Modbus Universal MasterOPC сервер

Подключение контроллеров OBEH серии ПЛК1хх по протоколу Modbus TCP

Руководство пользователя

ОГЛАВЛЕНИЕ

Mod	dbus U	niversal MasterOPC сервер	1
1	Вве	дение	3
2	Оп	исание контроллеров ОВЕН серии ПЛК1хх	3
3	Нас	тройка контроллера и OPC сервера на протокол Modbus RTU	3
	3.1	Настройка контроллера	3
	3.2	Настройка ОРС сервера	6
4	Доб	бавление Modbus переменных	8
	4.1	Адресация переменных в контроллере	8
	4.2	Добавление переменных в контроллер и ОРС сервер	9
5	Нас	тройка контроллера и OPC сервера на протокол Modbus TCP	20
6	Рек	омендации по организации переменных	22
	6.1	Не использовать переменную 8 Bits	23
	6.2	Задать настройку «Максимально допустимый разрыв адресов в запросе	
	чтени	1я»	23
	6.3	Формировать адреса в определенной последовательности	24
	6.4	Вычислять адреса с помощью функции «Групповые операции»	24

1 Введение

Разработчикам систем управления требуется подключать к SCADA системам различные устройства – модули ввода, регуляторы, программируемые контроллеры. Наиболее распространенным протоколом обмена в промышленности является протокол *Modbus*. Ранее нами была выпущена документация по подключению контроллеров *ABB AC500*, *Siemens S7-1200* и *Delta Electronics* к нашему *Modbus Universal MasterOPC* серверу. В данной статье мы рассмотрим подключение контроллеров фирмы *OBEH* серии *ПЛК1хх*.

2 Описание контроллеров ОВЕН серии ПЛК1хх

Контроллеры OBEH предназначены для построения систем управления котельными установками, объектов водоснабжения, систем вентиляции и кондиционирования, а также различного промышленного оборудования.

Программирование контроллеров осуществляется в среде разработки *Codesys v2.3*. Контроллеры имеют несколько встроенных сетевых интерфейсов – *RS-232, RS-485, Ethernet*, а также имеют поддержку работы с модемами. Контроллеры поддерживают несколько протоколов – *Codesys Gateway* (для связи со средой разработки), протокол *OBEH*, а также поддерживают протокол *Modbus* – версий *Modbus RTU, ASCII* и *TCP*, как в режиме <u>*Master*</u> (ведущий), так и в режиме <u>*Slave*</u> (ведомый).

3 Настройка контроллера и OPC сервера на протокол Modbus RTU

3.1 Настройка контроллера

В качестве примера мы подключим <u>контроллер</u> **ОВЕН ПЛК100**, к **Modbus Universal MasterOPC** серверу по протоколу **Modbus RTU**. Опустим описание создание проекта, выбора целевой платформы контроллера, написание программы и перейдем сразу к настройке **Modbus** протокола.

Настройка **Modbus** осуществляется в <u>окне</u> Конфигурация ПЛК, на <u>закладке</u> Ресурсы (<u>Рисунок 3-1</u>).

🗞 CoDeSys - Examle.pro - [Конфигурация ПЛК]			The second se	
🗐 Файл Правка Проект Вставка Дополнения	Онлайн Окно Справка			- 6 ×
665 5 40 0 ~ 12 2 2 4				
Vision Vision	B - PLC100.R +	Настройки Перенетры надуля Автонатическое вынисление адресов: Контроль перекрытия адресов: Сокранять конф. фейлы в проекте:	् न न	
Tel . and rel . sues transmit (helpschauschneid 200, gold crit	1.00			ОНЛАЙН (ЗАМ (ПРОСМОТР

Сначала в контроллер, через контекстное меню нужно добавить модуль Modbus(Slave).



Рисунок 3-2

В дерево контроллера добавился новый <u>модуль</u>. В его дочерний элемент – **Modbus [Fix]** – нужно добавить <u>интерфейс</u> по которому будет происходить обмен с верхним уровнем. В нашем случае это будет порт **RS-485** (<u>Рисунок 3-3</u>).

Стр. 5

□ PLC100.R □ Discrete input 8	bit[FIX]	•	Базовь	іе параметры	
	- relay[FIX] - relay[FIX]			Идент. модуля:	302
Discrete output	- relay[FIX] - relay[FIX]			Идент, узла;	0
⊡ Discrete output	- relay[FIX]			Адрес входов:	%IB8.0
Discrete output	- relay[FIX]			Адрес выходов:	%QB8.0
B──Special output[F □──ModBus (slave)	IX] [VAR]			Адрес диагностики:	%MB8.0
Modbus[FI	Вставить Элемент			Комментарий:	
	Добавить Подэлемент		+	Debug RS-232	
	Заменить элемент			RS-485-1	
	Вычислять адреса			RS-232	
	Вырезать		Ctrl+X	TCP	
	Копировать		Ctrl+C	Modem	
	Вставить		Ctrl+V		
	Удалить		Del		

Рисунок 3-3

На <u>закладке</u> **Параметры** модуля можно задать <u>параметры обмена</u> – <u>скорость</u>, <u>четность</u>, <u>стоп-биты</u>, <u>тип протокола</u> (**Modbus RTU** или **Modbus ASCII**), <u>задержка</u> <u>ответа</u>. Протокол обмена установим **Modbus RTU**, а остальные параметры оставим по умолчанию.

□—PLC100.R □—Discrete input 8 bit[FIX] □—Discrete output - relay[FIX]	• 111	Базо	вые парам	етры Параметры модул	IЯ			
⊡ Discrete output - relay[FIX]			Индекс	Имя	Значение	По умолч.	Мин.	Макс.
⊡ Discrete output - relay[FIX]			1	Communication speed				
⊡Discrete output - relay[FIX]			3	Pany Data bits	8 bits	8 bits		
Discrete output - relay[FIX]			4	Stop length	One stop bit 📃	One stop bit		
Discrete output - relay[FIX]			5	Interface Lype Frame oriented	RS485	ASCII		
Special output[FIX]			7	Framing time ms	0	0	0	32000
□ModBus (slave)[VAR]			8	Visibility	No	No		
Modbus[FIX]								
RS-485-1[VAR]								

Рисунок 3-4

Теперь укажем <u>адрес</u> нашего контроллера на шине **Modbus**. Данная настройка осуществляется на <u>закладке</u> **Параметры модуля**, элемента **Modbus (Slave)**. Оставим адрес стандартным – 1 (<u>Pucyнok 3-5</u>)



3.2 Настройка ОРС сервера

Настроим ОРС сервер на работу с нашим контроллером. Создадим новую конфигурацию

ОРС сервера и добавим в *Server* новый <u>узел</u>. <u>Тип узла</u> установим – *СОМ*, укажем

параметры связи такие же, как в настройках контроллера (*Рисунок 3-6*).

	MasterOPC Universal Modbus Server key unli modem Build - 3.0.	0.1
Конфигурация	Редактирование коммуникационного узла	
 Создать Сохр Открыть Сохр Сохранить Имп 	Имя узла СОМ-порт	
Файл конфи	Общие настройки	
Текущая конфигу	Комментарий	
Объекты	Включен в работу	True
- I Server	Тип узла	СОМ
	⊟ Настройки СОМ	
	Порт	1
	Скорость	115200
	Данные	8
	Контроль четности	Нет
	Стоп биты	1
	Межсимвольный таймаут (мс)	0
	Использовать режим ASCII	False
	Использовать модем	False
	Скрипт	
	Выполнение скрипта	False
	Дополнительные настройки	
	Slave подключение	False
	Принудительный разрыв соединения в каждом цикле	False
	Пиражировать 1	Да Нет
	Свойства объекта	
Режим Конфигу	урирование Теги : Разрешено - 60000. Загружено	- 0

Рисунок 3-6

Добавим в узел <u>устройство</u>, через контекстное меню узла (Рисунок 3-7).



В окне настройки устройства дадим имя устройству, и зададим адрес – 1 (*Pucyнok 3-8*). Кроме того, необходимо установить настройку *He использовать команду Write Single Coils* в *False*, так как запись битов в контроллер производится функцией 0x05 (если вы планируете работать с отдельными битами памяти).

💑 Редактирование устройства	
Имя устройства ПЛК	
⊟Общие настройки	A
Комментарий	
Включено в работу	True
Тип устройства	MODBUS
Адрес (0х01)	1
Время ответа (мс)	1000
Повторы при ошибке	3
Повторное соединение после ошибки через (с)	10
Реинициализация узла при ошибке	False
Период опроса	1000 _
Размерность периода опроса	ms
Начальная фаза	0
Размерность фазы	ms
Старт после запуска	True
🗏 Скрипт	
Выполнение скрипта	False
🗏 Настройка запросов	
Максимальное количество HOLDING регистров в запросе чтения	125
Максимальное количество INPUT регистров в запросе чтения	125
Не использовать команду WRITE_SINGLE_COIL (0x05)	False
Не использовать команду WRITE_SINGLE_REGISTER (0x06)	True
П Тиражировате 1	Да Нет

Рисунок 3-8

Устройство будет добавлено в дерево (*Рисунок 3-9*).

Текущая	конфигурация : Г	1ЛК100 Example.mbp
Объекты		
⊟ 🗊 Ser ⊟ 🛩 C ⊨ 🌍	ver ЮМ-порт ПЛК	Устройство <<МОЕ Общие настро Комментарий Включено в ра

4 Добавление Modbus переменных

4.1 Адресация переменных в контроллере

Контроллеры ОВЕН могут передавать по *Modbus* переменные следующих типов – *Byte* (1 байт), *Word* (2 байта), *DWord* (4 байта), *Real* (4 байта). В контроллерах *OBEH* все <u>Modbus</u> <u>переменные</u> находятся в одном сегменте памяти, доступ к которой осуществляется с помощью следующих функций:

0x01 – чтение битов, 0x05 – запись бита, регион *Coils*.

0x03 – чтение регистров, 0x10 – запись регистров, регион *Holding Registers*.

Память контроллера можно представить в виде следующей таблицы:

Адрес контроллера		Адрес Modbus бита (регион Coils)							Адрес Modbus регистра (регион Holding Registers)
0x0000	0	1	2	3	4	5	6	7	0x0000
0x0001	8	9	10	11	12	13	14	15	
0x0002	16	17	18	19	20	21	22	23	0x0001
0x0003	24	25	26	27	28	29	30	31	
0x0004	32	33	34	35	36	37	38	39	0x0002
0x0005	40	41	42	43	44	45	46	47	
0x0006	48	49	50	51	52	53	54	55	0x0003
0x0007	56	57	58	59	60	61	62	63	

Таким образом, к байтам памяти контроллера обращаться через регион *Holding Registers*, или обратится через конкретный бит – используя регион *Coils*.

Кроме того, при добавлении переменных используется <u>выравнивание области памяти</u>.

Выравнивание можно описать следующими правилами:

- 1-байтовая переменная (*Byte*) может располагаться в любом адресе памяти контроллера;
- 2-байтовая переменная (*Word*) может располагаться только в четных адресах памяти контроллера;
- 4-байтовая переменная (*DWord* и *Real*) может располагаться только адресах памяти кратных четырем.

Таким образом возможна ситуация, когда отдельные адреса памяти контроллера не будут использоваться.

Подробнее про выравнивание памяти можно прочитать в <u>специальной документации</u> компании OBEH. Кроме того, далее, мы разберем несколько примеров добавления различных типов переменных.

4.2 Добавление переменных в контроллер и ОРС сервер

Добавление Modbus переменных осуществляется через <u>контекстное меню модуля</u>



Modbus (Slave) – <u>Добавить Подэлемент</u>.

Рисунок 4-1

К <u>Modbus переменным</u> относятся элементы – **2 byte** (тип **Word**), **8 bits** (тип **Byte**), **4 byte** (тип **DWord**), **Float** (тип **Real**) (<u>Pucyнoк 4-1</u>).

4.2.1 Добавление переменных типа Byte

Добавим нескольких различных *Modbus* переменных. Сначала добавим переменную 8

bits (<u>Рисунок 4-2</u>).

ModBus (slave)[VAR]
 Modbus[FIX]
 RS-485-1[VAR]
 AT %QB8.1.0: BYTE; (* *) [CHANNEL (Q)]
 AT %QX8.1.0.0: BOOL; (* Bit 0 *)
 AT %QX8.1.0.1: BOOL; (* Bit 1 *)
 AT %QX8.1.0.2: BOOL; (* Bit 2 *)
 AT %QX8.1.0.3: BOOL; (* Bit 3 *)
 AT %QX8.1.0.4: BOOL; (* Bit 3 *)
 AT %QX8.1.0.5: BOOL; (* Bit 4 *)
 AT %QX8.1.0.6: BOOL; (* Bit 6 *)
 AT %QX8.1.0.7: BOOL; (* Bit 7 *)

Рисунок 4-2

Данная переменная имеет тип **Byte**, размер переменной составляет 1 байт, при этом к каждому биту переменной можно обратится через переменную, поэтому, как правило, данную переменную используют для передачи дискретных значений. Присвоим первым двум битам переменные, назовем их **Discrete1** и **Discrete2** (<u>Pucyнok 4-3</u>).

```
ModBus (slave)[VAR]
Modbus[FIX]
RS-485-1[VAR]
AT %QB8.1.0: BYTE; (* *) [CHANNEL (Q)]
Discrete1 AT %QX8.1.0.0: BOOL; (* Bit 0 *)
Discrete2 AT %QX8.1.0.1: BOOL; (* Bit 1 *)
AT %QX8.1.0.2: BOOL; (* Bit 2 *)
AT %QX8.1.0.3: BOOL; (* Bit 3 *)
AT %QX8.1.0.4: BOOL; (* Bit 3 *)
AT %QX8.1.0.5: BOOL; (* Bit 5 *)
AT %QX8.1.0.6: BOOL; (* Bit 6 *)
AT %QX8.1.0.7: BOOL; (* Bit 7 *)
```

Рисунок 4-3

Теперь добавим такие же переменные в ОРС сервер. Для этого через контекстное меню устройства добавим тег. Для удобства зададим ему такое же имя в *Codesys – Discrete1*.

<u>Регион тега</u> будет **Coils**. <u>Адрес</u> тега будет равен **0**. Остальные параметры можно оставить стандартными (<u>Рисунок 4-4</u>).

🎎 Редактирс	ование тега		
Имя тега	Discrete1		
🗉 Общие	настройки		
Коммент	гарий		
Включен	н в работу	True	
Регион		COILS	
Адрес	(0x0000)	0	
Тип дан	ных в устройстве	bool	
Тип дан	ных в сервере	bool	
Тип дост	гупа	ReadWrite	
🗏 Скрипт			
Разреше	ение выполнения скрипта после чтения	False	
Разреше	ние выполнения скрипта перед записью	False	
🗏 Дополн	ительно		
Наличие	е отдельного регистра записи	False	
Чтение о	сразу после записи	False	
HDA			
HDA дос	ступ	False	
🗌 Тира:	жироват 1	Да	Нет

Рисунок 4-4

Аналогично добавим второй тег, дадим ему имя *Discrete2*, адрес укажем 1 (*Рисунок 4-5*).

1	Редактиро	ование тега					
	Имя тега	Discrete2					
6	Общие	настройки					
	Коммент	арий					
	Включен	н в работу	True				
	Регион		COILS				
	Адрес	(0x0000)	1				
	Тип дан	ных в устройстве	bool				
	Тип дан	ных в сервере	bool				
	Тип дост	гупа	ReadWrite				
	Скрипт						
	Разреше	ние выполнения скрипта после чтения	False				
	Разреше	ние выполнения скрипта перед записью	False				
	Дополн	ительно					
	Наличие	е отдельного регистра записи	False				
	Чтение о	сразу после записи	False				
	HDA						
	HDA дос	туп	False				
┝							
	🗌 Тира:	жироват 1	Да Нет				

Рисунок 4-5

Проверим получение данных. Подключимся к контроллеру, запишем в него программу, а также запустим ОРС сервер в режим исполнения.

Из среды разработки изменим состояние одного из битов переменной – изменение отобразилось в ОРС сервере (*Рисунок 4-6*).

Discrete output - relay[FIX]			Master	OPC Universal	Modbus Se	erver key un	li modem E	Build - 3.0.0.1		_ 0	×
Discrete output - relay[FIX]	ROPC										
Discrete output - relay[FIX]	Стартовая	конфигураци	я : ПЛК10	00 Example	e.mbp						
Discrete output - relay[FIX]	Объекты										
Special output[FIX]	B II Son	or	Устройст		>						
⊡ModBus (slave)[VAR]		ОМ-порт	Jerpower								
□Modbus[FIX]	0	плк	Теги								
RS-485-1[VAR]		Discrete1	ИМЯ			Реги	Адрес	Значе	Качест	Время (U	Тип в
ia8 bits[VAR]		Discretez	СОМ-по	орт.ПЛК.Di	iscrete1	COILS	(0x0	True	ок	2014-04	bool
А <u>Т %QB8.1.0: ВҮТЕ; (*_*) [CHANNEL (Q)]</u> – 1			сом-по		iscrete2	COILS	(0x0	Faise	OK	2014-04	bool
Discrete1 AT %QX8.1.0.0: BOOL; (* Bit											
Discrete2 AT %QX8.1.0.1; BOOL; (* Bit											
AT %QX8.1.0.2: BOOL; (* Bit 2 *)											
AT %QX8.1.0.3: BOOL; (* Bit 3 *)											
AT %QX8.1.0.4: BOOL; (* Bit 4 *)											
AT %QX8.1.0.5: BOOL; (* Bit 5 *)											
AT %QX8.1.0.6: BOOL; (* Bit 6 *)			•								•
• AI %QX8.1.0.7: BOOL; (* Bit 7 *)			Сообще	ния Запрос	сы Сообь	цения скр	иптов				
			Режим	вывода: З	Запушен	Фильтр	: плк				•
			07-04-2	2014 08:58	8:47.169	Э СОМ-по	орт::ПЛК	:(COM1)	Rx: [0006] (01 01 01 01	90 🗏 📗
			07-04-2	2014 08:58	8:47.163	3 СОМ-по	орт::ПЛК	:(COM1)	Fx: [0008] (01 01 00 00	00 (
			07-04-2	2014 08:58	8:46.113	3 СОМ-по	орт::ПЛК	:(COM1)	Rx: [0006] (01 01 01 01	90 ·
			07-04-2	2014 08:58	8:46.107	7 СОМ-по	орт::ПЛК	:(COM1) 1	Гх: [0008] (01 01 00 00	00 (
			07-04-2	2014 08:58	8:45.057	7 СОМ-по	орт::ПЛК	:(COM1)	Rx: [0006] (01 01 01 01	90 ·
			07-04-2	2014 08:58	8:45.050) СОМ-по	орт::ПЛК	:(COM1) 1	Fx: [0008] (01 01 00 00	00 (
			07-04-2	2014 08:58	8:44.000	COM-no	орт::ПЛК	:(COM1)	Rx: [0006] (01 01 01 01	90 ·
			07-04-2	2014 08:58	8:43.994	COM-no	орт::ПЛК	:(COM1) 1	Fx: [0008] (01 01 00 00	00 (
			07-04-2	2014 08:58	8:42.929	Э СОМ-по	орт::ПЛК	:(COM1)	Rx: [0006] (01 01 01 01	90 . *
	Режим	RunTime		Клиен	ты DA -	0 Клиен	ты HDA -	0			

Рисунок 4-6

Отключимся от контроллера, и остановим режим исполнения ОРС сервера.

4.2.2 Добавление переменных типа Word (uint16)

Теперь добавим в контроллер две целочисленных переменных типа Word – элемент 2

Modbus (🖆			
b-Modi_	Вставить Элемент	•	
	Добавить Подэлемент	+	Statistic
B bit	Заменить элемент		Button
	Вычислять адреса	Universal network module	
	Вырезать	Ctrl+X	ModBus (Master)
	Копировать	CtrLLC	DCON (Master)
	Поторовать	Challer	OWEN (slave)
	Вставить	Ctrl+V	OWEN (spv)
	Удалить	Del	Owen (Master)
			ModBus (slave)
			2 byte
			8 bits
			4 byte
			Float
			File

Рисунок 4-7

Дадим имена переменным, назовем из VarWord1 и VarWord2 (Pucyнok 4-8)

⊟ModBus (slave)[VAR]
- ₽Modbus[FIX]
RS-485-1[VAR]
₽ 8 bits[VAR]
- ₽2 byte[VAR]
VarWord1 AT %QW8.2.0: WORD; (* *) [CHANNEL (Q)]
2 byte[VAR]
VarWord2 AT %QW8.3.0: WORD; (* *) [CHANNEL (Q)]

Рисунок 4-8

Добавим теги в ОРС сервер. <u>*Регион*</u> тега будет использоваться **Holding Registers**. Нулевой адрес памяти контроллера уже занят байтовой переменной, а значит и занят весь нулевой Modbus адрес. Поэтому у переменной **VarWord1** адрес Modbus регистра будет равен **1**. Ситуацию можно проиллюстрировать с помощью таблицы:

Адрес контроллера	Расположение переменных				Адрес Modbus регистра (регион Holding Registers)	
0x0000	Discrete1	Discrete2				
0x0000	(бит0)	(бит1)				0x0000
0x0001	Незанятое пространство					
0x0002						
		Va	arWord1			0x0001
0x0003						
0x0004						
		Va	arWord2			0x0002
0x0005						

Добавим тег, имя также дадим *VarWord1*, <u>peruon</u> – *Holding Registers*, <u>adpec</u> – 1, <u>mun</u> <u>daнных в устройстве</u> – *uint16* (соответствует типу *Word*), <u>mun daнных в сервере</u> – *uint32*. Остальные параметры можно оставить по умолчанию (<u>Pucyнok 4-9</u>).

💃 Редактирование тега	
Имя тега VarWord1	
⊟Общие настройки	A
Комментарий	
Включен в работу	True
Регион	HOLDING_REGISTERS
Адрес (0х0001)	1
Тип данных в устройстве	uint16
Тип данных в сервере	uint32
Тип доступа	ReadWrite
Использовать перестановку байтов в значении	True
Перестановка байтов в значении	10325476 =
Последний тег в групповом запросе	False
Пересчет (А*Х + В)	False
🗏 Скрипт	
Разрешение выполнения скрипта после чтения	False
Разрешение выполнения скрипта перед записью	False
🖻 Дополнительно	
Наличие отдельного регистра записи	False
Извлечение бита из данных	False
Чтение сразу после записи	False
Принудительная запись командой б	False
	Ť
Птиражироват 1	Да Нет

Рисунок 4-9

Аналогично добавим второй тег, его <u>адрес</u> Modbus регистра будет равен **2** (<u>Рисунок 4-10</u>).

Server	Ter < <holding_registers>> : VarWord2</holding_registers>		
	Комментарий		
Discrete1	Включен в работу		True
Discrete2	Адрес	(0x0002)	2
VarWord1	Тип данных в устройстве		uint16
····· 😰 VarWord 2	Тип данных в сервере		uint32
	Тип доступа		ReadWrite

Рисунок 4-10

Проверим получение данных – подключимся к контроллеру и обновим программу, а также запустим ОРС сервер в режим исполнения.

Изменим в контроллере одно из значений – значение отобразилось в ОРС сервере (*Рисунок 4-11*).

e──Discrete output - relay[FIX] e──Discrete output - relay[FIX] e──Discrete output - relay[FIX]		Maste	erOPC Universal Modbus Serv	er key unli n	iodem Build	d - 3.0.0.1			x
Discrete output - relay[FIX]	Стартовая конфигурац	ия : ПЛК	100 Example.mbp						
BSpecial output[FIX]	Объекты								
⊡ModBus (slave)[VAR]	🕀 🕷 Server	Теги							
éModbus[FIX]		Имя		Реги	Адрес	Значе	Качест	Время (U	Ти
₽····8 bits[VAR]		COM-	порт.ПЛК.Discrete1	COILS	(0x0	False	OK	2014-04	bo
E vyte[VAR]		COM-	порт.ПЛК.Discrete2 nopt.ПЛК.VarWord1	HOL	(0x0	1 alse 100	ок	2014-04 2014-04	bo uii
	L L	сом-	порт.ПЛК.VarWord2	HOL	(0x0	0	ОК	2014-04	uiı
VarWord2 AT %QW8.3.0: WORD; (* *) [CHANNEL (Q)] = 0									

Рисунок 4-11

4.2.3 Добавление переменной типа Real (Float)

Добавим 4-байтовую переменную – переменную типа Real. Добавим подэлемент Float

(Рисунок 4-12). ⊟ ModBus (₽ Вставить Элемент doM Добавить Подэлемент Þ Statistic... L..... Button... Заменить элемент8 bits Вычислять адреса Universal network module... ḋ----2 byt ModBus (Master)... Ctrl+X Вырезать DCON (Master)... b ____2 byt Копировать Ctrl+C OWEN (slave)... Ctrl+V . Вставить OWEN (spy)... Del Удалить Owen (Master)... ModBus (slave)... 2 byte... 8 bits... 4 byte... Float... R File...

Рисунок 4-12

Дадим ей имя VarFloat1 (<u>Рисунок 4-13</u>).



Рисунок 4-13

Определим адрес этой Modbus переменной. Последний использованный адрес памяти -*0x0005,* занят переменной *VarWord2*. Следующий за ним адрес – *0x0006*. Однако согласно правилам выравнивания, 4 байтовые переменные (которой является переменная *Float*) могут располагаться только в адресах памяти кратных четырем. Значит переменная будет находится в следующем ближайшем адресе, который будет делиться на 4, то есть - *0x0008*.

Ситуацию можно проиллюстрировать на таблице:

Адрес контроллера	Расположение переменных					Адрес Modbus регистра (регион Holding Registers)					
0x0000	Discrete1	Discrete2									
	(бит0)	(бит1)						0x0000			
0x0001		Незанято	е прост	ранств	0						
0x0002		Ve	rWord1					0x0001			
0x0003											
0x0004		Ver Word?					0x0002				
0x0005											
0x0006		Незанято	е прост	ранств	0			0x0003			
0x0007					•						
0x0008								0x0004			
0x0009		V	arFloat1								
0x000A								0x0005			
0x000B											

Таким образом <u>адрес</u> **0x0003** останется не использованным, а переменная **VarFloat** будет занимать **Modbus** адреса **0x0004** и **0x0005**.

Добавим в ОРС сервер <u>тег</u>. Зададим ему имя – VarFloat1, адрес – 4, <u>тип в устройстве</u> – Float, <u>тип в сервере</u> – Float. Также нужно указать правильное <u>чередование байт.</u> Обычно для четырехбайтовых переменных чередование устанавливается в режим Старшим словом вперед (32107654), однако в контроллерах ОВЕН ПЛК1хх чередование байт у четырехбайтовых переменных такое же, как и двухбайтовых – Старшим байтом вперед (10325476). (<u>Рисунок 4-14</u>)

Примечание. Данная особенность свойственна только контроллерам

ОВЕН ПЛК1xx. Остальные приборы данного производителя

(регуляторы, модули ввода-вывода) для четырехбайтных переменных

используют чередование байт «Старшим словом вперед».

Имя тега	VarFloat1		
Общие	настройки		
Коммен	тарий		
Включе	н в работу	True	
Регион		HOLDING_REGISTERS	
Адрес	(0x0004)	4	
Тип дан	ных в устройстве	float	
Тип дан	ных в сервере	float	
Тип дос	тупа	ReadWrite	
Использ	зовать перестановку байтов в значении	True	
Переста	новка байтов в значении	10325476	
Последн	ний тег в групповом запросе	False	
Пересче	ет (А*Х + В)	False	
Скрипт			
Разреше	ение выполнения скрипта после чтения	False	
Разреше	ение выполнения скрипта перед записью	False	
Дополн	ительно		
Наличи	е отдельного регистра записи	False	
Извлече	ение бита из данных	False	
Чтение	сразу после записи	False	
Принуд	ительная запись командой б	False	

Рисунок 4-14

Тег добавится в устройство (*Рисунок 4-15*)

Server	Ter < <holding_registers>> : VarFloat1</holding_registers>		
	Комментарий		
Discrete1	Включен в работу	True	
Discrete2	Адрес (0х0004)	4	
🛛 🛱 VarWord1	Тип данных в устройстве	float	
VarWord2	Тип данных в сервере	float	
	Тип доступа	ReadWrite	
	Использовать перестановку байтов в значении	True	



Аналогично проверим получение данных с контроллера (*Рисунок 4-16*).

Modbus Universal MasterOPC сервер. Подключение контроллеров ОВЕН ПЛК1хх Стр. 18



Рисунок 4-16

4.2.4 Добавление переменной типа DWord

Переменная типа DWord добавляется через подэлемент 4 Byte (Рисунок 4-17 и Рисунок





Рисунок 4-17

```
Special output[FIX]
ModBus (slave)[VAR]
Modbus[FIX]
—RS-485-1[VAR]
8 bits[VAR]
2 byte[VAR]
—2 byte[VAR]
—VarWord1 AT %QW8.2.0: WORD; (* *) [CHANNEL (Q)]
E—2 byte[VAR]
—VarWord2 AT %QW8.3.0: WORD; (* *) [CHANNEL (Q)]
E—Float[VAR]
—VarFloat1 AT %QD8.4.0: REAL; (* *) [CHANNEL (Q)]
E—4 byte[VAR]
—VarDWord1 AT %QD8.5.0: DWORD; (* *) [CHANNEL (Q)]
```

Рисунок 4-18

Данная переменная является 4-байтовой, поэтому к ней применимы те же правила что и для переменной типа *Float*. В данном случае переменная *VarDWord1* будет располагаться в памяти контроллера по адресам 0x000С – 0x000F, которым соответствуют Modbus адреса *6* и *7*.

В ОРС сервере настройки тега <u>Тип данных в устройстве</u> и <u>Тип данных в сервере</u>, нужно задать *uint32*. <u>Чередование байт</u> – также, как и у *Float*, *старшим байтом вперед* (<u>Pucyнok 4-19</u>).

💃 Редактирование тега	
Имя тега VarDWord1	
Общие настройки	
Комментарий	
Включен в работу	True
Регион	HOLDING REGISTERS
Адрес (0х0006)	6
Тип данных в устройстве	uint32
Тип данных в сервере	uint32
Тип доступа	ReadWrite
Использовать перестановку байтов в значении	True
Перестановка байтов в значении	32107654 =
Последний тег в групповом запросе	False
Пересчет (А*Х + В)	False
⁼ Скрипт	
Разрешение выполнения скрипта после чтения	False
Разрешение выполнения скрипта перед записы	o False
Дополнительно	
Наличие отдельного регистра записи	False
Извлечение бита из данных	False
Чтение сразу после записи	False
Принудительная запись командой 6	False
Пиражироват 1	Да Нет

Рисунок 4-19

5 Настройка контроллера и ОРС сервера на протокол Modbus TCP

Если опрос контроллера планируется вести по протоколу **Modbus TCP**, то в <u>модуль</u> **Modbus[FIX]** нужно добавить <u>элемент</u> **TCP**. (<u>Pucyнok 5-1</u>)

Modbus[FIX1			
8 bits[VAR	Вставить Элемент		
2 byte[VAI	Добавить Подэлемент	+	Debug RS-232
2 byte[VAI	Заменить элемент		RS-485-1
Float[VAR]	Вычислять адреса		RS-232
4 byte[VAI	Вырезать	Ctrl+X	TCP
	Копировать	Ctrl+C	Modem
	Вставить	Ctrl+V	
	Удалить	Del	

На <u>з*акладке</u> Параметры модуля есть лишь одна настройка – <u>номер порта TCP</u>, по которому будет происходит обмен. По умолчанию – 502 (<u>Рисунок 5-2</u>).</u>*





IP задается самому контроллеру. Это делается при установленном соединении, с помощью окна *ПЛК-Браузер*. Для задания IP адреса используется команда *SetIP* (*Рисунок* <u>5-3</u>).



После выполнения команды контроллер необходимо перезагрузить.

В ОРС сервере необходимо добавить узел, <u>тип узла</u> задать – **ТСР/IР**. Задать необходимые <u>параметры связи</u> – **IP адрес** контроллера, и заданный в настройках **порт** (<u>Pucyнok 5-4</u>). Также можно включить настройку **Отслеживать Transaction ID** – если данная настройка включена, то в специальном поле **Modbus TCP** запроса, будет меняться поле <u>идентификатора запроса</u>, что позволяет избежать коллизий разных запросов, при медленном ответе со стороны устройств.

🞎 Редактирование коммуникационного узла	
Имя узла ТСР	
Общие настройки	
Комментарий	
Включен в работу	True
Тип узла	TCP/IP
Настройки TCP/IP	
IP адрес	192.168.100.6
IР порт	502
🗏 Скрипт	
Выполнение скрипта	False
🗏 Дополнительные настройки	
Slave подключение	False
Modbus поверх TCP	False
Отслеживать Transaction ID	True
Принудительный разрыв соединения в каждом цикле	False
Птиражировать 1	Да Нет

Рисунок 5-4

После добавления <u>узла</u>, в него добавляется <u>устройство</u>, и указывается **адрес**, установленный в настройках <u>модуля</u> **Modbus (Slave)** в контроллере (<u>Рисунок 5-5</u>). Как правило данный адрес оставляют равным **1**.

👺 Редактирование устройства		X
Имя устройства ПЛК ТСР		
⊟Общие настройки		
Комментарий		
Включено в работу	True	
Тип устройства	MODBUS	
Адрес (0х01)	1	
Время ответа (мс)	1000	
Повторы при ошибке	3	
Повторное соединение после ошибки через (с)	10	
Реинициализация узла при ошибке	False	
Период опроса	1000	≡
Размерность периода опроса	ms	
Начальная фаза	0	
Размерность фазы	ms	
Старт после запуска	True	
🗏 Скрипт		
Выполнение скрипта	False	
⁼ Настройка запросов		
Максимальное количество HOLDING регистров в запросе чтения	125	
Максимальное количество INPUT регистров в запросе чтения	125	
Не использовать команду WRITE_SINGLE_COIL (0x05)	True	
Не использовать команлу WRITE SINGLE REGISTER (0x06)	True	-
Птиражировате 1 🗭 Да	Нет	

Рисунок 5-5

Добавление тегов и определение их адресов осуществляется аналогично, как и при

работе по *Modbus RTU*.

При необходимости можно добавить в *Modbus[FIX]* несколько <u>интерфейсов</u> – например

RS-485 и **TCP** (<u>Рисунок 5-6</u>),что позволяет опрашивать контроллер одновременно

несколькими ведущими (например, ОРС сервером и панелью оператора).

ModBus (slave)[VAR]
Modbus[FIX]
TCP[VAR]
RS-485-1[VAR]

Рисунок 5-6

Также можно добавить два модуля **ТСР** – для опроса несколькими сетевыми

устройствами по локальной сети. Однако в этом случае, необходимо в настройках

каждого элемента TCP, задать разные <u>порты</u> (например, **502** и **503**) по которым и вести опрос.

6 Рекомендации по организации переменных

Для упрощения настройки обмена по Modbus далее будут приведены несколько рекомендаций.

6.1 Не использовать переменную 8 Bits

Как было сказано ранее в контроллере **OBEH**, все **Modbus** переменные размещены в одном сегменте памяти, к которому можно обращаться через <u>peruon</u> **Holding Registers** или **Coils**.

Переменную **8 bits**, обычно используют для передачи отдельных бит. Однако гораздо эффективнее использовать для этих целей переменную типа **Word** (<u>подэлемент</u> **2 Byte**). Для записи и чтения отдельных битов в **Codesys** можно использовать специальные функциональные блоки – **Pack** и **Unpack** (библиотека «**Util.lib**»), а на языке **ST** можно обращаться к отдельным битам через точку (например **VarWord1.0:=true**). Если на верхнем уровне используется MasterSCADA, то для упаковки и извлечения битов

можно использовать <u>ФБ</u> «**Упаковка 32-битного значения»** и «**Распаковка 32-битного** значения».

Отказ от работы с отдельными битами через *регион Coils* позволит сэкономить как лицензионные теги OPC сервера, так и SCADA системы, а кроме того снизит сетевую нагрузку по обмену данных.

6.2 Задать настройку «Максимально допустимый разрыв адресов в запросе чтения».

Из-за правил выравнивания может получится, что некоторые промежуточные *Modbus* адреса окажутся неиспользованными. В нашем примере сначала идут переменные *VarWord1* и *VarWord2* с адресами *1* и *2*, а затем *VarFloat1* с адресом *4*, и *VarDWord1* с адресом *6*, т.е. адрес *3* не используется. В этом случае OPC сервер выполнит два Modbus запроса – сначала опросит адреса *1* и *2*, а затем адреса с *4* по *7*.

При необходимости можно сделать, чтобы подобные регистры были опрошены за один запрос чтения. Для этого необходимо у устройства задать настройку *Максимально допустимый разрыв в запросе чтения* (*Рисунок 6-1*). По умолчанию данная настройка равна нулю. Если же задать данный параметр, то все разрывы адресов меньшие заданного значения будут игнорироваться.

То есть, если в данном случае мы установим этот параметр равным **1**, то ОРС сервер запросит регистр с **1** по **7** одним запросом. Пустой, третий, Modbus регистр также будет опрошен, но его значение сервер просто проигнорирует.

Текущая конфигурация : ПЛК10	00 Example.mbp	
Объекты		
Server	Устройство < <modbus>> : ПЛК</modbus>	
	Включено в работу	Irue 🔺
Discrete1	Адрес (0x01)	1
Discrete2	Время ответа (мс)	1000
VarWord1	Повторы при ошибке	3
VarWord2	Повторное соединение после ошибки через (с)	10
VarDWord1	Реинициализация узла при ошибке	False
	Период опроса	1000
	Размерность периода опроса	ms
	Начальная фаза	0
	Размерность фазы	ms
	Старт после запуска	True _
	□ Скрипт	E
	Выполнение скрипта	False
	⊟ Настройка запросов	
	Максимальное количество HOLDING регистров в запросе чтения	125
	Максимальное количество INPUT регистров в запросе чтения	125
	Не использовать команду WRITE_SINGLE_COIL (0x05)	True
	Не использовать команду WRITE_SINGLE_REGISTER (0x06)	True
	Максимальное допустимый разрыв адресов в запросе чтения	1
	Использовать преамбулу	False 🔻

6.3 Формировать адреса в определенной последовательности

Чтобы уменьшить количество неиспользуемых ячеек памяти, и упростить подсчет Modbus адресов, рекомендуем структурировать переменные по типам. Например – сначала добавляем переменные типа **Word** (**2 Byte**), затем добавляем переменные типа **Real** (**Float**).

6.4 Вычислять адреса с помощью функции «Групповые операции»

При добавлении ОРС переменных можно легко ошибиться при задании *Modbus* адресов. В третьей версии *Modbus Universal MasterOPC* сервера появилась новая функция – *Групповые операции*. С помощью данной функции можно быстро вносить изменения в группе тегов – менять адреса, тип данных в устройстве, чередование байт. Данная функция может облегчить и задание адресов для опроса контроллеров OBEH. Рассмотрим следующим пример. В модуль *Modbus (Slave)* добавлено **7** переменных типа *Word, 4* переменных типа *Float* и **2** переменных типа *DWord* (*<u>Рисунок 6-2</u>).*

- ModBus (slave)[VAR]
 -Modbus[FIX]
 - ia model = 2 byte[VAR]

 - 2 byte[VAR]

 - Float[VAR]
 - Float[VAR]
 - a Hoat(VAR) a 4 byte[VAR]

Создадим конфигурацию ОРС сервера для их опроса.

Добавим в <u>устройство</u> теги в таком же порядке, как и модуля Modbus (Slave) и

соответствующими типами данных (2 byte – uint16, Float – Float, 4 Byte – uint32). Адрес у

всех тегов можно оставлять равным нулю или оставить вариант предлагаемым сервером

- позже мы вычислим через <u>групповые операции</u>.

В итоге у нас получилась следующая ОРС конфигурация:

	Теги				
	Имя	Адрес	Регион	Тип в серве	Тип в устройс
Word1	Порт.ПЛК.Word1	(0x0000) 0	HOLDING_REGI	uint32	uint16
Word2	Порт.ПЛК.Word2	(0x0000) 0	HOLDING_REGI	uint32	uint16
Word3	Порт.ПЛК.Word3	(0x0000) 0	HOLDING_REGI	uint32	uint16
Word5	Порт.ПЛК.Word4	(0x0000) 0	HOLDING_REGI	uint32	uint16
Word6	Порт.ПЛК.Word5	(0x0000) 0	HOLDING_REGI	uint32	uint16
Float1	Порт.ПЛК.Word6	(0x0000) 0	HOLDING_REGI	uint32	uint16
Float2	Порт.ПЛК.Word7	(0x0000) 0	HOLDING_REGI	uint32	uint16
Float3	Порт.ПЛК.Float1	(0x0000) 0	HOLDING_REGI	float	float
DWord1	Порт.ПЛК.Float2	(0x0000) 0	HOLDING_REGI	float	float
DWord2	Порт.ПЛК.Float3	(0x0000) 0	HOLDING_REGI	float	float
	Порт.ПЛК.Float4	(0x0000) 0	HOLDING_REGI	float	float
	Порт.ПЛК.DWord1	(0x0000) 0	HOLDING_REGI	uint32	uint32
	Порт.ПЛК.DWord2	(0x0000) 0	HOLDING_REGI	uint32	uint32
					4
	Свойства объекта Табл	пица тегов			

Рисунок 6-3

Через контекстное меню устройства вызовем команду Групповые операции (Рисунок

<u>6-4</u>).

Объекты				
B Server	Те	СП		
	И	МЯ	Адрес	Регис
1	Добавить	IK.Word1	(0x0000) 0	HOLD
* *	Переименовать	1K.Word2	(0x0000) 0	HOLD
¥ 12	Incpennenobarb	1K.Word3	(0x0000) 0	HOLD
- T	Удалить	1K.Word4	(0x0000) 0	HOLD
1	Вырезать	1K.Word5	(0x0000) 0	HOLD
작 영	V Е Копировать	1K.Word6	(0x0000) 0	HOLD
174	Групповые операцки	1K.Word7	(0x0000) 0	HOLD
- 1	А Экспорт устройства	1K.Float1	(0x0000) 0	HOLD
よ で 1	Г Импорт полустройства	1K.Float2	(0x0000) 0	HOLD
1		1K.Float3	(0x0000) 0	HOLD
	Экспорт тегов	1K.Float4	(0x0000) 0	HOLD
	Импорт тегов	IK.DWord1	(0x0000) 0	HOLD
	Видимость тегов	1K.DWord2	(0x0000) 0	HOLD

Рисунок 6-4

Выберем <u>операцию</u> Изменить адрес, укажем <u>способ изменения</u> – По типам, <u>базовый</u>

<u>адрес</u> укажем **– 0**.

🔐 Групповые операции						_ 0	X
Изменить адрес							
Теги	Адрес (1	0) Адрес (16	5)				
Image: Construct of the system Image: Construct of the system Image: Construct of the system Image: Construct of the system Image: Construct of the system Image: Construct of the system Image: Construct of the system Image: Construct of the system Image: Construct of the system Image: Construct of the system Image: Construct of the system Image: Construct of the system Image: Construct of the system Image: Construct of the system Image: Construct of the system Image: Construct of the system Image: Construct of the system Image: Construct of the system Image: Construct of the system Image: Construct of the system Image: Construct of the system Image: Construct of the system Image: Construct of the system Image: Construct of the system Image: Construct of the system Image: Construct of the system Image: Construct of the system Image: Construct of the system Image: Construct of the system Image: Construct of the system Image: Construct of the system Image: Construct of the system Image: Construct of the system Image: Construct of the system Image: Construct of the system Image: Construct of the system Image: Construct	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0000 0000 0000 0000 0000 0000 0000 0000 0000					
Операция		Изменить адрес					
О Копировать		🗏 Настройки из	вменения				
О Бырезать		Тип изменения	a 👘	По типам			
		Базовый адре	C	0			
Изменить перестановку байтов		Смещение		1			
О Изменить регион							
Изменить тип тега в устройстве							
🔘 Изменить тип тега в сервере						Примен	ИТЬ
		L <u></u>			Готово	Отм	ена

Рисунок 6-5

Нажмем кнопку Применить.

🎎 Групповые операции					
Изменить адрес					
Теги → ✓ Ф Tags → ✓ Порт.ПЛК.Word1 → ✓ Порт.ПЛК.Word2 → ✓ Порт.ПЛК.Word3 → ✓ Порт.ПЛК.Word4 - ✓ ♥ Порт.ПЛК.Word5 → ✓ ♥ Порт.ПЛК.Word6 → ✓ ♥ Порт.ПЛК.Word7	Адрес (10 0 1 2 3 4 5 6	0) Addec (16 0000 0001 0002 0003 0004 0005 0006	5)		
 ✓ Порт.ПЛК.Float1 ✓ Порт.ПЛК.Float2 ✓ Порт.ПЛК.Float3 ✓ Порт.ПЛК.Float4 ✓ Порт.ПЛК.DWord1 ✓ Порт.ПЛК.DWord2 	7 9 11 13 15 17	0007 0009 000B 000D 000F 0011			
Операция		Изменить адрес			
🔘 Копировать		⊟Настройки из	менения		
Вырезать		Тип изменения	1	По типам	
Удалить		Базовый адрес	0	0	
 Изменить перестановку байтов 		Смещение		1	
🔘 Изменить регион					
🔘 Изменить тип тега в устройстве					
🔘 Изменить тип тега в сервере					применить
		·			Готово Отмена

Теги получили последовательно идущие адреса с шагом в зависимости от типа (<u>*Рисунок*</u> <u>6-6</u>).

Однако 4 байтовые теги (*Float* и *DWord* переменные) попали на нечетные адреса, в то время как по правилам 4 байтовые переменные должны находится в адресах памяти кратных 4. То есть Modbus адреса 4 байтовых переменных должны начинаться с четного числа. Исправим эту ошибку.

Нажмем кнопку *Готово*. Затем у устройства, на <u>закладке</u> **Таблица тегов** выделим все 4байтовые переменные, вызовем <u>контекстное меню</u>, а затем команду *Групповые операции* (<u>Рисунок 6-7</u>).

Теги						
Имя	Адрес	Регион	Тип в серве	Тип в устройс		
Порт.ПЛК.Word1	(0x0000) 0	HOLDING_REGI	uint32	uint16		
Порт.ПЛК.Word2	(0x0001) 1	HOLDING_REGI	uint32	uint16		
Порт.ПЛК.Word3	(0x0002) 2	HOLDING_REGI	uint32	uint16		
Порт.ПЛК.Word4	(0x0003) 3	HOLDING_REGI	uint32	uint16		
Порт.ПЛК.Word5	(0x0004) 4	HOLDING_REGI	uint32	uint16		
Порт.ПЛК.Word6	(0x0005) 5	HOLDING_REGI	uint32	uint16		
Порт.ПЛК.Word7	(0x0006) 6	HOLDING_REGI	uint32	uint16		
Порт.ПЛК.Float1	(0x0007) 7	HOLDING_REGI	float	float		
Порт.ПЛК.Float2	(0x0009) 9	HOLDING_REGI	float	float		
Порт.ПЛК.Float3	(0x000B) 11	HOLDING_REGI	float	float		
Порт.ПЛК.Float4	(0x000 Выделить	GI	float	float		
Порт.ПЛК.DWord1	(0x000	GI	uint32	uint32		
Порт.ПЛК.DWord2	(0x001-,	GI	uint32	uint32		
•	111			4		
Свойства объекта Табл	Свойства объекта Таблица тегов					

Переменная *Float* начинается с адреса 7. Укажем в качестве <u>базового адреса</u> следующее

четное число, то есть **8** (<u>Рисунок 6-8</u>).

🎎 Групповые операции		
Изменить адрес		
Теги Адрес	(10) Адрес (16)	
Гадз Г Г Порт.ПЛК.Float1 Г Порт.ПЛК.Float2 Г Порт.ПЛК.Float3 Порт.ПЛК.Float4 Порт.ПЛК.Float4 Порт.ПЛК.Float4 Порт.ПЛК.Float4 Порт.ПЛК.Float4 Порт.ПЛК.DWord1 Г Порт.ПЛК.DWord2	0007 0009 000B 000D 000F 0011	
Операция	Изменить адрес	
Копировать	Настройки изменения	
🔘 Вырезать	Тип изменения	По типам
🔘 Удалить	Базовый адрес	8
• Изменить адрес	Смещение	1
Изменить перестановку байтов		
Изменить регион		
О Изменить тип тега в сервере		Применить
		Готово Отмена

Рисунок 6-8

Нажмем кнопку Применить.

🌺 Групповые операции		
Изменить адрес		
Теги Адрес	(10) Адрес (16)	
Гадз Порт.ПЛК.Float1 8 Горт.ПЛК.Float2 10 Порт.ПЛК.Float3 12 Порт.ПЛК.Float4 14 Порт.ПЛК.Dword1 16 Порт.ПЛК.Dword2 18	0008 000A 000C 000E 0010 0012	
Операция	Изменить адрес	
О Копировать		
🔘 Вырезать	Тип изменения	По типам
🔘 Удалить	Базовый адрес	8
Изменить адрес	Смещение	1
🔘 Изменить перестановку байтов		
О Изменить регион		
О Изменить тип тега в устройстве		Примонить
🔘 Изменить тип тега в сервере		применитв
		Готово Отмена

Теперь все адреса корректные (*Рисунок 6-9*). Нажмем на кнопку *Готово*.

Теги				
Имя	Адрес	Регион	Тип в серве	Тип в устройс.
Порт.ПЛК.Word1	(0x0000) 0	HOLDING_REGI	uint32	uint16
Порт.ПЛК.Word2	(0x0001) 1	HOLDING_REGI	uint32	uint16
Порт.ПЛК.Word3	(0x0002) 2	HOLDING_REGI	uint32	uint16
Порт.ПЛК.Word4	(0x0003) 3	HOLDING_REGI	uint32	uint16
Порт.ПЛК.Word5	(0x0004) 4	HOLDING_REGI	uint32	uint16
Порт.ПЛК.Word6	(0x0005) 5	HOLDING_REGI	uint32	uint16
Порт.ПЛК.Word7	(0x0006) 6	HOLDING_REGI	uint32	uint16
Порт.ПЛК.Float1	(0x0008) 8	HOLDING_REGI	float	float
Порт.ПЛК.Float2	(0x000A) 10	HOLDING_REGI	float	float
Порт.ПЛК.Float3	(0x000C) 12	HOLDING_REGI	float	float
Порт.ПЛК.Float4	(0x000E) 14	HOLDING_REGI	float	float
Порт.ПЛК.DWord1	(0x0010) 16	HOLDING_REGI	uint32	uint32
Порт.ПЛК.DWord2	(0x0012) 18	HOLDING_REGI	uint32	uint32
		-		
•		1		•
	Имя Порт.ПЛК.Word1 Порт.ПЛК.Word2 Порт.ПЛК.Word3 Порт.ПЛК.Word4 Порт.ПЛК.Word5 Порт.ПЛК.Word6 Порт.ПЛК.Word7 Порт.ПЛК.Float1 Порт.ПЛК.Float2 Порт.ПЛК.Float3 Порт.ПЛК.Float3 Порт.ПЛК.Float4 Порт.ПЛК.DWord1 Порт.ПЛК.DWord2	Имя Адрес Порт.ПЛК.Word1 (0x0000) 0 Порт.ПЛК.Word2 (0x0001) 1 Порт.ПЛК.Word3 (0x0002) 2 Порт.ПЛК.Word4 (0x0003) 3 Порт.ПЛК.Word5 (0x0004) 4 Порт.ПЛК.Word6 (0x0005) 5 Порт.ПЛК.Word7 (0x0006) 6 Порт.ПЛК.Float1 (0x00002) 12 Порт.ПЛК.Float3 (0x00002) 12 Порт.ПЛК.Float4 (0x0002) 14 Порт.ПЛК.DWord1 (0x0012) 18	Имя Адрес Регион Порт.ПЛК.Word1 (0x0000) 0 HOLDING_REGI Порт.ПЛК.Word2 (0x0001) 1 HOLDING_REGI Порт.ПЛК.Word3 (0x0002) 2 HOLDING_REGI Порт.ПЛК.Word4 (0x0003) 3 HOLDING_REGI Порт.ПЛК.Word5 (0x0004) 4 HOLDING_REGI Порт.ПЛК.Word6 (0x0005) 5 HOLDING_REGI Порт.ПЛК.Word7 (0x0006) 6 HOLDING_REGI Порт.ПЛК.Float1 (0x0008) 8 HOLDING_REGI Порт.ПЛК.Float2 (0x000A) 10 HOLDING_REGI Порт.ПЛК.Float3 (0x000C) 12 HOLDING_REGI Порт.ПЛК.Float4 (0x000E) 14 HOLDING_REGI Порт.ПЛК.DWord1 (0x0012) 18 HOLDING_REGI	Имя Адрес Регион Тип в серве Порт.ПЛК.Word1 (0x0000) 0 HOLDING_REGI uint32 Порт.ПЛК.Word2 (0x0001) 1 HOLDING_REGI uint32 Порт.ПЛК.Word3 (0x0002) 2 HOLDING_REGI uint32 Порт.ПЛК.Word4 (0x0003) 3 HOLDING_REGI uint32 Порт.ПЛК.Word5 (0x0004) 4 HOLDING_REGI uint32 Порт.ПЛК.Word6 (0x0005) 5 HOLDING_REGI uint32 Порт.ПЛК.Word7 (0x0008) 8 HOLDING_REGI uint32 Порт.ПЛК.Float1 (0x0000) 0 HOLDING_REGI float Порт.ПЛК.Float3 (0x0000) 12 HOLDING_REGI float Порт.ПЛК.Float4 (0x0002) 12 HOLDING_REGI float Порт.ПЛК.Float4 (0x0010) 16 HOLDING_REGI float Порт.ПЛК.DWord1 (0x0012) 18 HOLDING_REGI uint32

Рисунок 6-10



Примечание. Проект Codesys v2.3 для контроллера OBEH ПЛК100-Р.М с полным кодом данного примера, а также конфигурация OPC сервера приложены к документации.