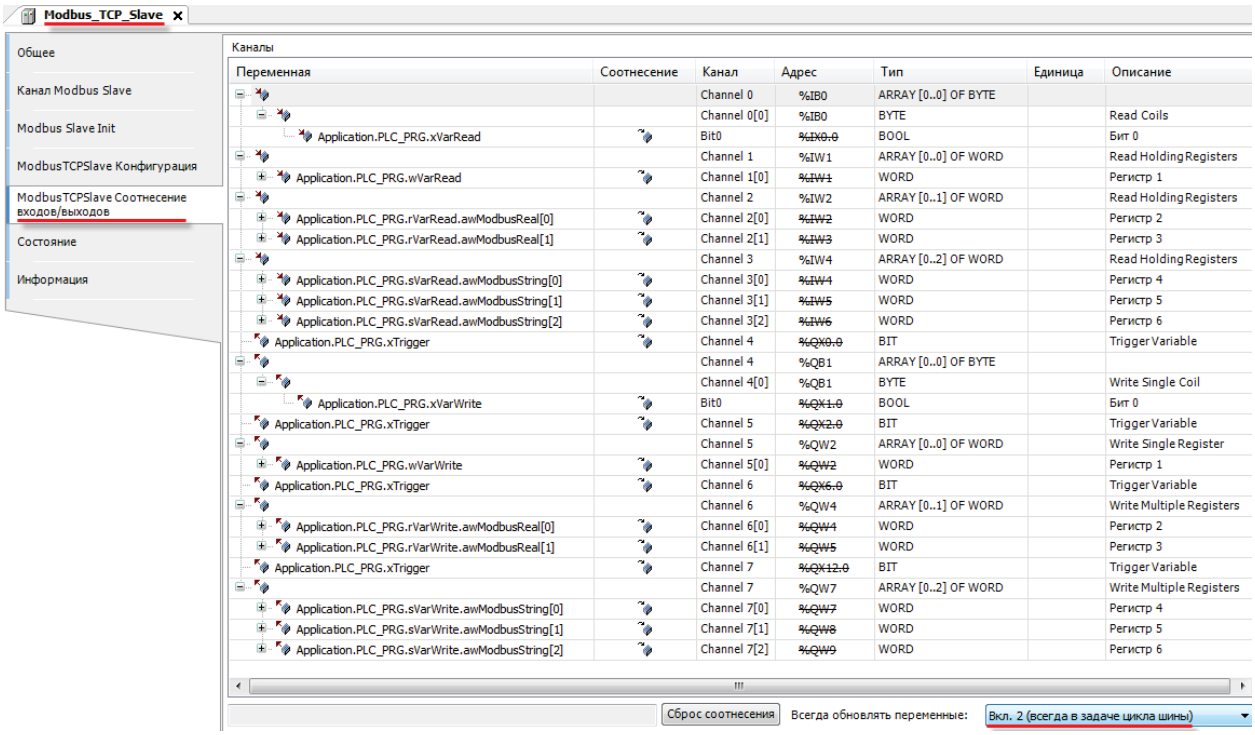


Рис. 5.20. Привязка переменных к каналу

На этом настройка **СПК (master)** завершена.

Обратите внимание, что проект не содержит каких-либо операций и используется только для отображения и ввода значений. Пользователь должен создать программу для реализации необходимым алгоритмов.

5.4. Работа с примером

Загрузите проекты в оба устройства и запустите их.

Изменяйте значения переменных в ПЛК и и наблюдайте соответствующие изменения в программе СПК:

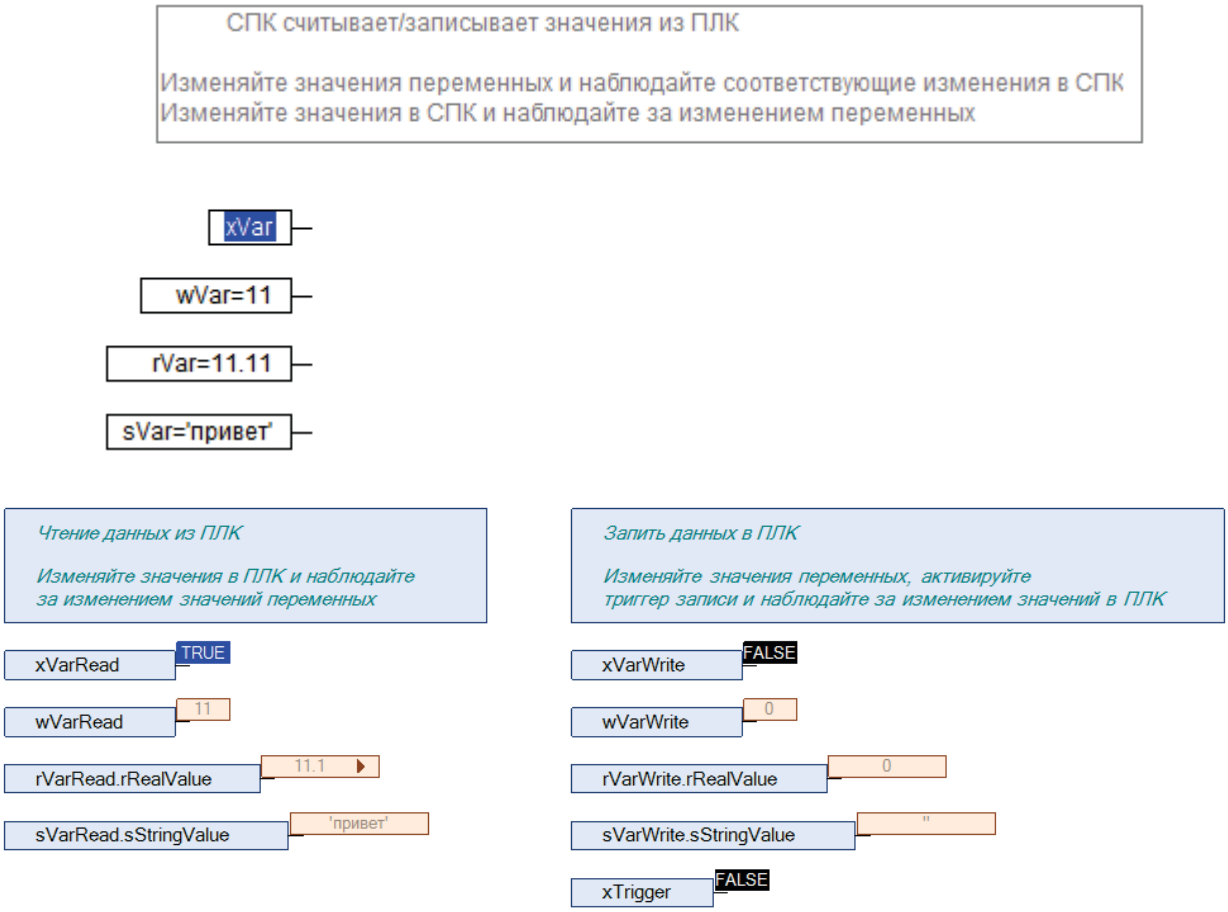


Рис. 5.21. СПК считывает данные из ПЛК

Измените значения **write** переменных СПК и активируйте триггер записи. Наблюдайте соответствующие изменения в программе ПЛК. Также новые значения будут считаны в **read** переменные программы СПК.

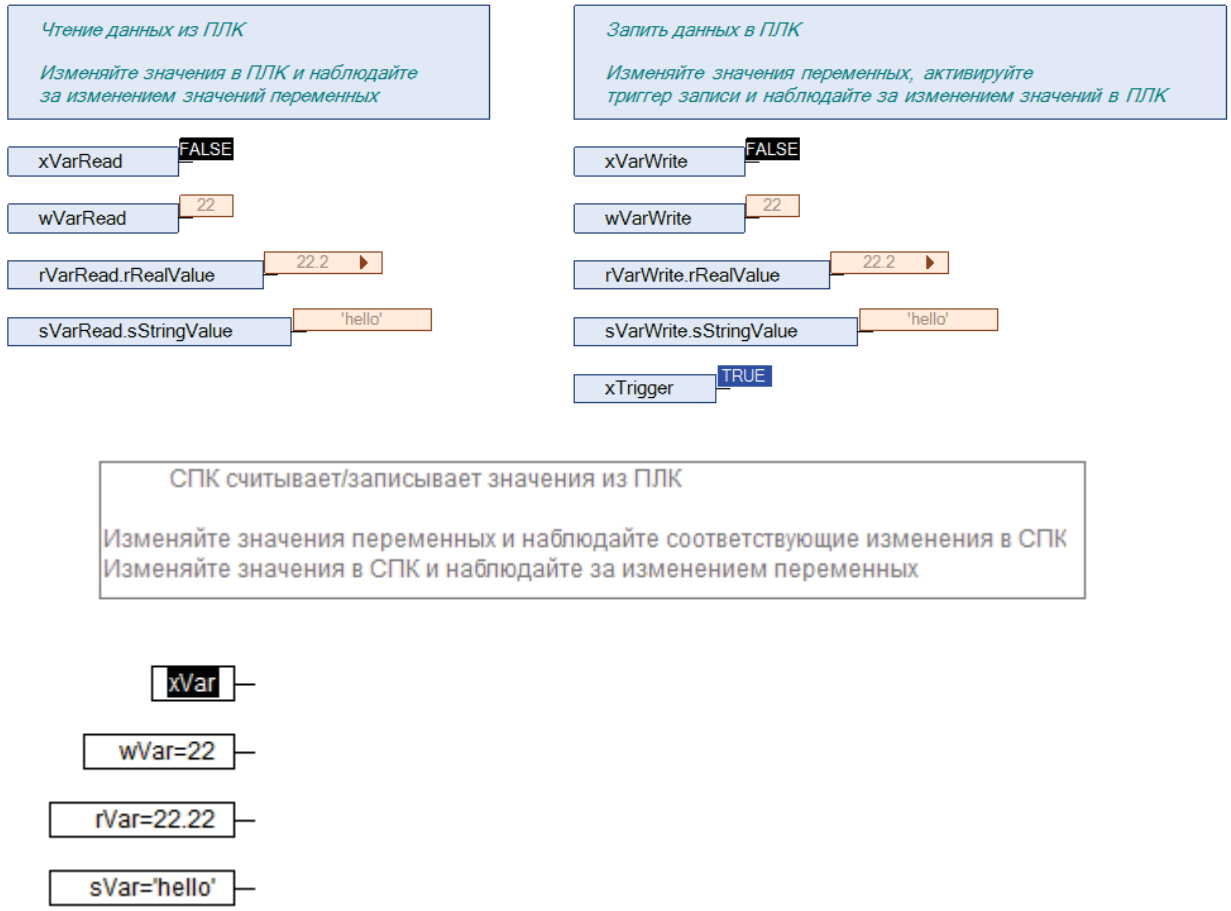


Рис. 5.22. СПК записывает данные в ПЛК

6. Modbus TCP. СПК – slave, ПЛК – master

6.1. Описание примера

Этот пример посвящен настройке обмена данными между сенсорным панельным контроллером **СПК207** и контроллером **ПЛК110 [M02]** по протоколу **Modbus TCP**. В этом примере СПК выполняет функцию **Slave**, а ПЛК – **Master**.

Основные характеристики используемых устройств приведены в табл. 6.1. Используемые в примере переменные описаны в табл. 6.2.

Табл. 6.1. Характеристики устройств

Устройство	**СПК207**	**ПЛК110 [M02]**
Функция	Slave	Master
IP адрес	10.2.11.20	10.2.11.20
Маска подсети	255.255.0.0	
Порт TCP	502	
Slave ID	1	-
Таргет	3.5.4.20 (023)	PLC110.30-M v2 (версия 3.08)
Среда разработки проекта	CODESYS 3.5 SP7 Patch4	CoDeSys 2.3.9.41
Название файла проекта	ModbusTCPslave.projectarchive	ModbusTCPmaster.pro

Табл. 6.2. Список переменных

СПК207 (Slave)			ПЛК110 [M02] (Master)
Переменные, которые считывает Master	**Адрес input регистра**	**Тип данных**	**Переменные ПЛК**
wVarFromSPK	0	WORD	wVarFromSPK
rVarFromSPK	1-2	REAL	rVarFromSPK
sVarFromSPK	3-5	STRING(6)	sVarFromSPK
Переменные, которые записывает Master	**Адрес holding регистра**	**Тип данных**	**Переменные ПЛК**
wVarToSPK	0	WORD	wVarToSPK
rVarToSPK	1-2	REAL	rVarToSPK
sVarToSPK	3-5	STRING(6)	sVarToSPK

Проекты примера доступны для скачивания: Example_SpkModbusTcpSlave.zip

6.2. Настройка СПК (slave)

1. Создайте новый проект **CODESYS 3.5** для **СПК207** с программой **PLC_PRG** на языке **CFC**.

2. Добавьте в проект объединение с именем **Real_Word**:

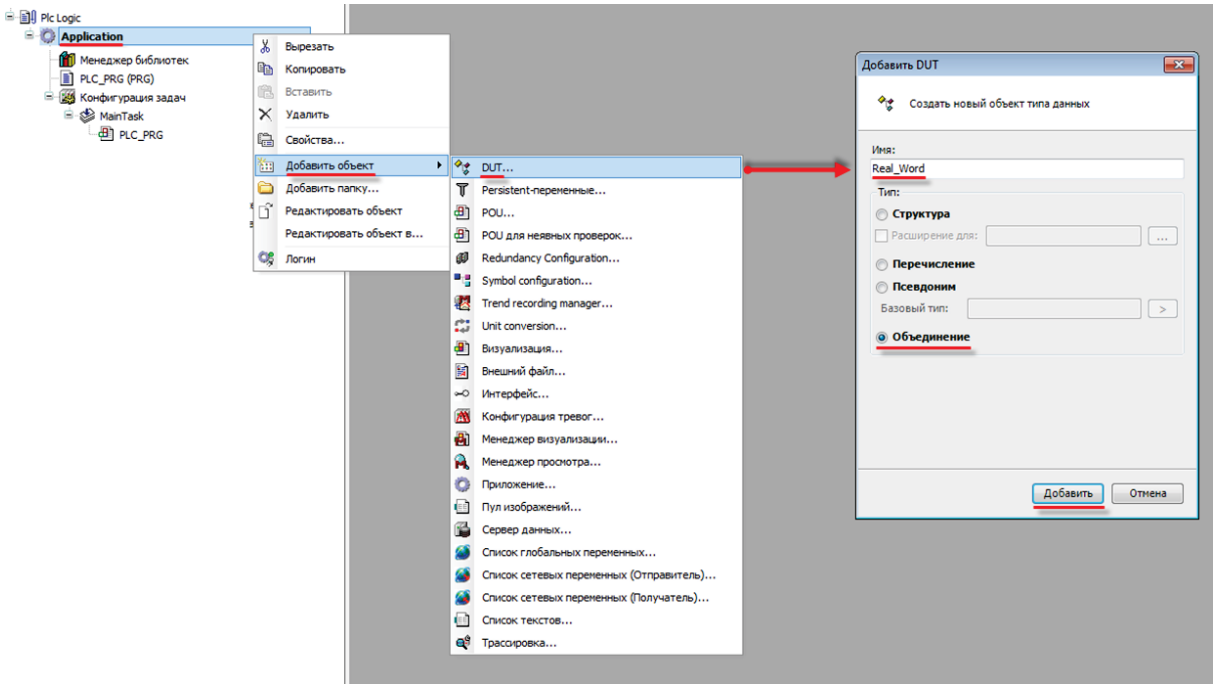


Рис. 6.1. Добавление в проект объединения

В объединении объявите переменную **rRealValue** типа **REAL** и массив **awModbusReal** типа **WORD**, содержащий два элемента:

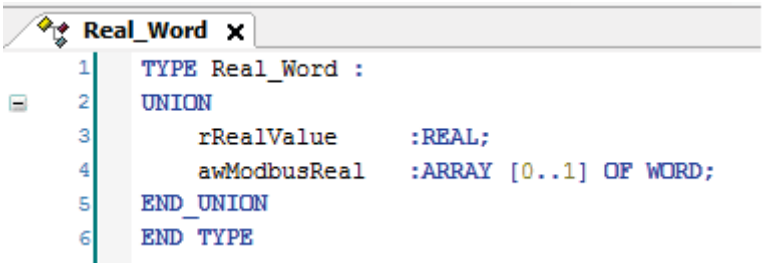


Рис. 6.2. Объявление переменных объединения

3. Добавьте в проект объединение с именем **String_Word**.

В объединении объявите переменную **sStringValue** типа **STRING** и массив **awModbusString** типа **WORD**, содержащий три элемента (**STRING** сможет содержать до 6 символов, поскольку каждый **WORD** может содержать два символа):

```
TYPE String_Word :
UNION
    awModbusString      :ARRAY [0..2] OF WORD;
    sStringValue        :STRING;
END_UNION
END_TYPE
```

Рис. 6.3. Объявление переменных объединения

4. Объявите в программе **PLC_PRG** 6 переменных – 3 из них будут считываться ПЛК, 3 – записываться ПЛК. ***Обратите внимание*** на п. 2.3.

PLC_PRG

```
PROGRAM PLC_PRG
VAR
    (*данные, считываемые ПЛК*)
    wVarFromSPK:    WORD;
    rVarFromSPK:    Real_Word;
    sVarFromSPK:    String_Word;

    (*данные, записываемые ПЛК*)
    wVarToSPK:      WORD;
    rVarToSPK:      Real_Word;
    sVarToSPK:      String_Word;
END_VAR
```

Рис. 6.4. Объявление переменных программы

5. Код программы будет выглядеть следующим образом:

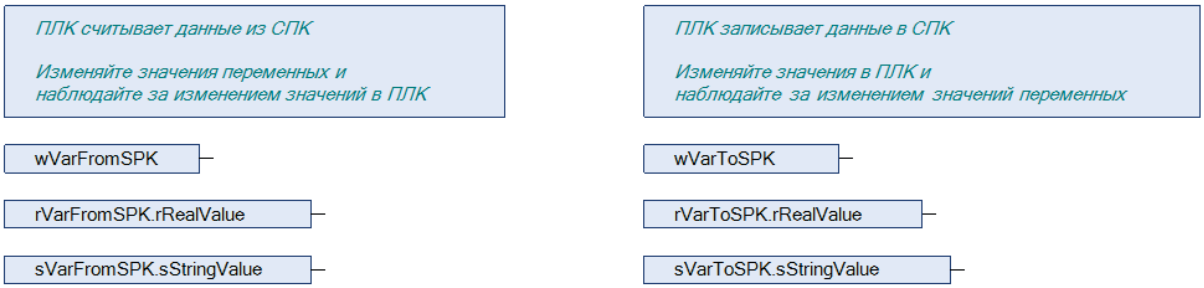


Рис. 6.5. Код программы на языке **CFC**

6. Добавьте в проект компонент **Ethernet**. ***Обратите внимание***, что версия компонента не должна превышать версию таргет-файла СПК. Подробнее см. в документе **СПК. Modbus**.

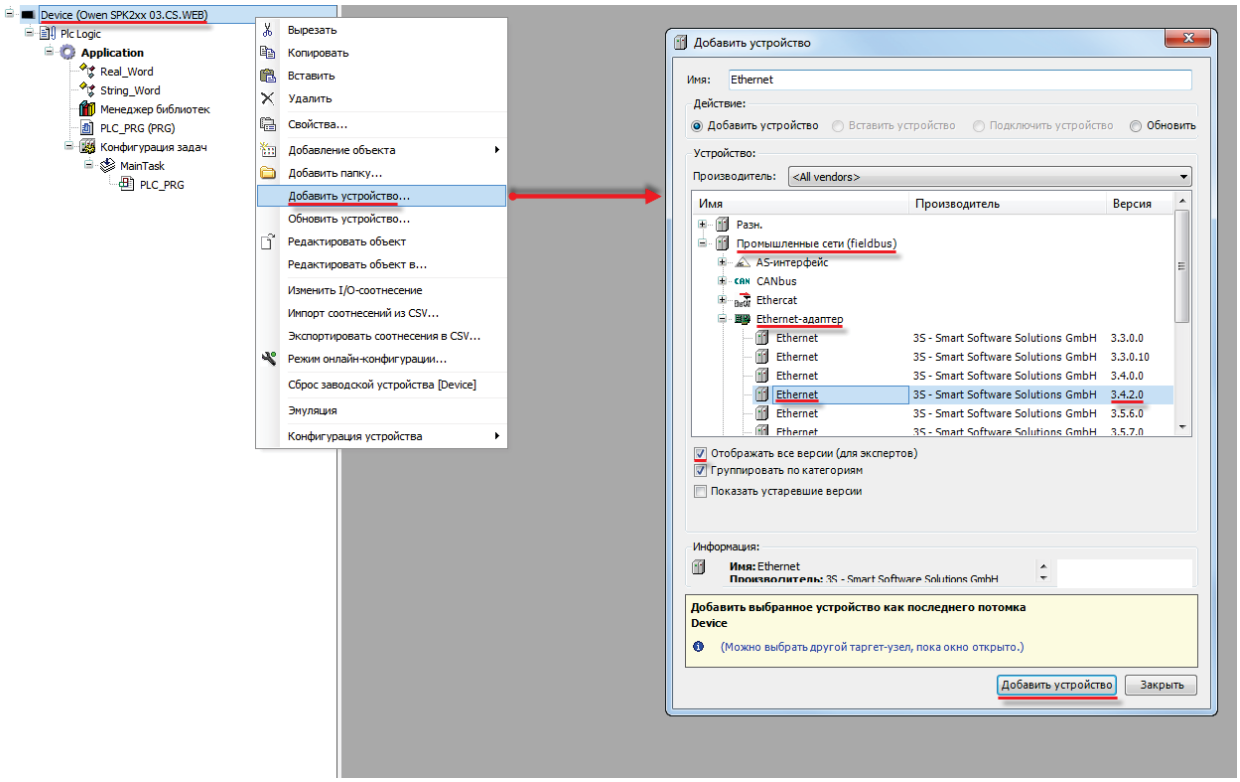


Рис. 6.6. Добавление компонента **Ethernet**

В конфигурации Ethernet укажите сетевые настройки в соответствии с табл. 6.1.

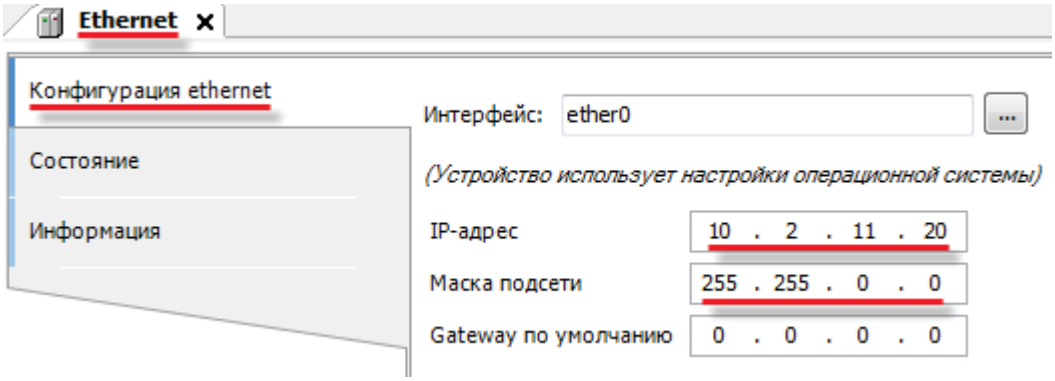


Рис. 6.7. Настройки компонента **Ethernet**

7. В компонент **Ethernet** добавьте компонент **Modbus TCP Slave Device**. ***Обратите внимание***, что версия компонента не должна превышать версию таргет-файла СПК. Подробнее см. в документе **СПК. Modbus**.

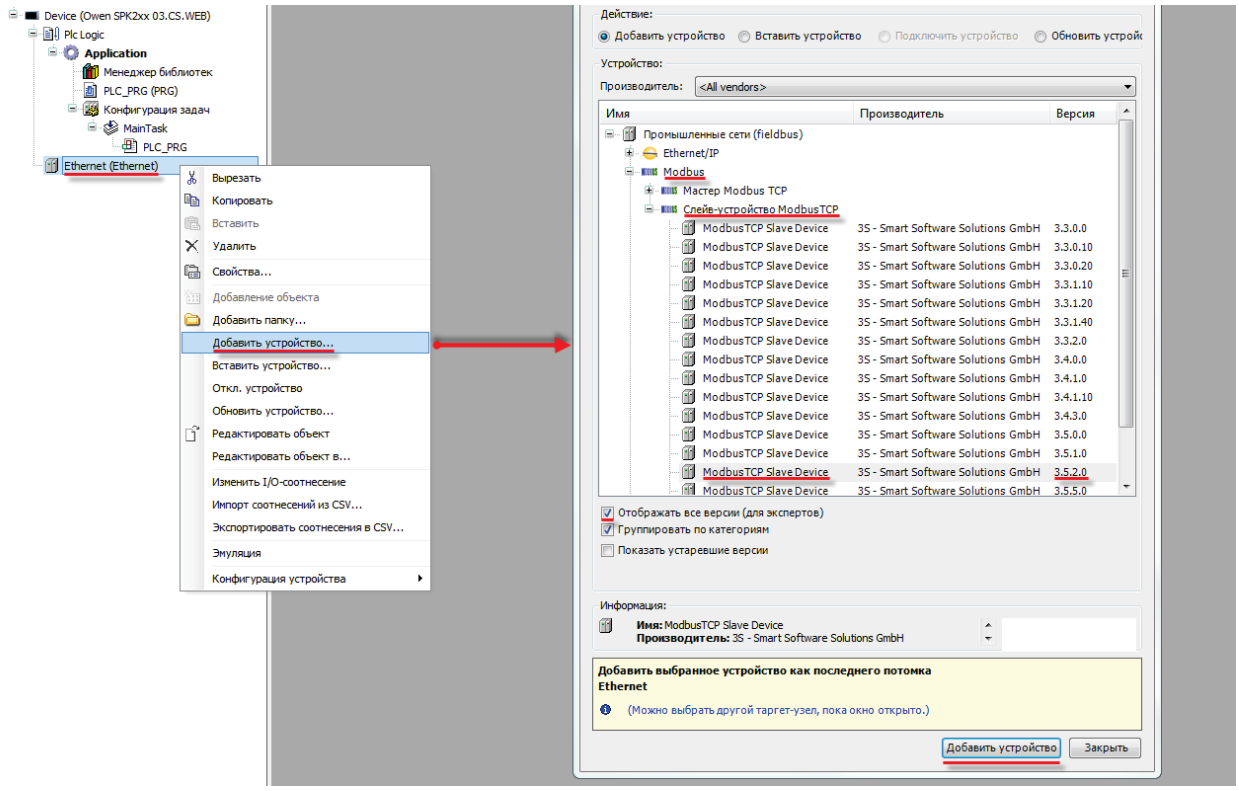


Рис. 6.8. Добавление компонента **Modbus TCP Slave Device**

В настройках компонента на вкладке **Страницу конфигурации** укажите порт и адрес slave-устройства (**502** и **1** в соответствии с табл. 6.1).

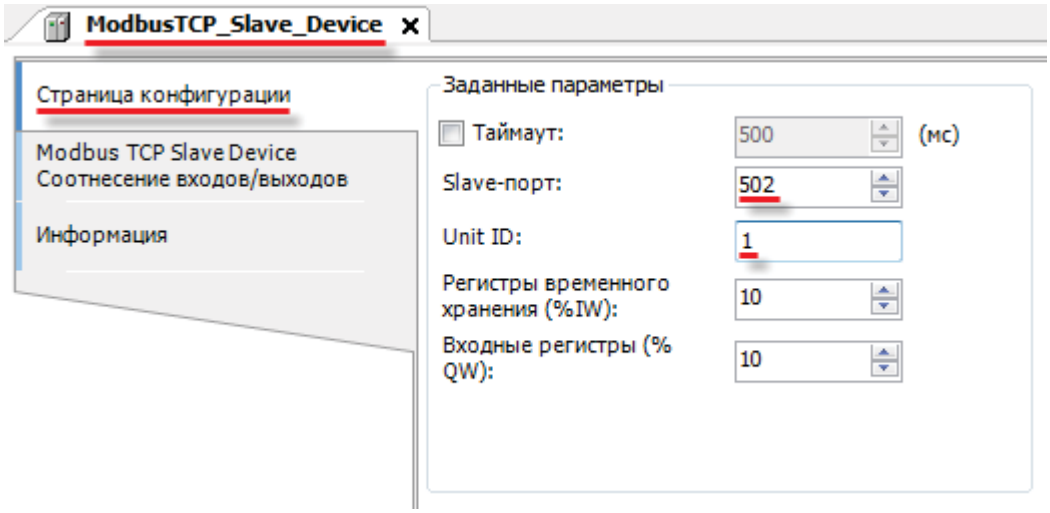


Рис. 6.9. Настройки компонента **Modbus TCP Slave Device**

На вкладке **Modbus Serial Device Соотнесение входов/выходов** привяжите к регистрам переменные программы в соответствии с табл. 6.2. ***Обратите внимание*** на порядок **WORD** для переменных типа **REAL**.

Обратите внимание, что канал **Inputs** содержит **Holding регистры**, а канал **Outputs** – **Input регистры**.

Не забудьте у параметра **Всегда обновлять переменные** выставить значение **Включено 2**.

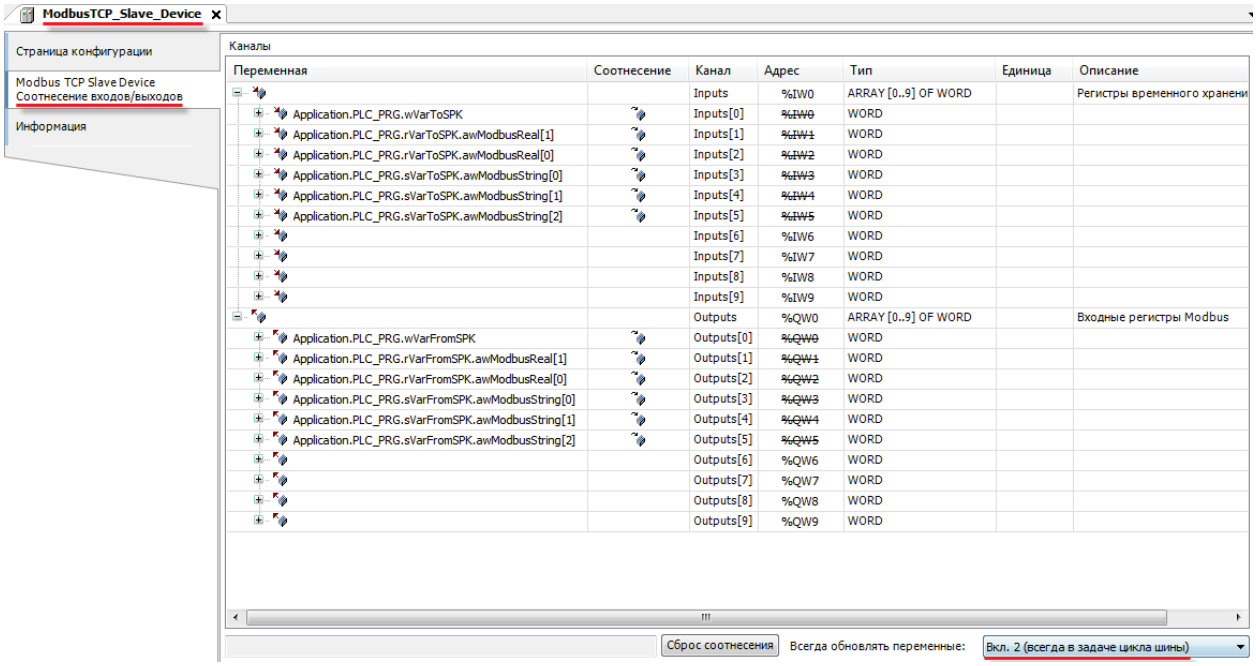


Рис. 6.10. Привязка переменных к регистрам slave-устройства

На этом настройка **СПК (slave)** завершена.

6.3. Настройка ПЛК (master)

1. Создайте новый проект **CoDeSys 2.3** для **ПЛК110** с программой **PLC_PRG** на языке **CFC**.

Нажмем **ПКМ** на название контроллера (в нашем примере - **PLC110_30**) и добавим подэлемент **Modbus (Master)**:

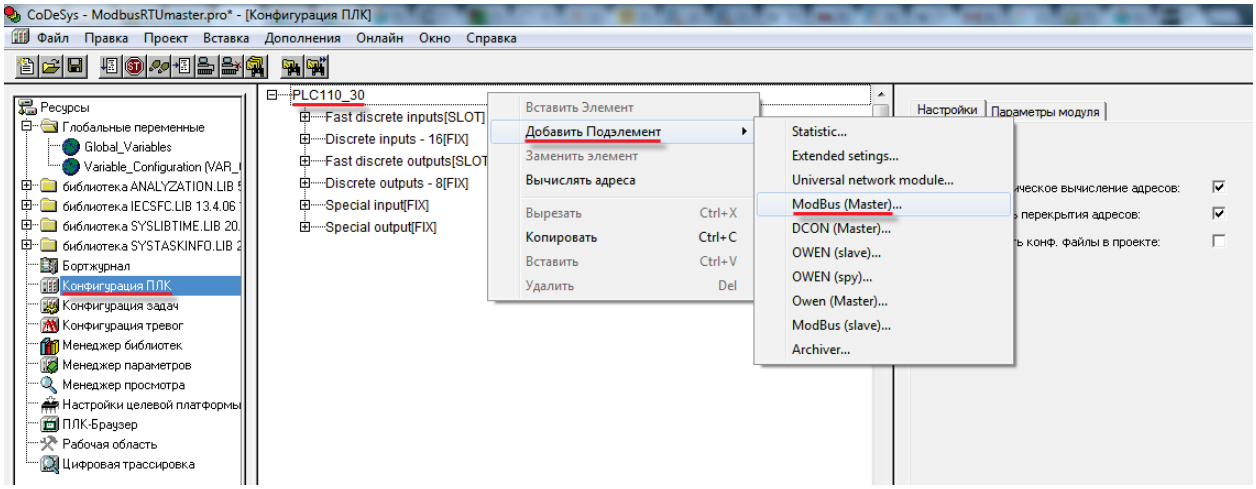


Рис. 6.11. Конфигурация ПЛК. Добавление **Modbus (Master)**

Этот элемент не нуждается в настройках.

2. Нажмем **ПКМ** на элемент **Modbus (Master)** и добавим два подэлемента **Universal Modbus Device**:

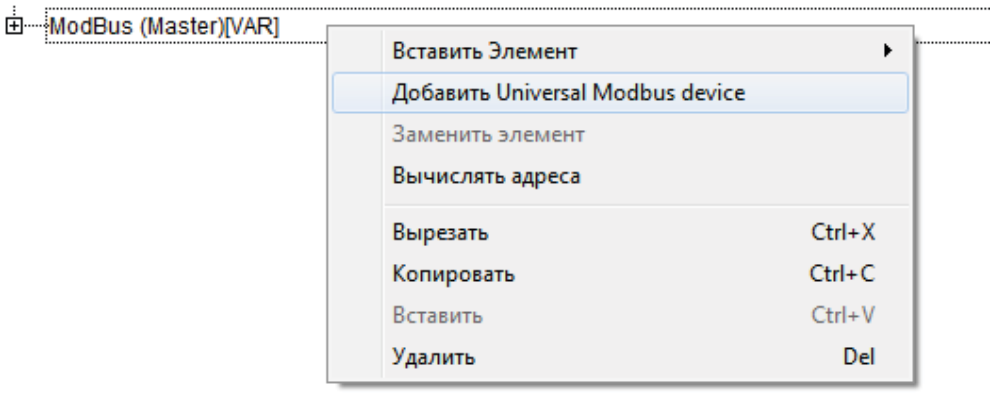


Рис. 6.12. Конфигурация ПЛК. Добавление **Universal Modbus Device**

Один из них будет использоваться для чтения значений из СПК, второй – для записи.

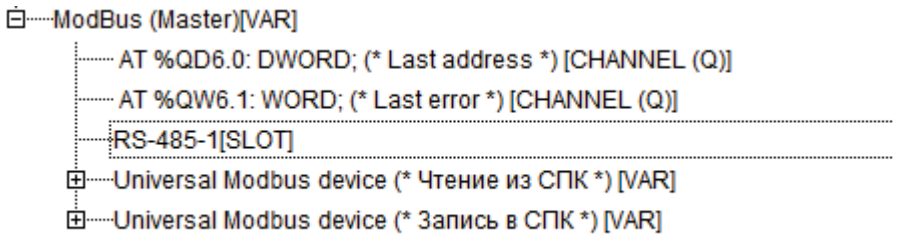


Рис. 6.13. Внешний вид **Конфигурации ПЛК** после добавления двух **Universal Modbus Device**

В настройках элемента **Чтение из СПК** укажем тип связи (**TCP**), IP-адрес, порт и **Slave ID** СПК (в соответствии с табл. 6.1) и режим опроса (**By poll time**, т.е. циклически).

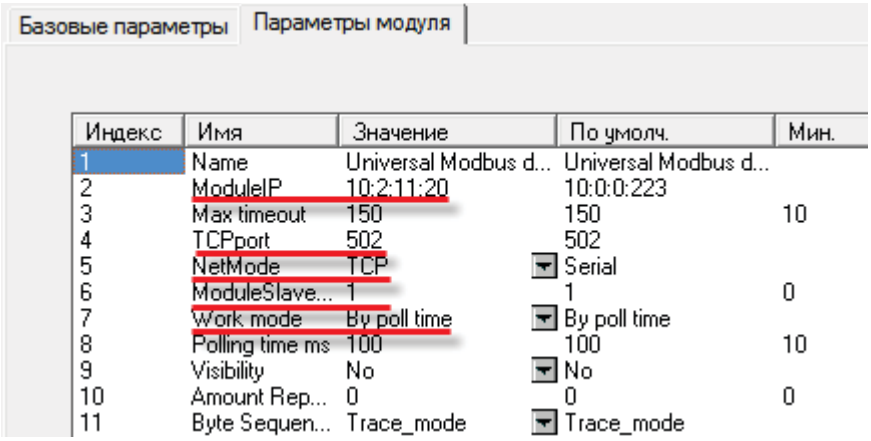


Рис. 6.14. Настройки **Universal Modbus Device (Чтение из СПК)**

В настройках элемента **Запись в СПК** укажем тип связи (**TCP**), IP-адрес, порт и **Slave ID** СПК (в соответствии с табл. 6.1) и режим опроса (**By value change**, т.е. спорадически).

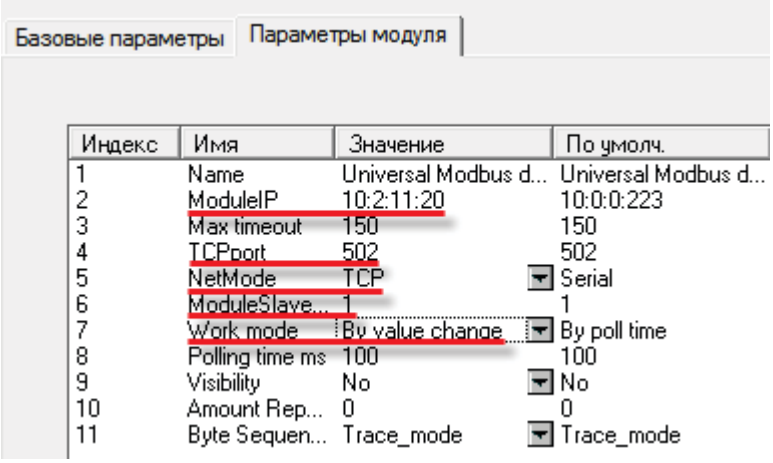


Рис. 6.15. Настройки **Universal Modbus Device (Запись в СПК)**

Нажмем **ПКМ** на элемент **Universal Modbus Device (Чтение из СПК**) и добавим в него подэлементы **Register Input Module**, **Real Input Module** и **String Input Module**.

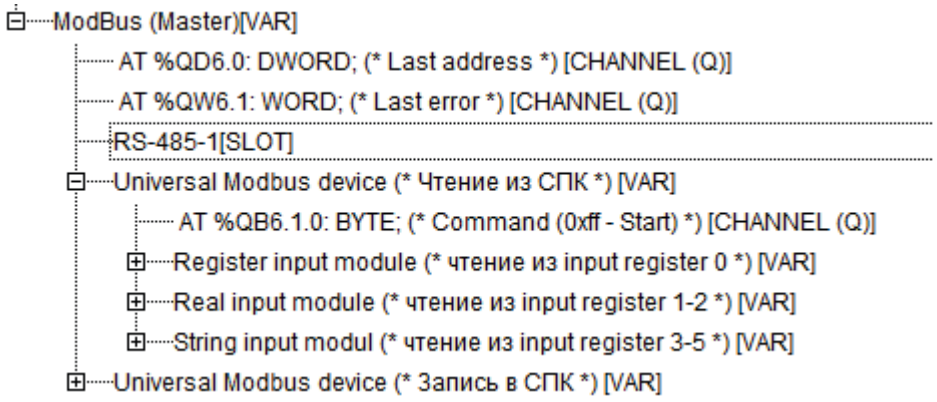


Рис. 6.16. **Universal Modbus Device (Чтение из СПК)** с добавленными **Input** модулями

Привяжем к каждому из каналов переменную (после ввода ее имени она автоматически будет добавлена в список глобальных переменных проекта). Для ввода имени переменной два раза нажмите на **AT**.

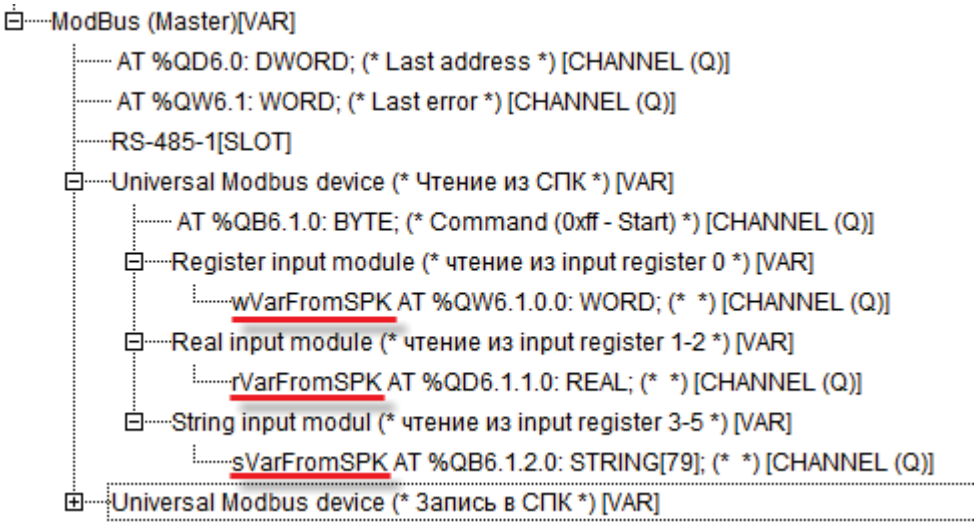


Рис. 6.17. Привязка переменных к каналам

Настройки модулей (используемые регистры СПК в соответствии с табл. 6.1 и функции Modbus) приведены ниже.

Обратите внимание, что при работе с переменными, занимающими несколько регистров СПК (тип **REAL** и **STRING**), указывается только первый из группы регистров.

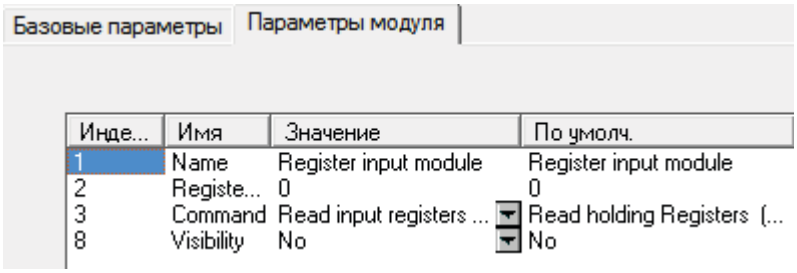


Рис. 6.18. Параметры **Register Input Module**

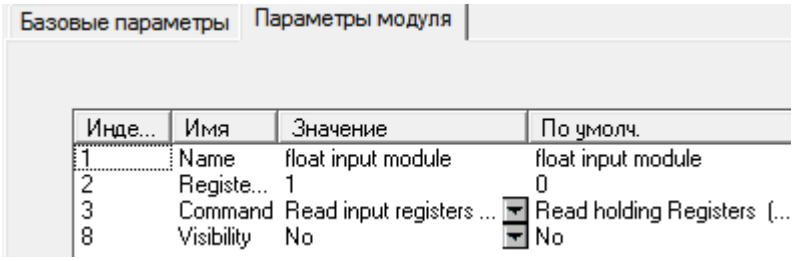


Рис. 6.19. Параметры **Real Input Module**

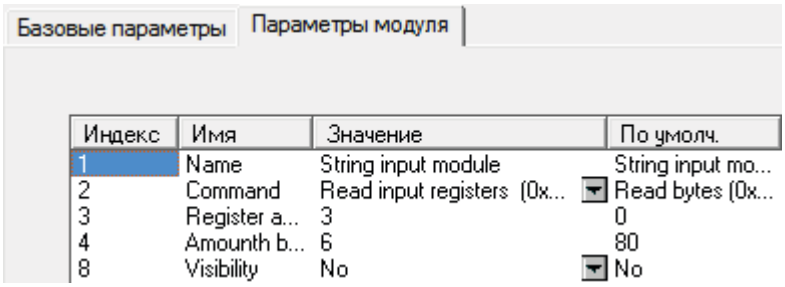


Рис. 6.20. Параметры **String Input Module**

Нажмем **ПКМ** на элемент **Запись в СПК** и добавим в него подэлементы **Register Output Module**, **Real Output Module** и **String Output Module**.

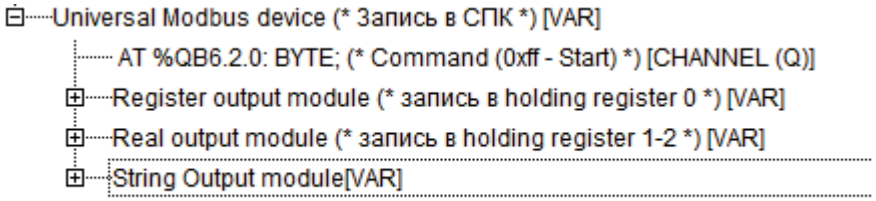


Рис. 6.21. **Universal Modbus Device (Запись в СПК)** с добавленными **Output** модулями

Привяжем к каждому из каналов переменную (после ввода ее имени она автоматически будет создана в проекте как глобальная). Для ввода имени переменной два раза нажмите на **AT**.

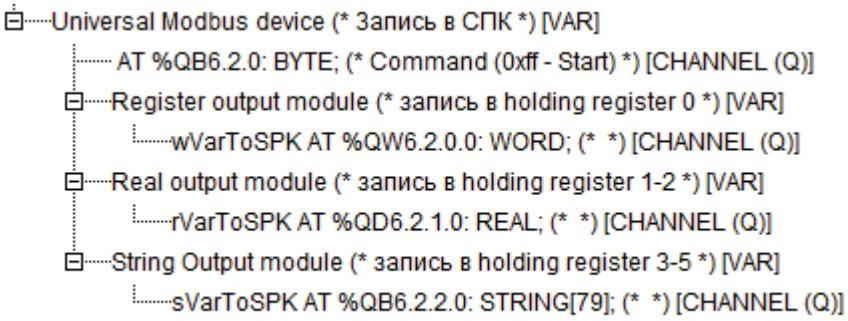


Рис. 6.22. Привязка переменных к каналам

Настройки модулей (используемые регистры СПК в соответствии с табл. 6.1 и функции Modbus) приведены ниже.

Обратите внимание, что при работе с переменными, занимающими несколько регистров СПК (тип **REAL** и **STRING**), указывается только первый из группы регистров

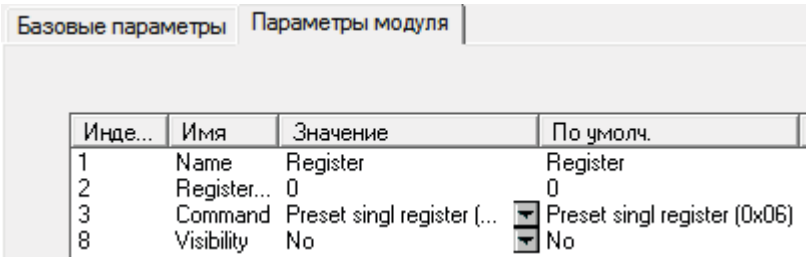


Рис. 6.23. Параметры **Register Output Module**

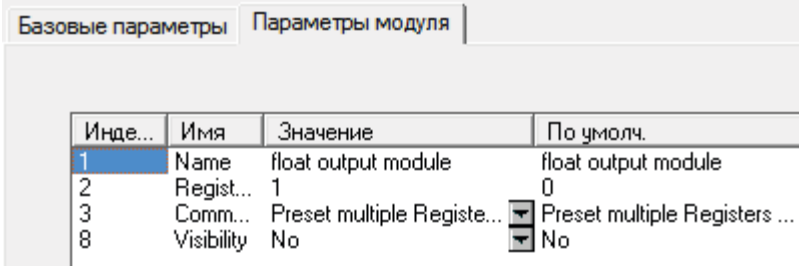


Рис. 6.24. Параметры **Real Output Module**

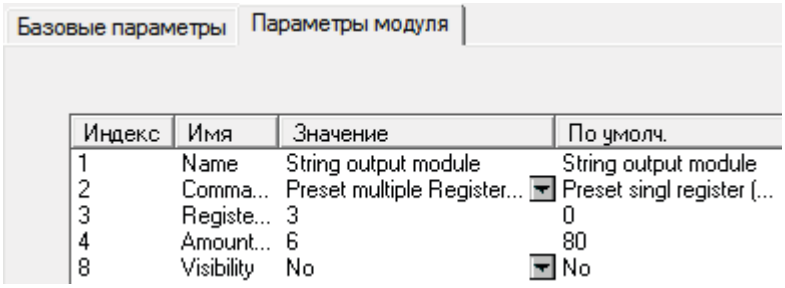


Рис. 6.25. Параметры **String Output Module**

4. Программа **PLC_PRG** будет выглядеть следующим образом:

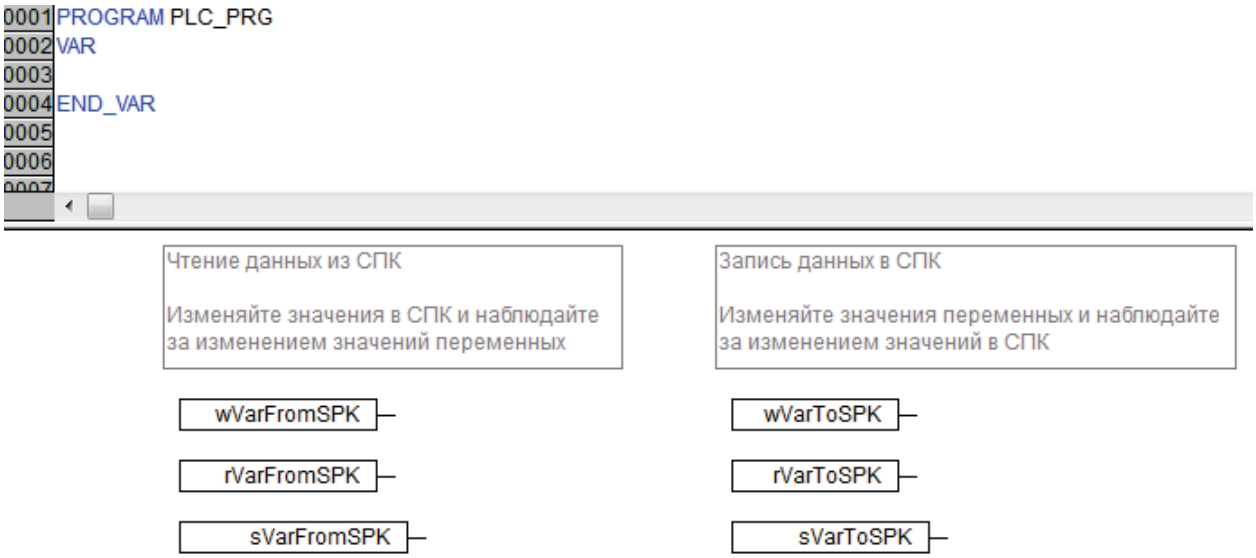


Рис. 6.26 Код программы **PLC_PRG**

На этом настройка **ПЛК (master)** завершена.

Обратите внимание, что проект не содержит каких-либо операций и используется только для отображения и ввода значений. Пользователь должен создать программу для реализации необходимым алгоритмов.

6.4. Работа с примером

Загрузите проекты в оба устройства и запустите их.

Изменяйте значения **ToSPK** переменных в ПЛК и наблюдайте соответствующие изменения в программе СПК:

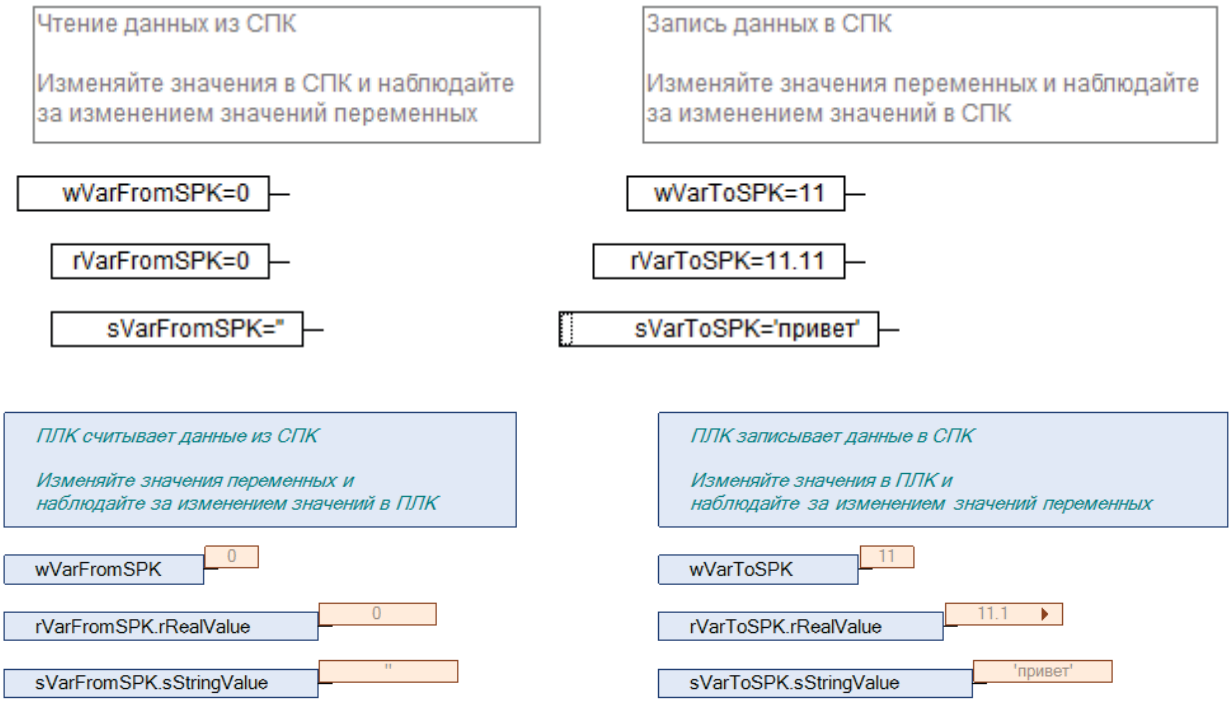


Рис. 6.27. ПЛК записывает данные в СПК

Изменяйте значения **FromSPK** переменных СПК и наблюдайте соответствующие изменения в программе ПЛК.

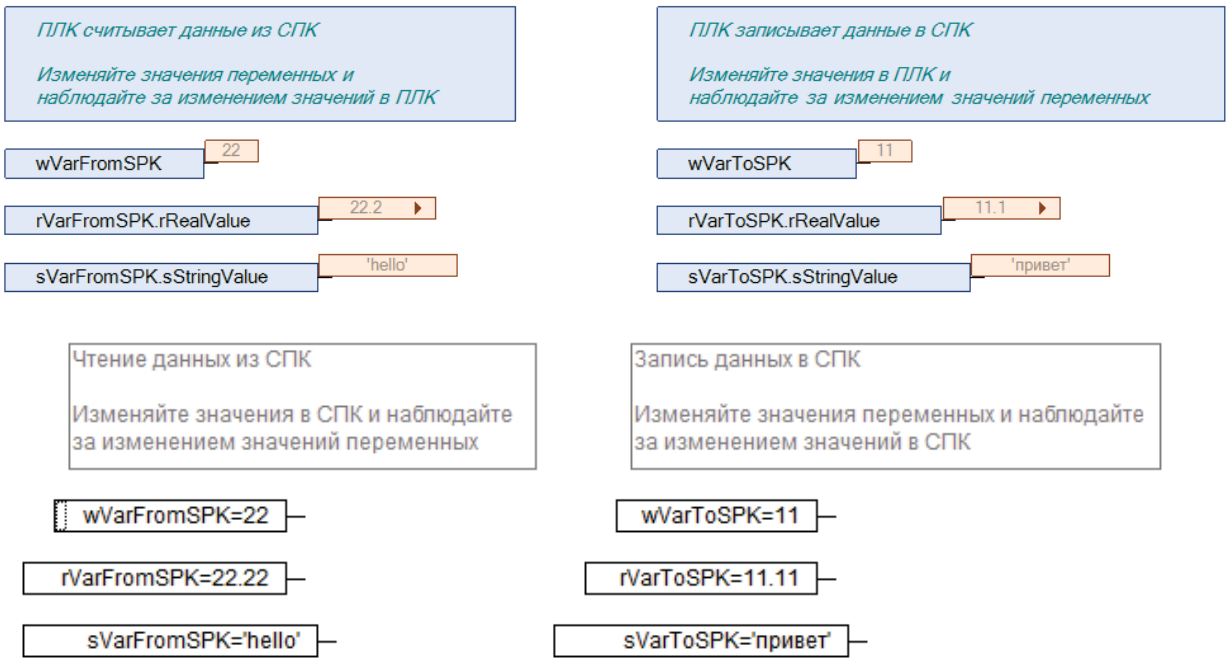


Рис. 6.28. ПЛК считывает данные из СПК