

Контроллеры промышленные серии ME200

Руководство по эксплуатации

Редакция от февраля 2025 года

optimusdrive.ru

Оглавление

Введение 3
Меры предосторожности при эксплуатации 4
Функциональное назначение 5
Перечень оборудования 6
Спецификация ЦПУ (контроллеров) 7
Спецификация дискретных входов-выходов на ЦПУ 8
Внешний вид и размеры ЦПУ 9
Расположение клемм ЦПУ 11
Схемы подключения ME204-C 12
Расположение интерфейсов ME204-C 16
Спецификация источника питания 18
Установка модулей расширения 19
Спецификация модулей дискретных входов-выходов 21
Внешний вид и размеры дискретных модулей расширения 22
Расположение клемм дискретных модулей расширения 23
Схемы подключения дискретных входов-выходов модулей расширения 27
Спецификация модулей аналоговых входов-выходов 31
Внешний вид и размеры аналоговых модулей 33
Расположение клемм аналоговых модулей 34
Схемы подключения аналоговых входов-выходов 35
Запуск среды программирования и создание проекта 37
Установка описания устройства для контроллера ME200 38
Определение версии библиотеки 3S SoftMotion (SM3) 44
Добавление контроллера в проект 45
Организация связи контроллеров типа ME200 и среды программирования. Загрузка программы. Онлайн режим 46
Изменение IP адреса контроллера из программы контроллера 56
Изменение IP адреса контроллера из среды программирования 62
Использование встроенных входов-выходов контроллера в обычном режиме 63
Добавление в проект модулей расширения 67
Поддерживаемые базовые типы данных 78
Список наиболее употребительных команд 79
Работа с высокоскоростными счётчиками 90
Импульсная ось движения 97
Последовательная связь по протоколу Modbus RTU Master 102
Последовательная связь по протоколу Modbus RTU Slave 113
Связь по протоколу Modbus TCP Master 119
Связь по протоколу Modbus TCP Slave 128
Связь по протоколу Ethernet/IP Scanner (Master) 134
Связь по протоколу Ethernet/IP Adapter (Slave) 154
Чтение и установка часов реального времени 166
Прерывание по сигналу на входе контроллера 169
Импульс при переводе в состояние RUN 172
Высокоскоростное сравнение 174
Работа в режиме ШИМ импульсов 179
Замена батарейки часов реального времени 180
Обмен тегами CODESYS с панелью оператора 182

Введение

Настоящее Руководство описывает порядок и особенности использования контроллеров серии ME200 в среде программирования DesignerAX 1.7 и выше или CODESYS 3.5.18.30 и содержит информацию по техническим характеристикам контроллеров и модулей расширения, а также освещаются вопросы связанные с организацией работы с аппаратными ресурсами контроллера и модулей такие как: работа с дискретными и аналоговыми входами-выходами, высокоскоростными счётчиками, добавление в проект импульсной оси. Работы с портами по различным протоколам связи и др.

Редакция от февраля 2025 года содержит следующие сведения:

- технические характеристики контроллеров и модулей расширения
- установка конфигурационных файлов
- подключение контроллера к ПК
- работа с модулями ввода-вывода
- работа с высокоскоростными счётчиками
- организация импульсной оси
- работа по протоколу Modbus
- работа по протоколу Ethernet/IP
- установка IP адреса
- экспорт тегов CODESYS
- организация прерываний по входу
- высокоскоростное сравнение
- замена батарейки

и другие вопросы.

За информацией по работе в самой среде разработки CODESYS обращайтесь на сайт компании разработчика данного программного обеспечения.

Меры предосторожности при эксплуатации

Контроллеры семейства ME200 и модули расширения к ним предназначены для использования только квалифицированным персоналом!
Перед началом эксплуатации внимательно ознакомьтесь с настоящим Руководством!

Меры предосторожности при монтаже и установке

⚠ ВНИМАНИЕ

- Не наступайте на контроллер (модули расширения) и не кладите на него тяжелые предметы
- Не блокируйте вентиляционные отверстия и не допускайте попадания в них посторонних частиц
- Контроллер и модули можно устанавливать только вертикально с обеспечением свободного пространства не менее 50 мм со всех сторон. В шкафу должна быть обеспечена свободная конвенция воздуха
- Не подвергайте контроллер и модули ударам
- Контроллер и модули имеют степень защиты IP20 и не являются водонепроницаемым. Примите меры, чтобы предотвратить попадание воды и т.п. внутрь контроллера и модулей
- Контроллер и модули предназначены для установки только в общую защитную оболочку (шкаф управления). Эксплуатация в открытом виде запрещена

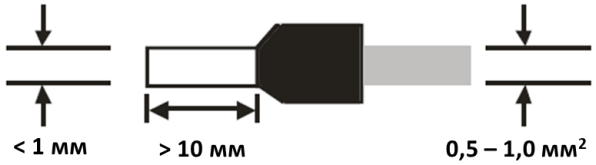
Меры предосторожности при подключении и работе

⚠ ВНИМАНИЕ

- Подключите кабели заземления во избежание поражения электрическим током и пожара, а также должной работе экранов кабелей связи
- Напряжение питания контроллера и модулей строго 24 VDC. Используйте только стабилизированные источники питания достаточной мощности. Учитывайте при выборе источника питания пусковые токи. Категорически запрещается подключение к контроллеру переменного напряжения. Нарушение данного требования однозначно выведет контроллер (модули) из строя
- Проверьте затяжку винтов клемм, неполная затяжка может привести к возгоранию
- Во избежание несчастных случаев и выхода из строя оборудования обратите внимание на правильность подключения кабелей

Рекомендации по использованию проводов и наконечников

На контроллерах ME200 и модулях расширения установлены пружинные клеммники, которые позволяют производить монтаж проводов без наконечников. Рекомендуются строго гибкие (многопроволочные) провода сечением 0,5 – 1,0 мм^2.
В случае использования наконечников рекомендуются наконечники без юбки. Если Вы принципиально используете наконечники с юбкой, то длина рабочей части наконечника должна быть 11 – 14 мм (оптимально 12 мм). Диаметр после обжима не более 1 мм.



Функциональное назначение

Приборы семейства ME200 являются общепромышленными контроллерами, предназначенными для решения широкого круга задач автоматизации технологических процессов с программированием из среды на базе CODESYS.
Также, контроллеры серии ME200 могут применяться для задач управления движением классическим импульсным способом с умеренной скоростью процессов. Обладают высокими программными возможностями с поддержкой сложных вычислений, логики и операций с данными. Имеют встроенные порты Ethernet, RS232, RS485, USB C. Поддерживают протоколы связи Modbus RTU/TCP и Ethernet/IP.

Для программирования контроллеров семейства ME200 рекомендуется среда разработки Designer-AX 1.7 и выше версией, которую можно получить по запросу в компании Оптимус Драйв.
Возможно использование среды программирования CODESYS 3.5.18.30. Но в данном случае пользователь контроллера изыскивает её самостоятельно.

Designer-AX предоставляет следующие инструменты для разработки проекта:

- Языки программирования стандарта IEC 61131-3: LD, ST, CFC, SFC и FBD
- Программные объекты типа POU, FB, FC, Interface, DUT, Task и др.
- Поддержка большого количества типов данных
- Большая библиотека прикладных команд для различных применений
- Всплывающие подсказки при вводе и настройке
- Развитый интерфейс программирования и настройки
- Различные инструменты отладки, симулятор, онлайн режим, правка программы в онлайн
- Многоуровневая защита исходного кода проекта
- Поддержка устройств разных производителей

Функции управления движением:
- CODESYS SM3_Basics/Robotics*/CNC v.4.16 (4.10)
- LS Motion Lib для интерполированного движения,
- динамический E-CAM, EGear, диагностика, мониторинг данных
- Поддержка физических и виртуальных осей;
- Графический редактор E-CAM;

** Библиотека Robotics поддерживается ограничено, так как у контроллера одноядерный процессор*

IP адрес по умолчанию: **192.168.1.3**
USB драйвер и XML файлы модулей расширения устанавливаются в составе пакета установки.

Система состоит из Центрального Процессорного Устройства (ЦПУ), т.е. контроллера и модулей расширения дискретных и аналоговых входов-выходов. Далее приводится перечень оборудования, входящего в состав системы на основе контроллеров ME200, а также приводятся характеристики аппаратной части Центральных Процессорных Устройств (ЦПУ) и модулей расширения.

Требования к системе:
Windows 10/11 64 бит
Процессор: минимум Intel Core i5 M520 2.4 ГГц
Оперативная память: минимум 8 Гб (оптимально 16 Гб)
Microsoft .Net Framework: не ниже 4.6.2 (оптимально 4.8)
Среда программирования: DIADesignerAX 1.7 и выше
или CODESYS 3.5 SP18 64 бит (32 бит не поддерживается)

Перечень оборудования

Наименование	Обозначение	Описание
Контроллеры (ЦПУ)	ME204-C	Контроллер, 16DI/16DO (4 AB счётчиков/4 AB импульсных каналов 200 кГц), Ethernet, USB C, RS232, RS485, 24 VDC
Модули дискретных входов/выходов	ME16DI	24 В DC 5 мА 16 входов NPN/PNP Пружинный клеммный блок
	ME16DOP	5 ~ 30 В DC 0.5 А 16 выходов Выходы: PNP Пружинный клеммный блок
	ME16DOR	240 В AC / 24 В DC 2 А 16 выходов Выходы: Реле Пружинный клеммный блок
	ME16DOT	5 ~ 30VDC 0.5A 16 выходов Выходы: NPN Пружинный клеммный блок
Модули аналоговых входов-выходов	ME04AD	4-канальный модуль аналоговых входов Аппаратное разрешение: 16 бит 0~10 В, 0/1~5 В, -5~+5 В, -10~+10 В, 0/4~20 мА, -20~+20 мА Время преобразования: 2 мс/канал
	ME04DA	4-канальный модуль аналоговых выходов Аппаратное разрешение: 12 бит 0~10 В, 0/1~5 В, -5~+5 В, -10~+10 В, 0/4~20 мА Время преобразования: 2 мс/канал

Примечание: Термин «ось движения» означает, что данный контроллер позволяет осуществлять скоординированное движение, т.е. группировать оси для совместного движения, например линейной и круговой интерполяции, E-CAM, GEAR и т.п.

Спецификация ЦПУ (контроллеров)

Модель	ME204-C
Кол-во поддерживаемых осей	Импульсное управление 4 осями 200 кГц
Процессор	ARM Cortex-A8 Single Core 1 ГГц
Макс. кол-во модулей расширения	16
EtherNET	1* EtherNET port, Modbus TCP, Socket, загрузка и выгрузка программы, отладка
Порт последовательной связи	RS232, RS485, пользовательский протокол, MODBUS RTU Ведущий/Ведомый
Память под программу	8 Мб
Память данных	16 Мб
Энергонезависимая память	256 Кб
Порт USB	Тип С, загрузка и выгрузка программы, отладка
Управление движением	Точка-точка, электронный кулачок (E-CAM), интерполяция
Высокоскоростные счетчики	4 группы AB (200 кГц)
Встроенные входы/выходы	16 входов NPN/PNP (8 входов 200 кГц/8 входов 1 кГц) 16 выходов (8 выходов 200 кГц/8 выходов 10 кГц (NPN)) выхода 14 и 15 могут работать в режиме ШИМ
Часы реального времени	Да (встроенная батарейка CR1632 с выводами и разъёмом)
Среда разработки	Designer-AX 1.7, CODESYS 3.5.18.30
Языки программирования	ST, LD, CFC, SFC, FBD
Библиотеки	SM3_Basics/Robotics*/CNC v.4.10 (4.16)
Напряжение питания	24 В постоянного тока (DC20.4～28.8V)
Потребляемая мощность	Номинально 3 Вт (пусковой ток >1 А)
Рабочая температура	0~50℃ (номинально не более 40℃)
Относительная влажность	40 – 90 % без образования конденсата
Вибрация	Не более, чем 0,15 мм на частотах 10 - 55 Гц
Габаритные размеры	ВхШхГ: 105.00 x 81.00 x 85,00 мм

* Библиотека Robotics может быть использована только с сильно ограниченным функционалом, так как процессор одноядерный

Спецификация дискретных входов-выходов на ЦПУ

Спецификация дискретных входов:

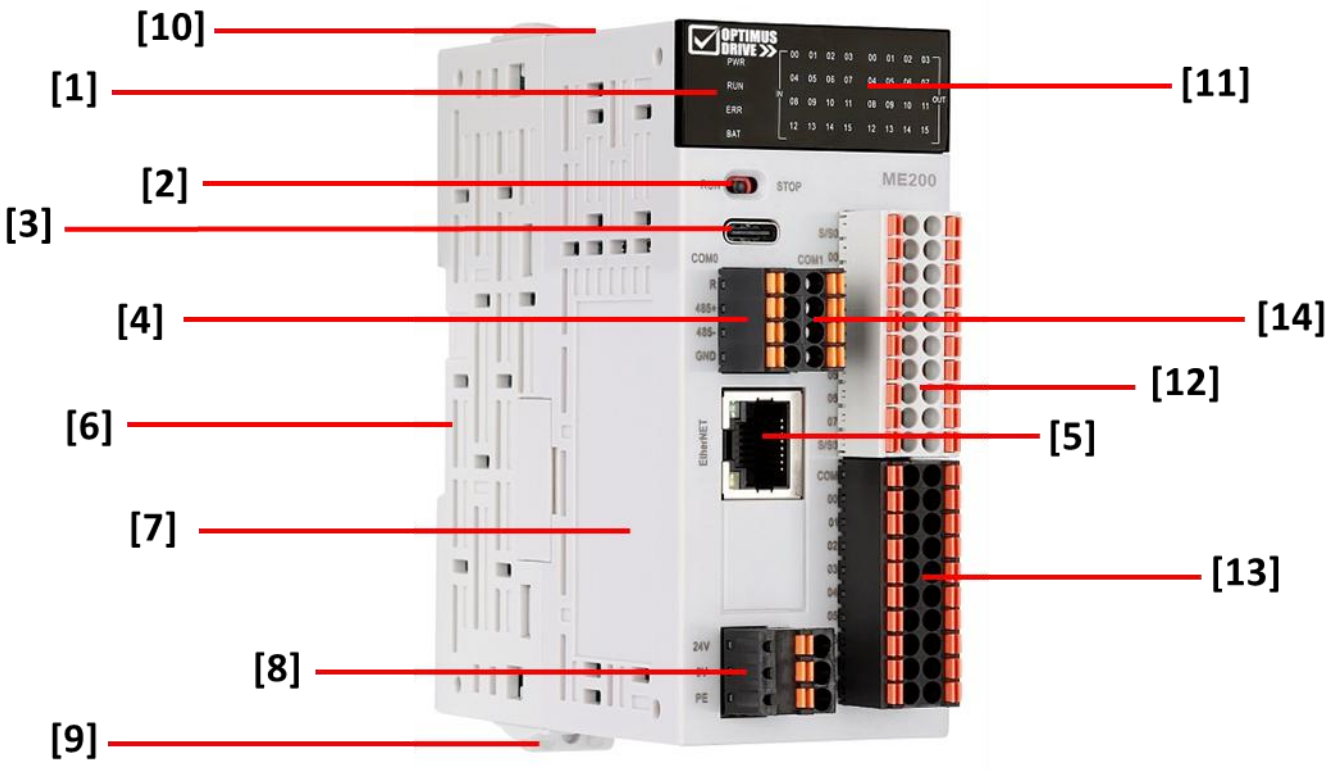
Параметр / Модель		ME204-C
Количество входов		16
Тип соединения		Съёмный пружинный клеммник
Тип входа		Дискретный вход
Форма входа		Постоянный ток (NPN/PNP, две общие точки SS0/SS1) SS0 для IN0~IN7, SS1 для IN8~IN15
Напряжение/ ток		24 VDC, 5 mA
Уровень вкл/выкл	OFF→ON	>15 VDC
	ON→OFF	<5 VDC
Максимальная входная частота		200 KHz
Входное сопротивление		4.7 kΩ обычные входы, 2.7 kΩ импульсные входы
Тип входного сигнала		Потенциальный сигнал Sinking: SS0/SS1 подключены к 24V+ Sourcing: SS0/SS1 подключены к 0V
Электрическая изоляция		Оптопары, фильтр 1-1000 мс
Индикация		Когда оптопара активна, светодиод входа включен

Спецификация дискретных выходов:

Параметр / Модель		ME204-C
Количество выходов		16
Тип соединения		Съёмный пружинный клеммник
Тип выхода		NPN (Sinking)
Напряжение		5~30 VDC
Макси-мальная нагрузка	Активная	0.3A/выход, 2.4A/8 выходов на одну общую точку
	Индуктивная	-
	Лампочка	-
Максимальная выходная частота		200 KHz
Общая точка COM		COM0: OUT0-OUT7; COM1: OUT8-OUT15

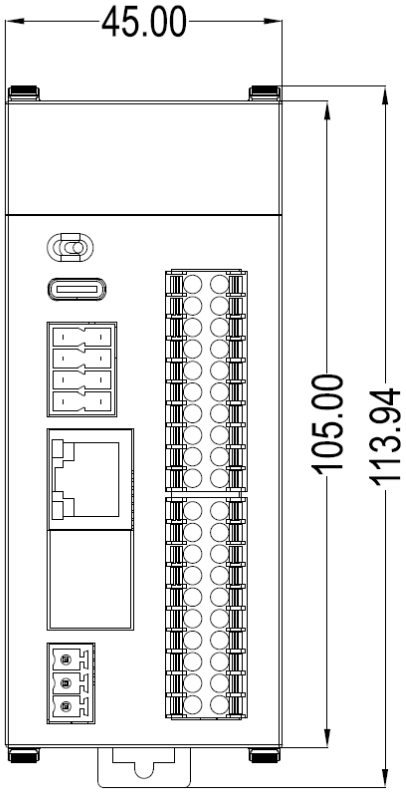
Внешний вид и размеры ЦПУ

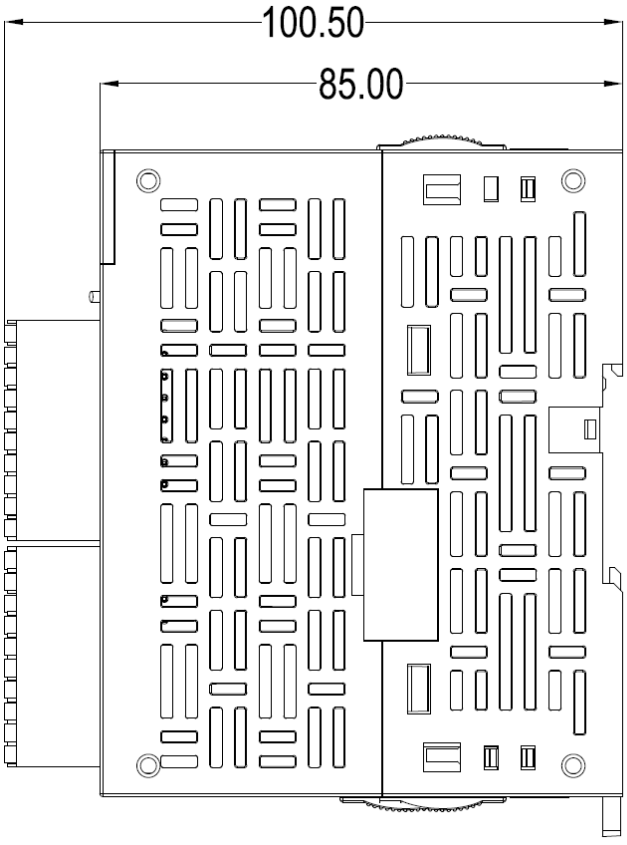
ME204-C



No.	Интерфейс	Маркировка	Назначение	Описание
[1]	**Светодиодная индикация**	PWR	Питание	Горит при поданном нормальном питании
		RUN	Работа	Горит в состоянии Работа, выкл. в Стопе
		ERR	Ошибка	Горит красным при наличии ошибки в работе
		BAT	Батарейка	Горит при разряде батарейки
[2]	**Переключатель RUN/STOP /RESET**	RUN	Работа	Если передёрнуть переключатель RUN/STOP 5 раз в течении 5 секунд, то IP адрес сбросится на заводской . 192.168.1.3
		STOP	Стоп	
[3]	**Порт USB**		Гнездо Type-C	Type-C для подключения к ПК Загрузка-выгрузка и мониторинг программы, подключение USB flash drive (формат FAT32, 32 Гб) для переноса приложения и обмена файлами
[4]	**Порт RS485**	R	Терминальный резистор RS485	Для подключения терминального резистора необходимо поставить перемычку между клеммами R и RS485+
		485+	RS485_A	Modbus RTU Свободный протокол (поставить библиотеку CAA SerialCom)
		485-	RS485_B	
		COM	GND	

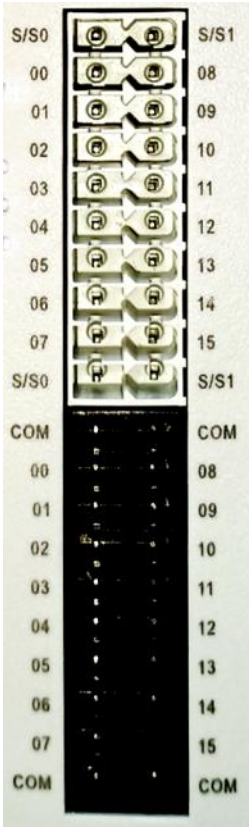
[5]	**Порт Ethernet**	Ethernet	Ethernet RJ45	Протоколы Modbus-TCP, Ethernet/IP Загрузка-выгрузка и мониторинг программы, обновление встроенного ПО (сервисная утилита)
[6]	**Installed**	--	Монтаж	Углубление для установки на DIN-рейку
[7]	**Label**	--	Шильдик	Место для заводского шильдика
[8]	**Клеммы питания**	24V	DC24V（+）	DC24V Power
		0V	0V（-）	
		PE	заземление	Защитное заземление сигнальных цепей
[9]	**Защёлка**	--	--	Защёлка для установки на DIN-рейку
[10]	**Батхолдер**	--	CR1632 (3V)	Питание часов реального времени. Срок 3 года.
[11]	**Индикация состояния входов-выходов**	In,Out 0~15	16 входов, 16 выходов	Светодиод включен при наличии сигнала на входе или активированном выходе
[12]	**Клеммы входов**	In 0~15	16 дискретных входов	In 0-7 входы 200 кГц In 8-15 входы 1 кГц
[13]	**Клеммы выходов**	Out0-15	16 дискретных выходов	Out 0-7 выходы 200 кГц Out 8-15 выходы 10 кГц
[14]	**Порт RS232**	.	точка	клемма не используется
		TXD	RS232_T	Modbus RTU Свободный протокол (поставить библиотеку CAA SerialCom)
		RXD	RS232_R	
		GND	RS232_G	





Расположение клемм ЦПУ

ME204-C



Входы 00 – 07 являются высокоскоростными, входы 08 – 15 являются обычными

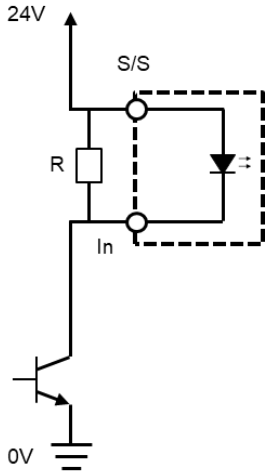
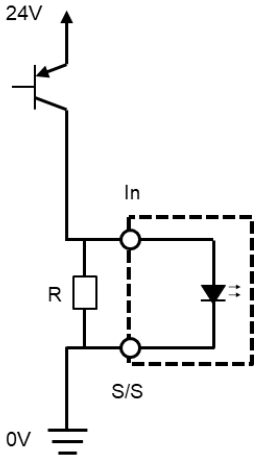
Выходы 00 – 07 являются высокоскоростными, выходы 08 -13 обычными, 14-15 для выдачи ШИМ (нечётный выход – направление, чётный – импульсы)

Внимание! При работе выхода на индуктивную нагрузку установка внешнего обратного диода на катушку является обязательной! В противном случае выход контроллера может выйти из строя уже при первом выключении индуктивной нагрузки (катушки реле/контактора).

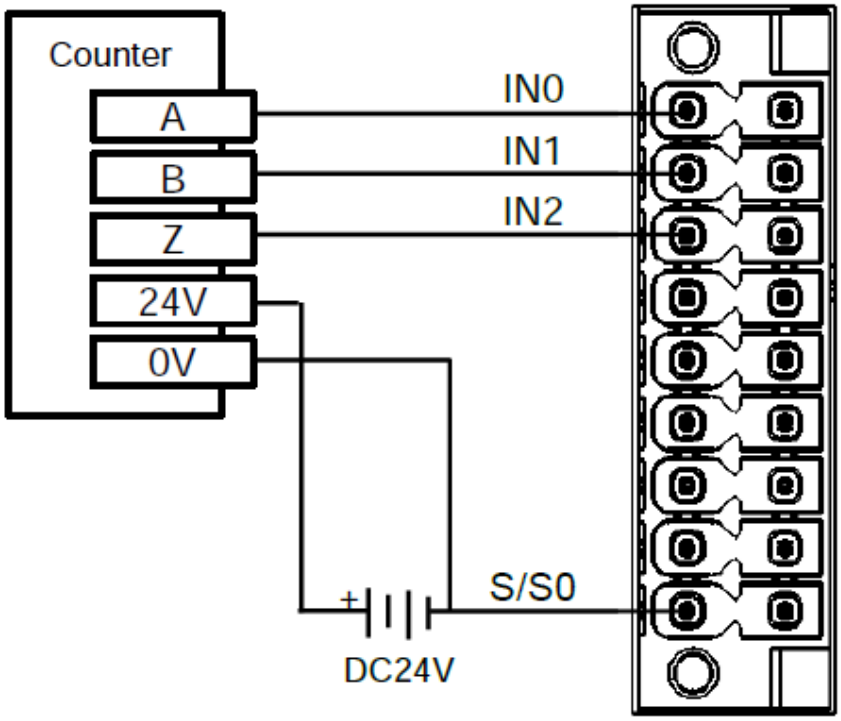
Схемы подключения ME204-C

Общие рекомендации:

1. При подключении к контроллеру источника высокочастотного сигнала типа открытый коллектор для достижения частоты 200 кГц необходимо подключить параллельно входу и точкой S/S резистор номиналом 3 Вт/470 Ом или 2 Вт/1 кОм. На схеме ниже обозначен как R. Вход обозначен как In.
2. Если в качестве источника сигнала используется тип push-pull, то резисторы не нужны.

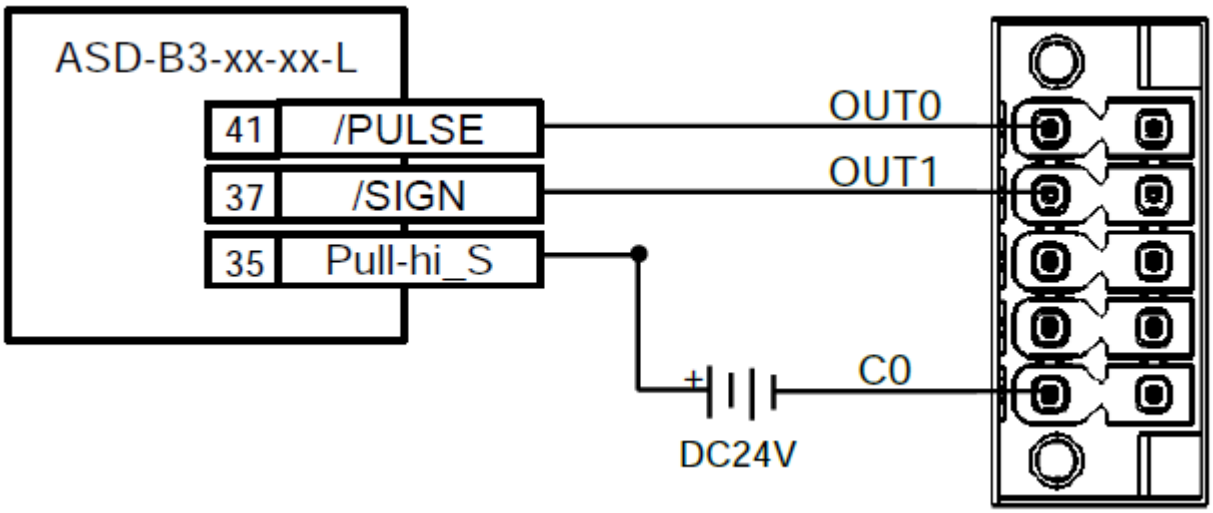
NPN (общий плюс)	PNP (общий минус)
	

Подключение входов в режиме высокоскоростного счётчика

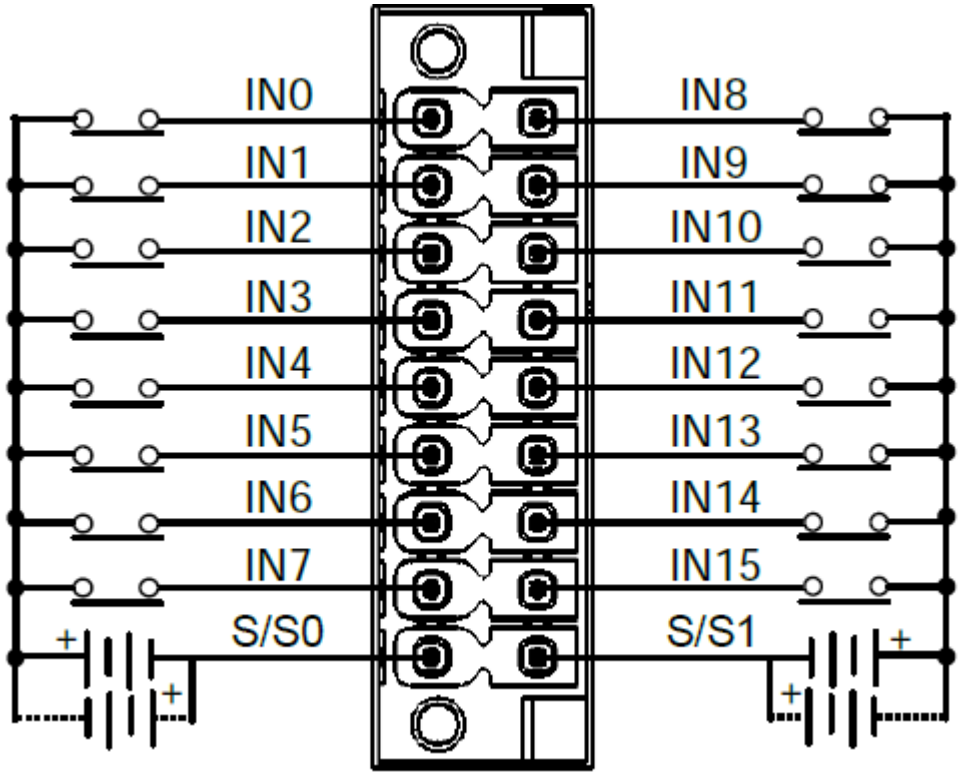


Подключение выходов в импульсном режиме

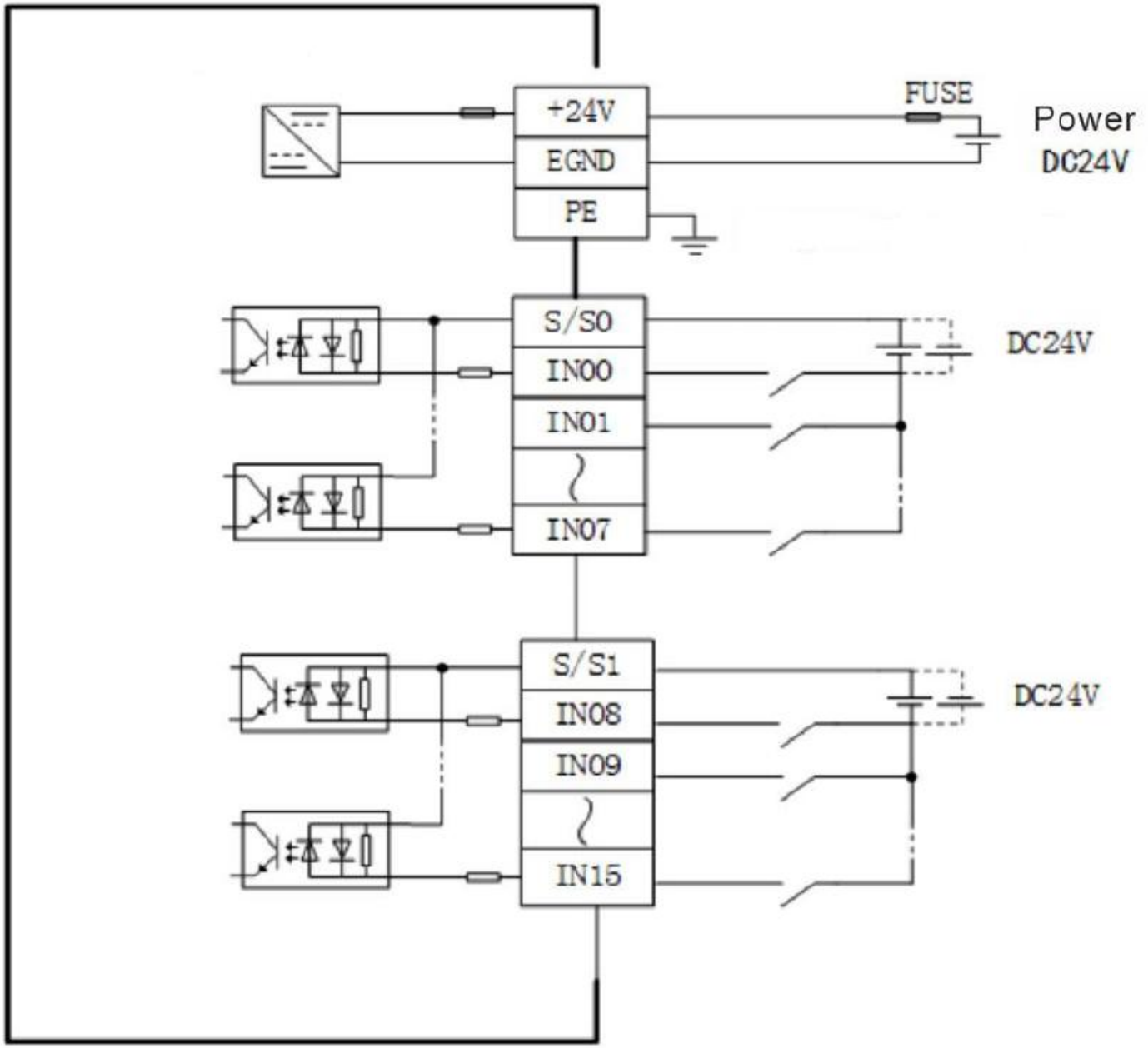
(на примере выдачи импульсов на сервопривод Delta ASD)



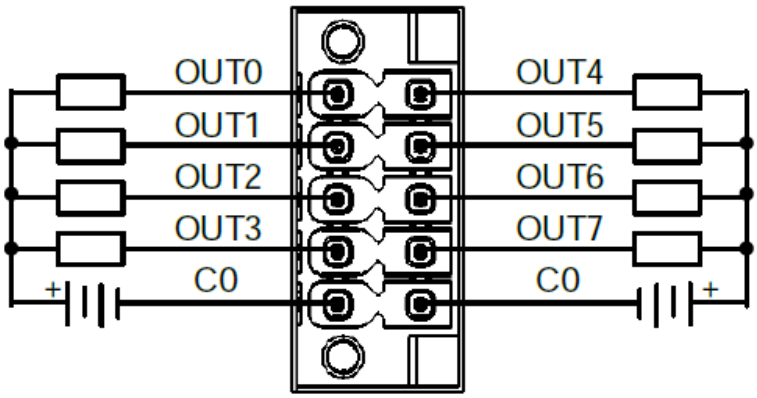
Подключение входов в обычном режиме



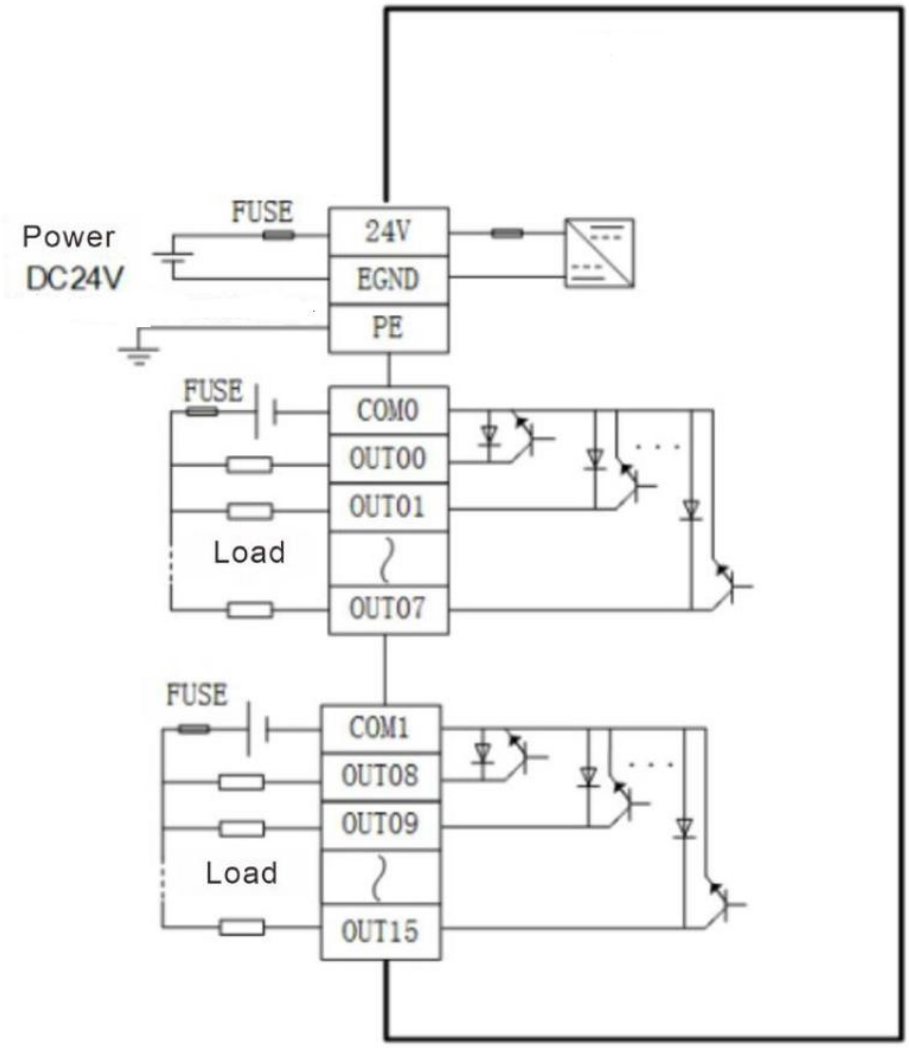
Эквивалентная схема входов:



Подключение выходов типа NPN в обычном режиме



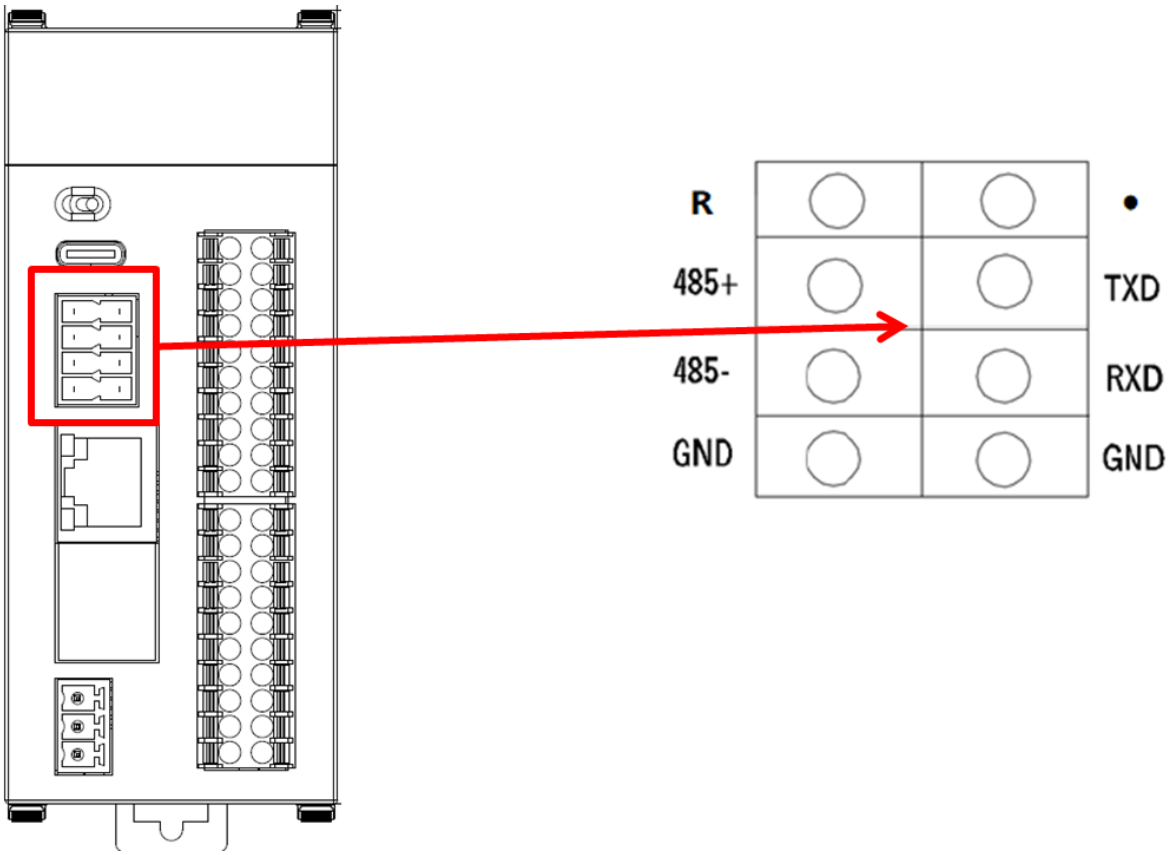
Эквивалентная схема выходов:

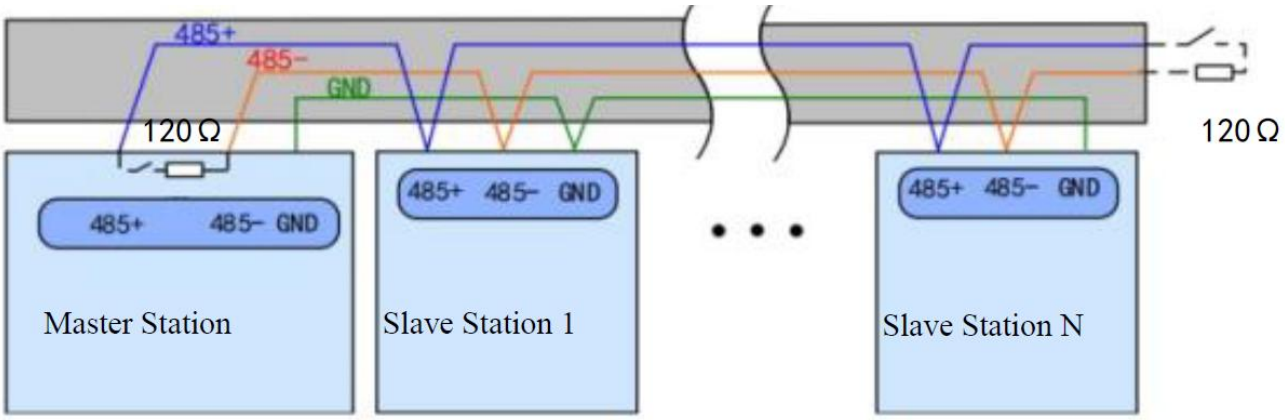


Внимание! При работе выхода на индуктивную нагрузку установка внешнего обратного диода на катушку является обязательной! В противном случае выход контроллера может выйти из строя уже при первом выключении индуктивной нагрузки (катушки реле/контактора).

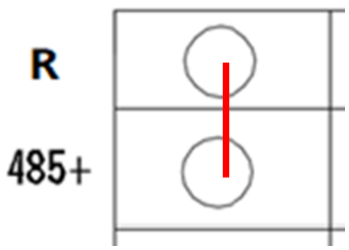
Расположение интерфейсов ME204-C

RS485/RS232

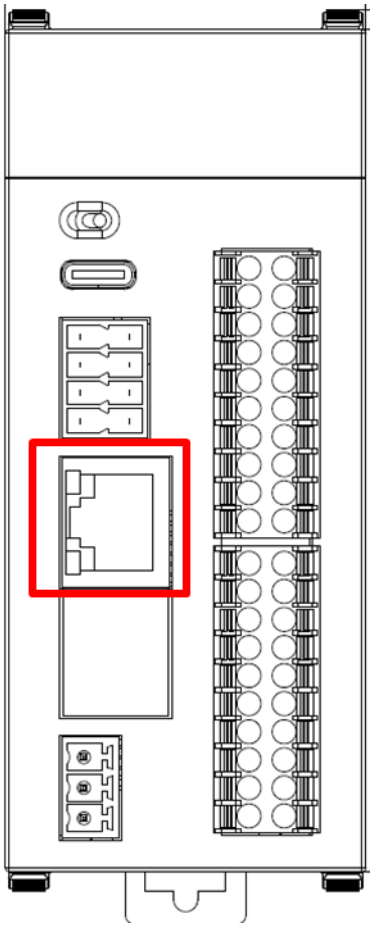




ЦПУ содержит встроенные резисторы 120 Ом для портов RS485 COM0 и COM1. Для подключения резистора необходимо перемкнуть перемычкой клеммы 485+ и R.



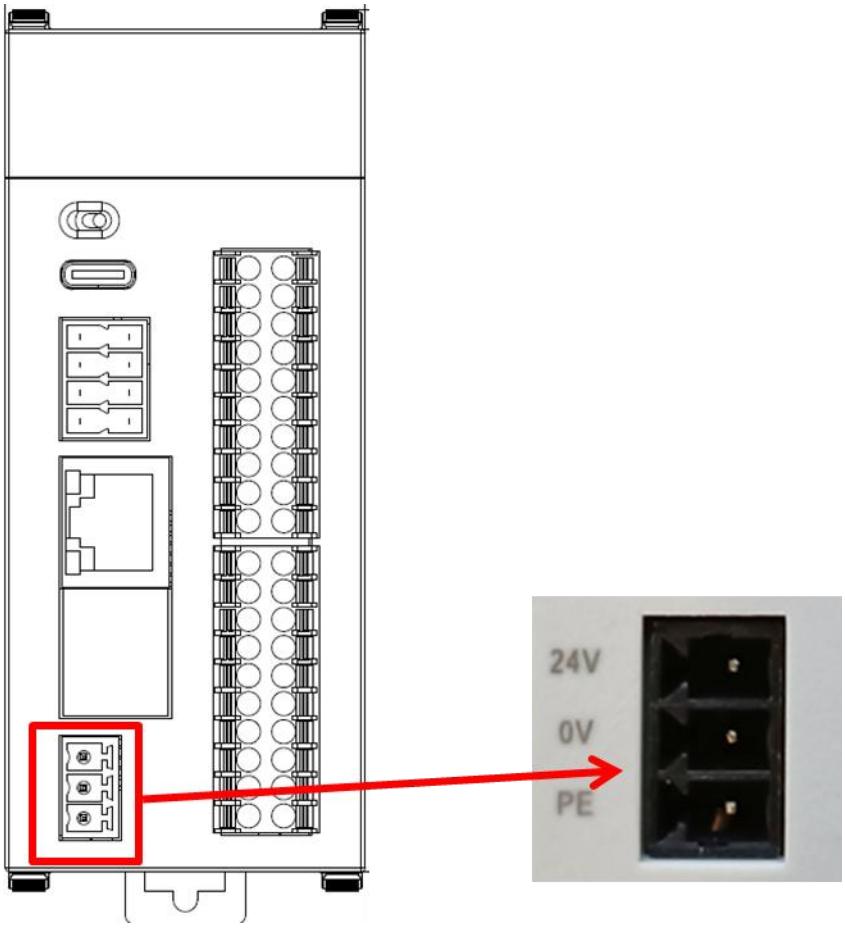
Ethernet



Контроллер может быть подключен к сети Ethernet напрямую или через коммутатор.

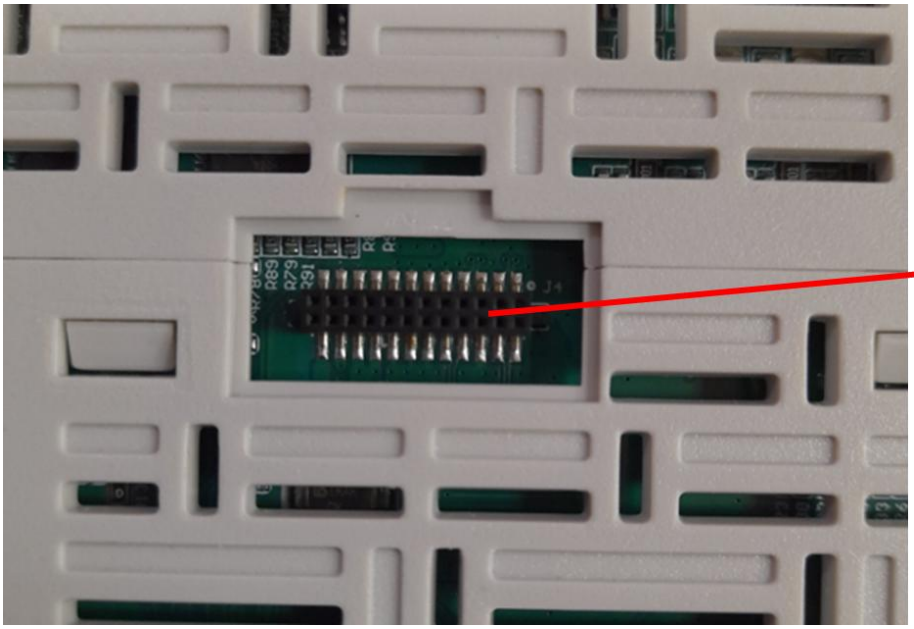
Спецификация источника питания

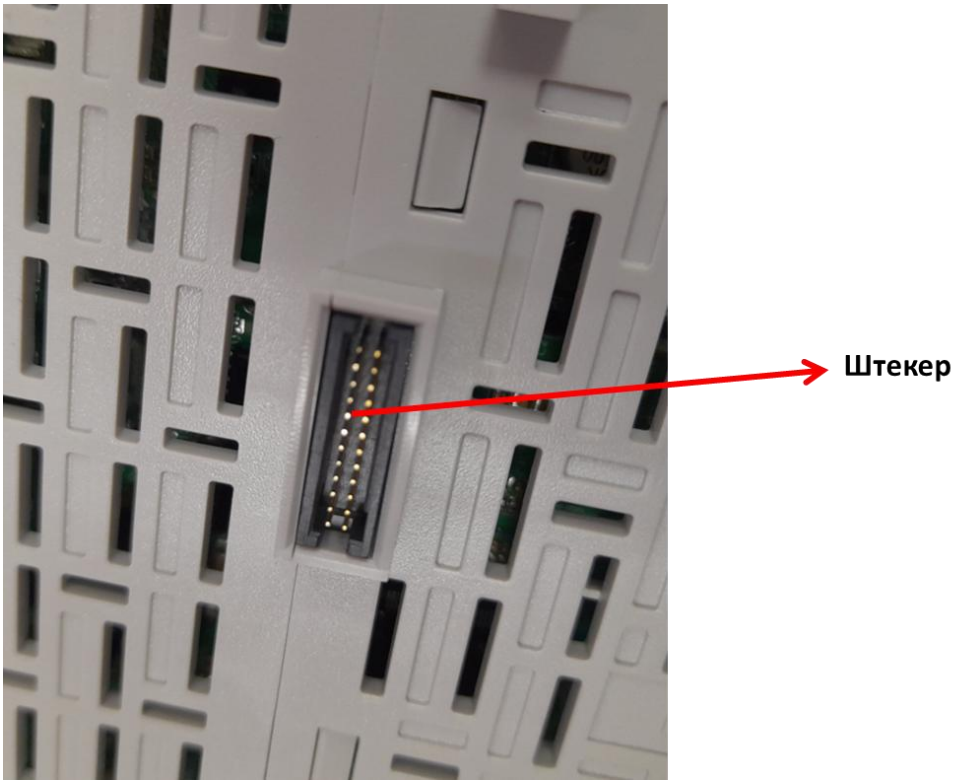
Контроллеры требуют питание 24 VDC от стабилизированного источника питания. В комплекте с контроллером идёт клеммник для подключения внешнего источника питания, мощность которого определяется количеством и составом подключенных к контроллеру модулей расширения. Питание подключается на соответствующие клеммы:



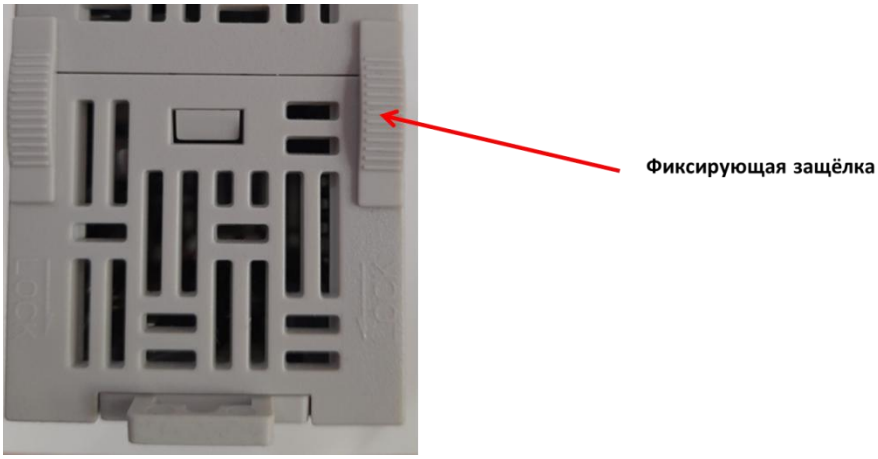
Установка модулей расширения

Модули расширения устанавливаются вплотную к ЦПУ по методу разъёмов штекре-гнездо и далее модуль к модулю.



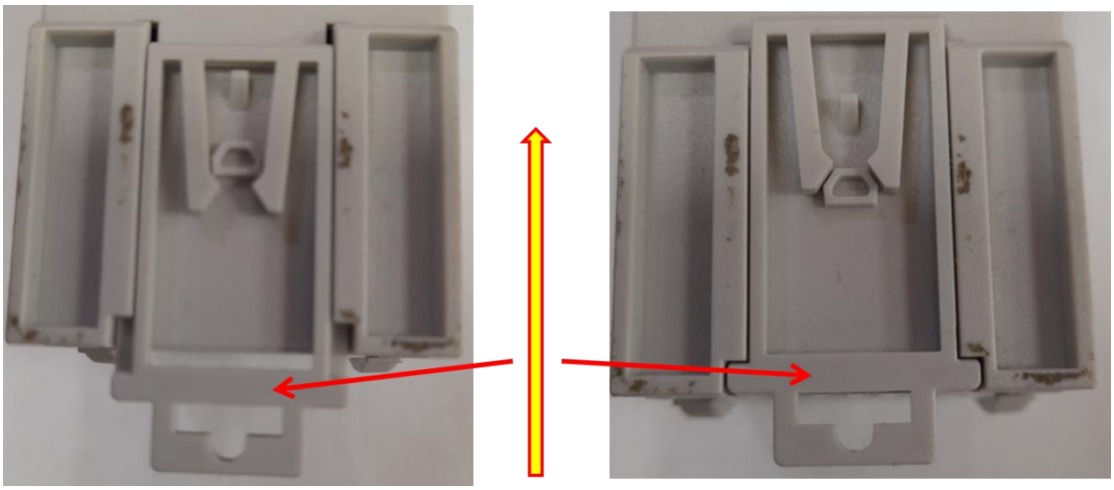


Далее модули необходимо зафиксировать защёлкой.



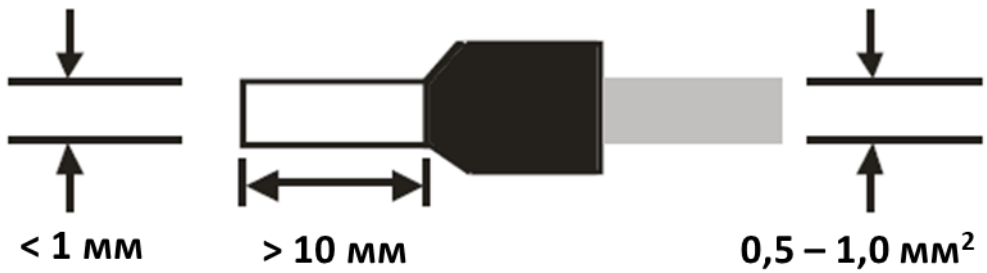
Для извлечения модуля из сборки необходимо раздвинуть модули.

На ДИН рейку ЦПУ и модули защёлкиваются при помощи фиксатора.



Рекомендации по использованию проводов и наконечников

На контроллерах ME200 и модулях расширения установлены пружинные клеммники, которые позволяют производить монтаж проводов без наконечников. Рекомендуются строго гибкие (многопроволочные) провода сечением 0,5 – 1,0 мм^2.
В случае использования наконечников рекомендуются наконечники без юбки. Если Вы принципиально используете наконечники с юбкой, то длина рабочей части наконечника должна быть не менее 11 мм (оптимально 12 мм). Диаметр после обжима не более 1 мм.



Спецификация модулей дискретных входов-выходов

Спецификация дискретных входов на модулях расширения:

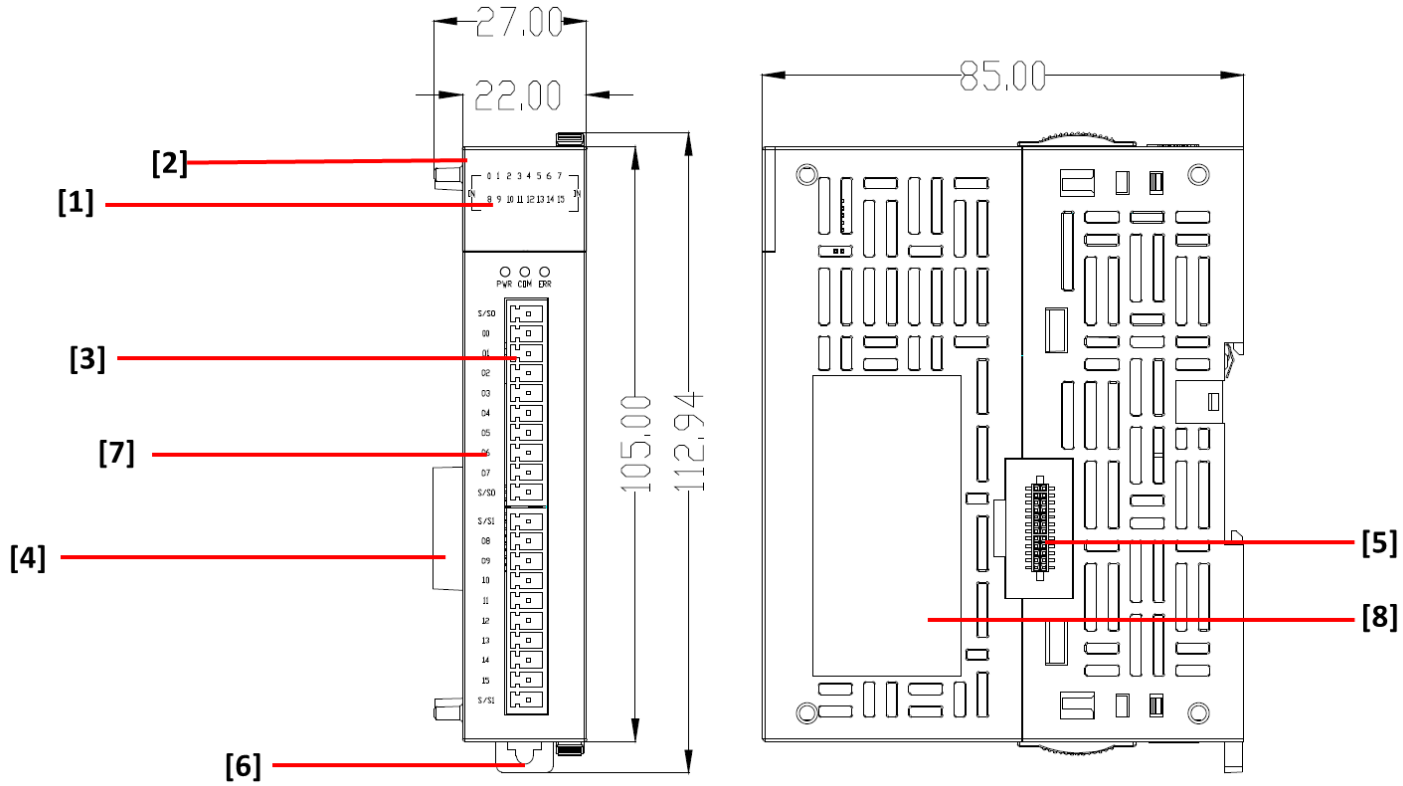
Параметр / Модель		ME16DI
Количество входов		16
Тип соединения		Съёмный пружинный клеммник
Тип входа		Дискретный вход
Форма входа		Постоянный ток (NPN/PNP, две общие точки SS0/SS1) SS0 для 00~07, SS1 для 08~15
Напряжение/ ток		24 VDC, 5 mA
Уровень вкл/выкл	OFF→ON	>15 VDC
	ON→OFF	<5 VDC
ON/OFF		20мкс/50мкс
Входное сопротивление		4.7 kΩ
Тип входного сигнала		Потенциальный сигнал Sinking: SS0/SS1 подключены к 24V+ Sourcing: SS0/SS1 подключены к 0V
Электрическая изоляция		оптопары
Индикация		Когда оптопара активна, светодиод входа включен
Потребляемая мощность		0,7 Вт
Габаритные размеры		ВхШхГ: 105.00 x 22.00 x 85.00 мм

Спецификация дискретных выходов на модулях расширения:

Параметр / Модель		ME16DOT	ME16DOP	ME16DOR
Количество выходов		16		
Тип соединения		Съёмный пружинный клеммник		
Тип выхода		NPN (Sinking)	PNP (Sourcing)	Реле
Напряжение		5~30 VDC		250VAC/30VDC
Макси-мальная нагрузка	Активная	500 мА(канал)		2А(канал)
	Индуктивная	-		
	Лампочка	-		
Максимальная выходная частота		1 КГц	1 КГц	1 Гц
Время переключения	ON	20мкс/50мкс	15мкс/40мкс	15мс/15мс
	OFF			
Потребляемая мощность		1.66 Вт	3.3 Вт	3.0 Вт
Габаритные размеры		ВхШхГ: 105.00 x 22.00 x 85.00 мм		

Внешний вид и размеры дискретных модулей расширения

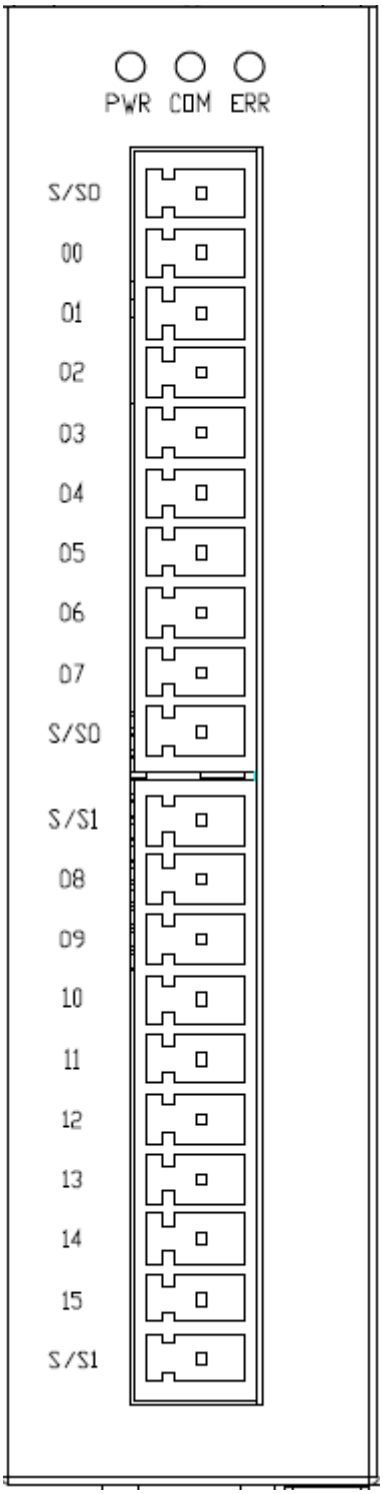
ME16DI/ME16DOT/ME16DOP/ME16DOR



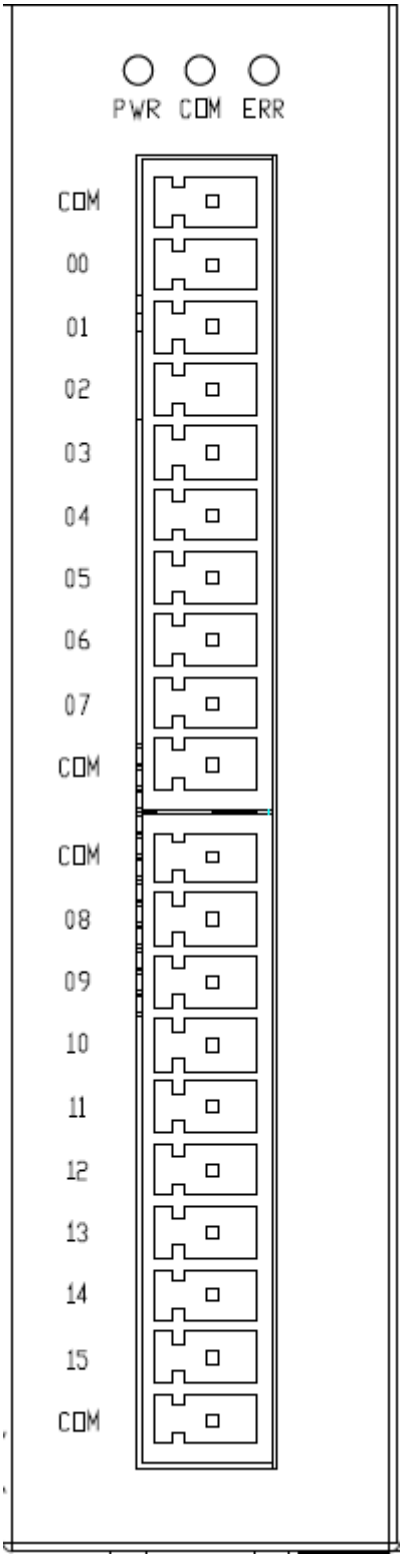
1	Светодиоды для входов/выходов: Горят при наличии сигнала на входе или выходе
2	Название модели
3	Съемный клеммник с пружинными клеммами
4	Разъем подключения модуля расширения
5	Разъем подключения модуля расширения
6	Крепление на DIN-рейку
7	Названия клемм
8	Шильдик

Расположение клемм дискретных модулей расширения

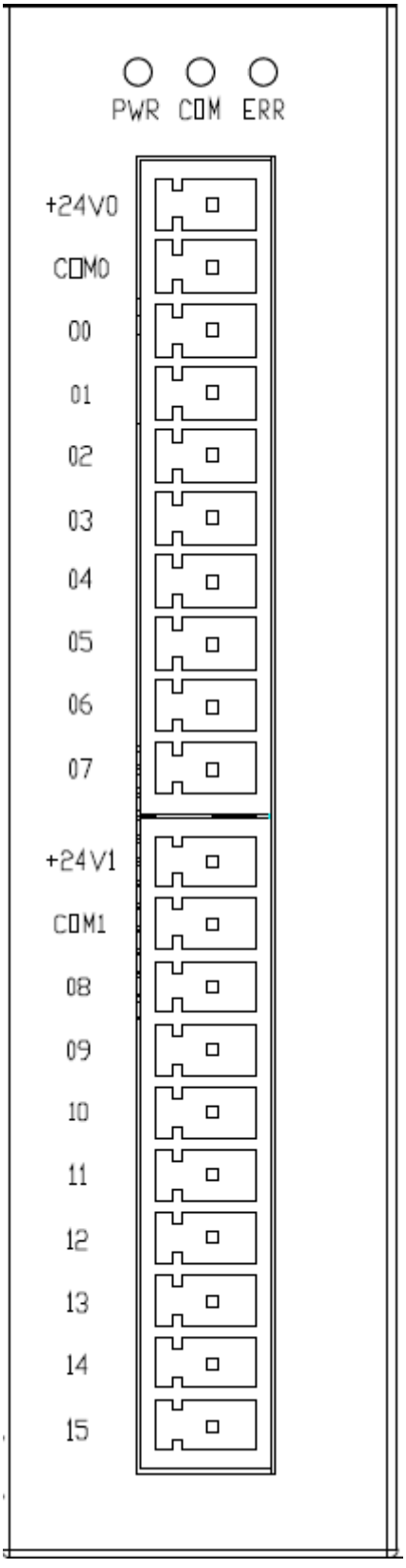
ME16DI



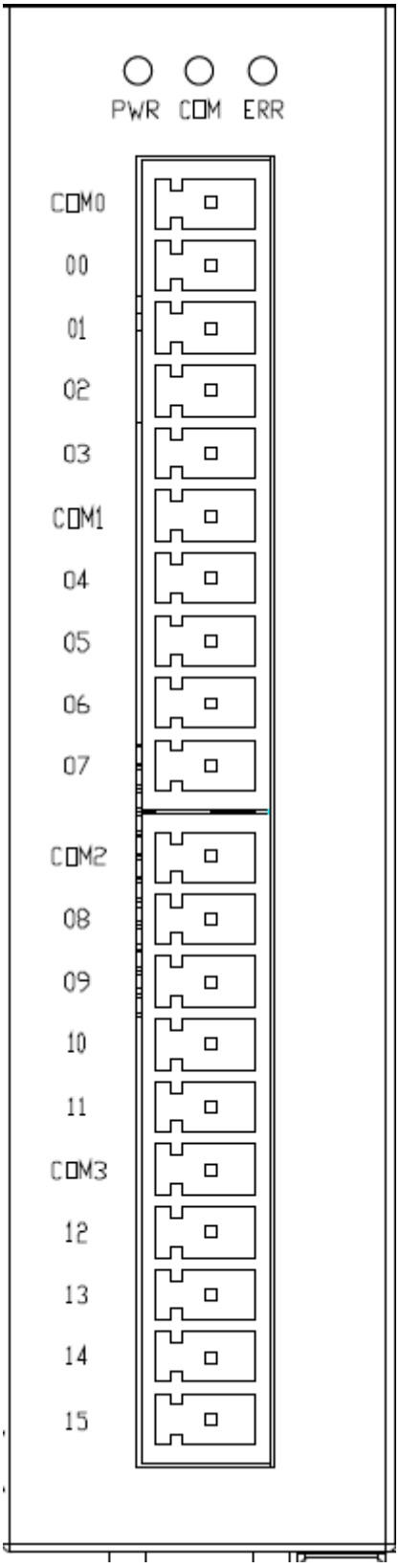
ME16DOT


PWR
COM
ERR
COM
00
01
02
03
04
05
06
07
COM
COM
08
09
10
11
12
13
14
15
COM

ME16DOP

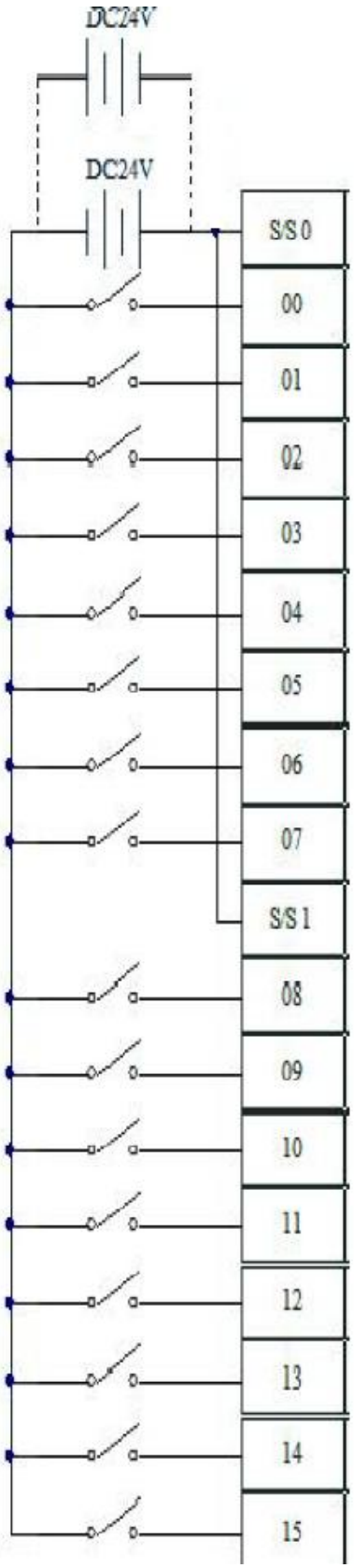

PWR
COM
ERR
+24V0
COM0
00
01
02
03
04
05
06
07
+24V1
COM1
08
09
10
11
12
13
14
15

ME16DOR

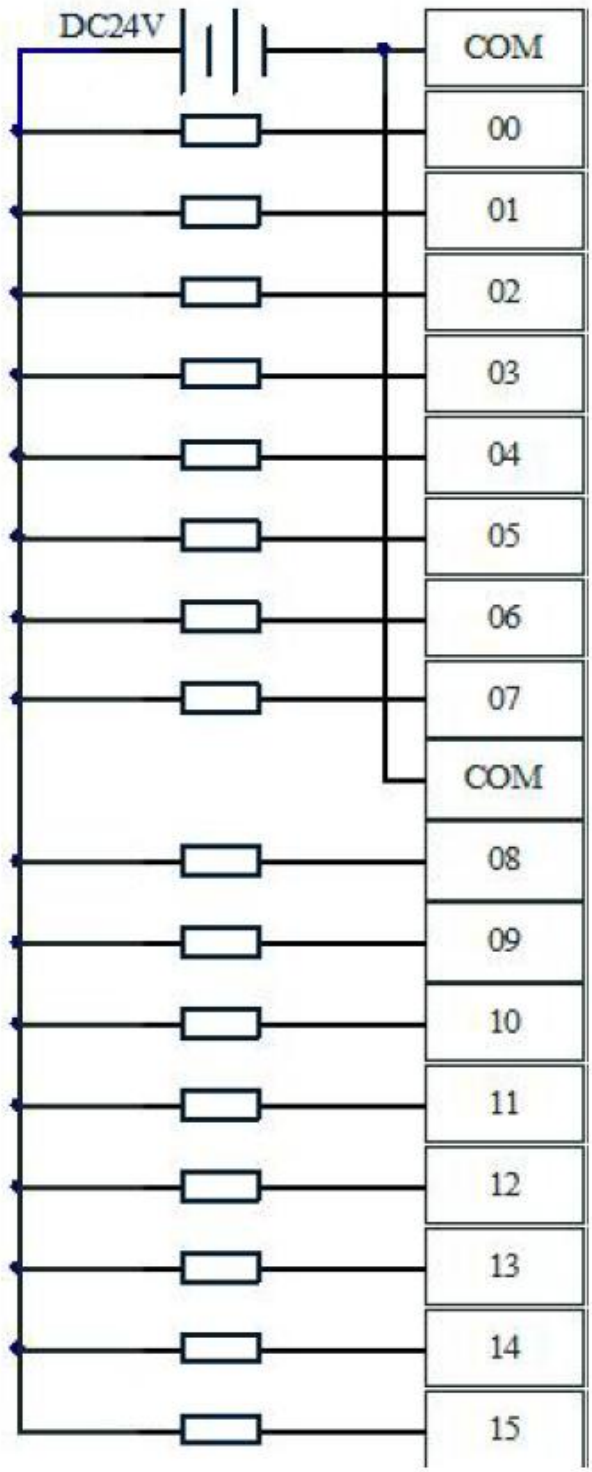

PWR
COM
ERR
COM0
00
01
02
03
COM1
04
05
06
07
COM2
08
09
10
11
COM3
12
13
14
15

Схемы подключения дискретных входов-выходов модулей расширения

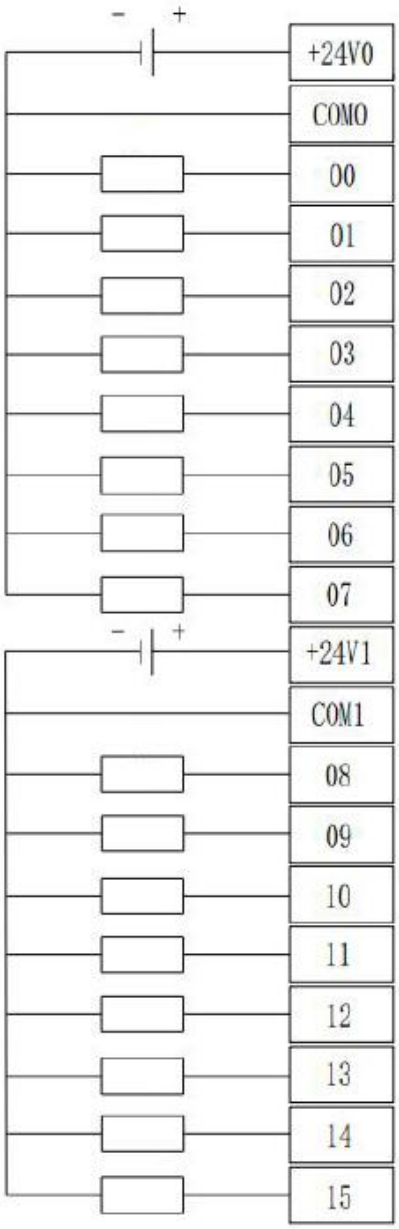
Схемы подключения входов на модуле расширения ME16DI



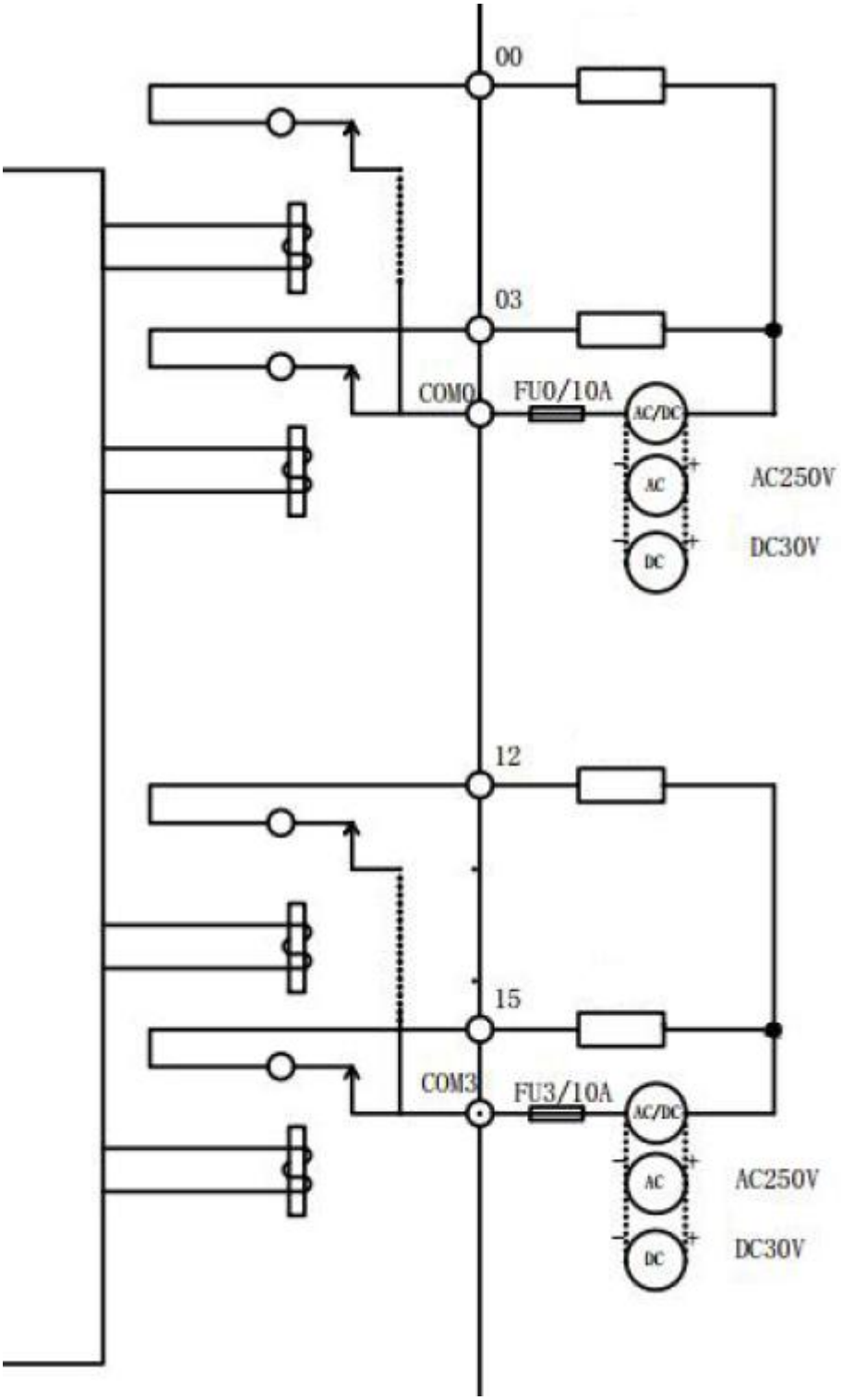
Схемы подключения выходов на модуле расширения ME16DOT



Схемы подключения выходов на модуле расширения ME16DOP



Схемы подключения выходов на модуле расширения ME16DOR



Спецификация модулей аналоговых входов-выходов

ME04AD

Общие характеристики

Модуль	MX04AD
Число входов	4
АЦП	Вход по напряжению / вход по току
Напряжение питания	24 VDC
Разъем	Разъемный клеммный блок
Время преобразования	1 мс/4 канала
Габаритные размеры	ВхШхГ: 105.00 x 22.00 x 85.00 мм
Потребляемая мощность	2.0 Вт
Рабочая температура	0-55℃

Характеристики преобразования входных сигналов

АЦП	Вход по напряжению				
Рабочий диапазон	-10 В~10 В	0 В~10 В	±5 В	0 В~5 В	1 В~5 В
Аппаратный диапазон	-10.2 В ~10.2 В	-0.2 В ~10.2 В	-5.1 В ~5.1 В	-0.1 В ~5.1 В	0.2 В ~5.1 В
Диапазон цифровой шкалы	-32000 ~ 32000				
Погрешность измерения при температуре 25℃	±0.2% FS				
Погрешность измерения при полном температурном диапазоне 0-55℃	±0.3% FS				
Аппаратное разрешение	16 бит				
Входной импеданс	>1 MΩ				

АЦП	Вход по току		
Рабочий диапазон	±20 мА	0 мА~20 мА	4 мА~20 мА
Аппаратный диапазон	-20.2 мА~20.2 мА	-0.2 мА~20.2 мА	3.8 мА~20.2 мА
Диапазон цифровой шкалы	-32000 ~ 32000		
Погрешность измерения при температуре 25℃	±0.3%		
Погрешность измерения при полном температурном диапазоне 0-55℃	±0.4%		
Аппаратное разрешение	16 бит		
Входной импеданс	250 Ω		

FS – Full Scale (полный размах шкалы, полный диапазон шкалы измерения)

MX04DA

Общие характеристики

Модуль	ME04DA
Число выходов	4
ЦАП	Выход по напряжению / выход по току
Напряжение питания	24 VDC
Разъем	Разъемный клеммный блок
Время преобразования	1 мс/4 канала
Габаритные размеры	ВхШхГ: 105.00 x 22.00 x 85.00 мм
Потребляемая мощность	2.0 Вт
Рабочая температура	0-55℃

Характеристики преобразования входных сигналов

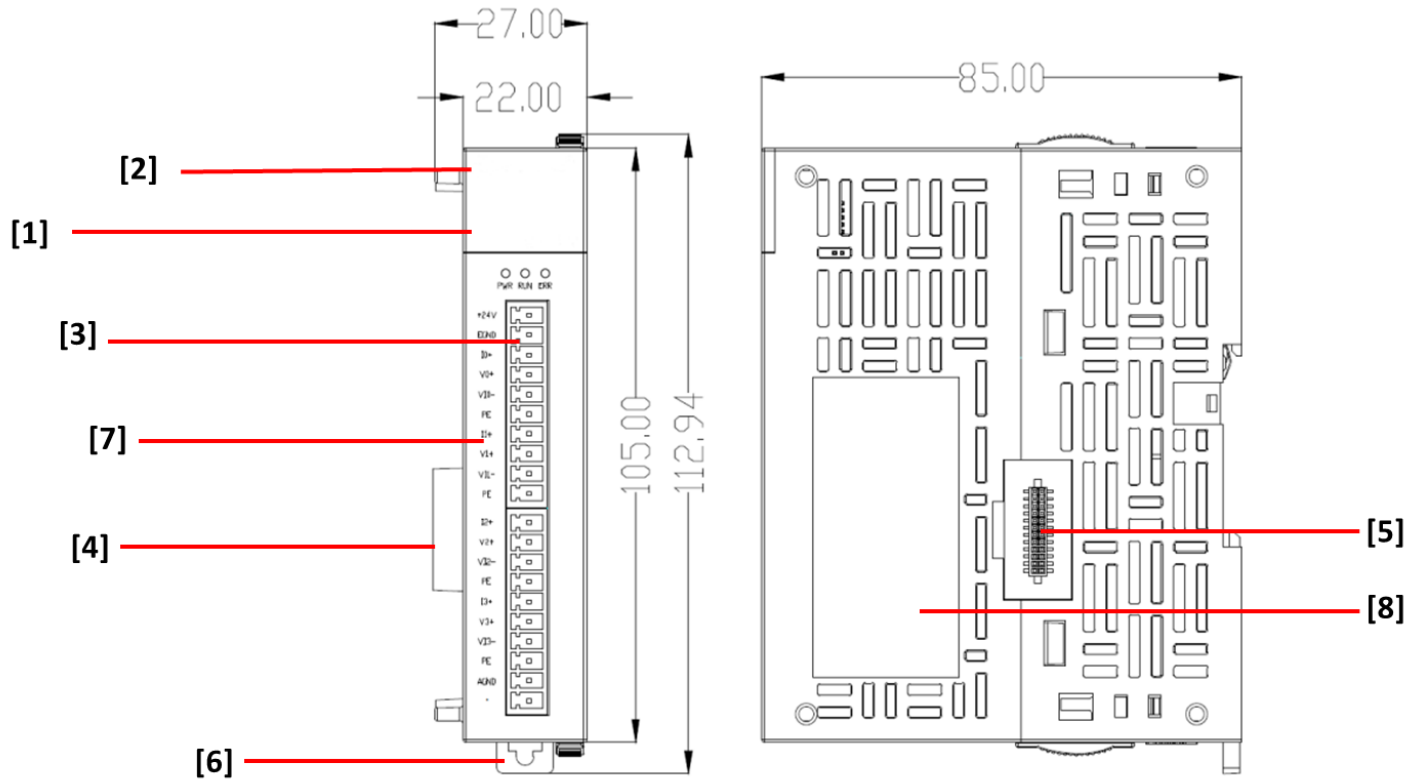
ЦАП	Выход по напряжению				
Рабочий диапазон	-10 В~10 В	0 В~10 В	±5 В	0 В~5 В	1 В~5 В
Аппаратный диапазон	-10.2 В ~10.2 В	-0.2 В ~10.2 В	-5.1 В ~5.1 В	-0.1 В ~5.1 В	0.2 В ~5.1 В
Диапазон цифровой шкалы	-32000 ~ 32000				
Погрешность осреднения при температуре 25℃	±0.2% FS				
Погрешность осреднения при полном температурном диапазоне 0-55℃	±0.3% FS				
Аппаратное разрешение	16 бит				
Входной импеданс нагрузки	1 KΩ ~ 1 MΩ				

ЦАП	Выход по току	
Рабочий диапазон	0 мА~20 мА	4 мА~20 мА
Аппаратный диапазон	-0.2 мА~20.2 мА	3.8 мА~20.2 мА
Диапазон цифровой шкалы	-32000 ~ 32000	
Погрешность измерения при температуре 25℃	±0.3% FS	
Погрешность измерения при полном температурном диапазоне 0-55℃	±0.4% FS	
Аппаратное разрешение	16 бит	
Входной импеданс нагрузки	100 - 500 Ω	

FS – Full Scale (полный размах шкалы, полный диапазон шкалы измерения)

Внешний вид и размеры аналоговых модулей

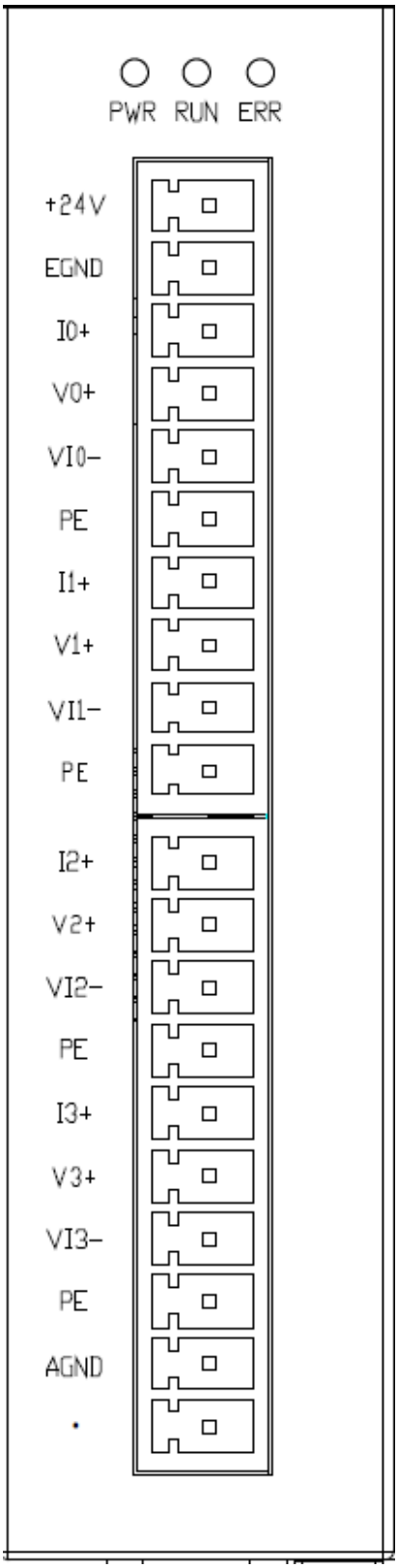
ME04AD/ME04DA



1	Светодиоды: RUN: работа (зеленый), ERR: ошибка (красный)
2	Название модели
3	Съемный клеммник с пружинными клеммами
4	Разъем подключения модуля расширения
5	Разъем подключения модуля расширения
6	Крепление на DIN-рейку
7	Названия клемм
8	Шильдик

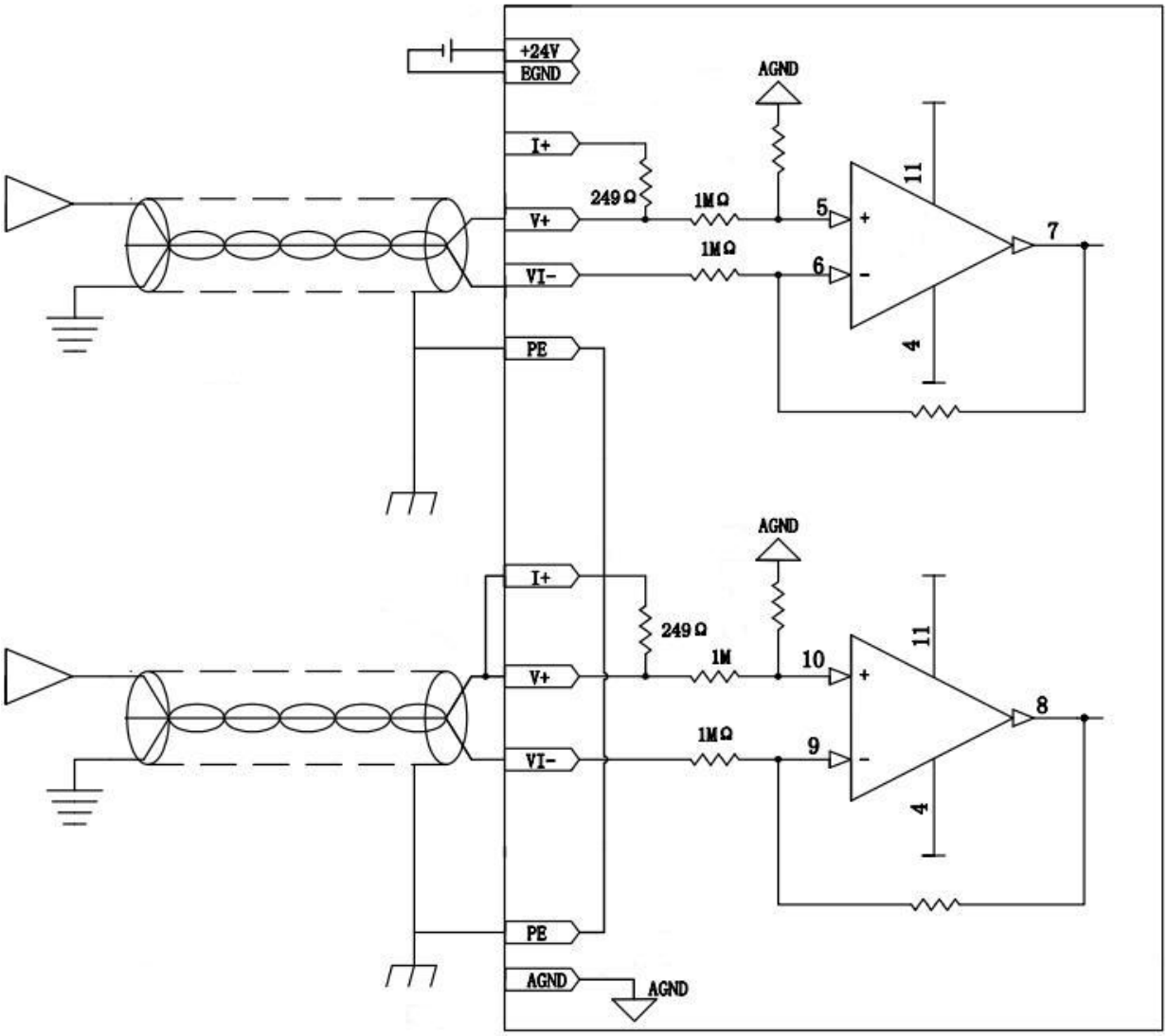
Расположение клемм аналоговых модулей

MX04AD/MX04DA



Схемы подключения аналоговых входов-выходов

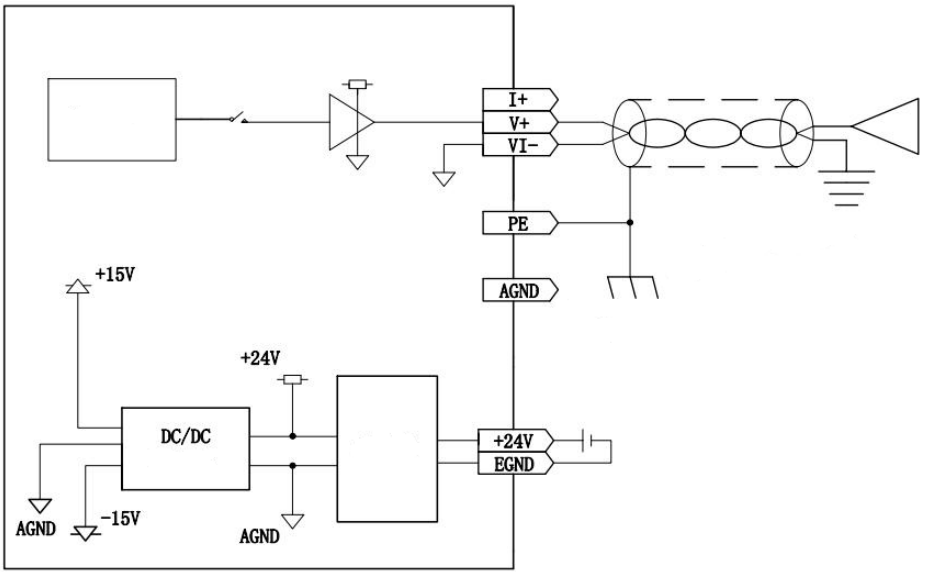
Схемы подключения входов на модуле расширения MX04AD



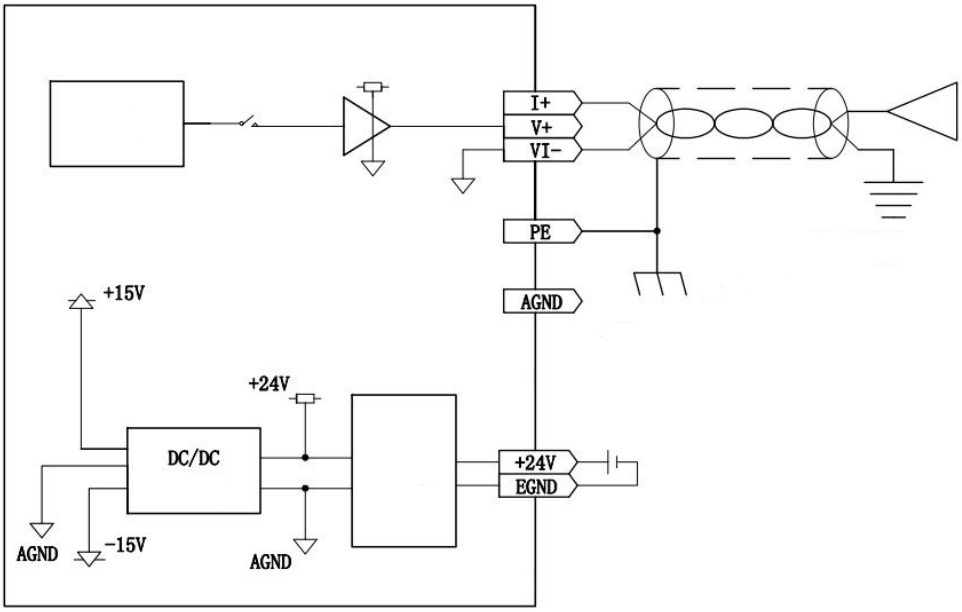
Модуль требует внешнего питания. Подключите 24 VDC к клеммам +24V и EGND.
Для подключения датчика используйте экранированную витую пару, экран которой нужно подключить к сигнальному заземлению.
Модуль необходимо установить на хорошо заземленную металлическую ДИН-рейку. В верхней части защёлки имеется металлический вывод, который при защёлкивании модуля на ДИН-рейке в ходит в контакт с ней. Обе клеммы PE соединены с этим выводом и между собой. Экраны можно подсоединить к клеммам PE при наличии хорошего контакта модуля и ДИН-рейки. При необходимости провод заземления можно подсоединить на одну из клемм PE.
При подключении токового сигнала поставьте перемычку между клеммами V+ и I+.
Когда входной сигнал является дифференциальным сигналом, клемму «AGND» можно подключить к аналоговой земле совместимого устройства (датчика) для устранения синфазных различий между устройствами (по сути это балластный провод). Данная мера повысит точность измерений.

Схемы подключения выходов на модуле расширения MX04DA

Потенциальный режим



Токовый режим



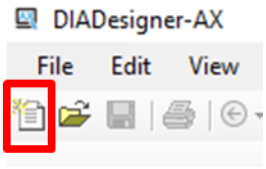
Модуль требует внешнего питания. Подключите 24 VDC к клеммам +24V и EGND.
Для подключения датчика используйте экранированную витую пару, экран которой нужно подключить к сигнальному заземлению.
Модуль необходимо установить на хорошо заземленную металлическую ДИН-рейку. В верхней части защёлки имеется металлический вывод, который при защёлкивании модуля на ДИН-рейке в ходит в контакт с ней. Обе клеммы PE соединены с этим выводом и между собой. Экраны можно подсоединить к клеммам PE при наличии хорошего контакта модуля и ДИН-рейки. При необходимости провод заземления можно подсоединить на одну из клемм PE.
Когда входной сигнал является дифференциальным сигналом, клемму «AGND» можно подключить к аналоговой земле совместимого устройства (датчика) для устранения синфазных различий между устройствами (по сути это балластный провод). Данная мера повысит точность измерений.

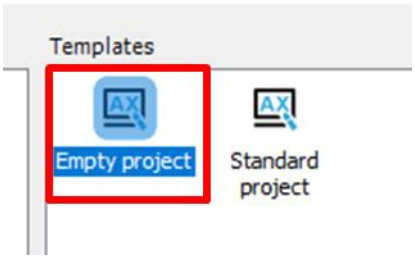
Запуск среды программирования и создание проекта

Установите на ПК (ноутбук) среду програм2мирования DesignerAX следуя командам Мастера установки. После установки на рабочем столе появится иконка:

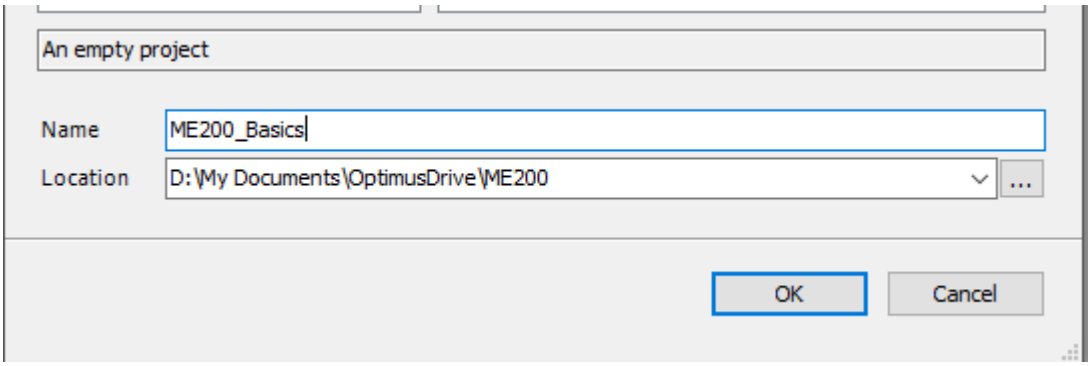


Для запуска среды программирования щёлкните дважды левой кнопкой мышки по иконке.
В открывшемся рабочем окне выберите меню создания проекта и выберите создание пустого проекта **«Empty Project»**.





Дайте название проекту:



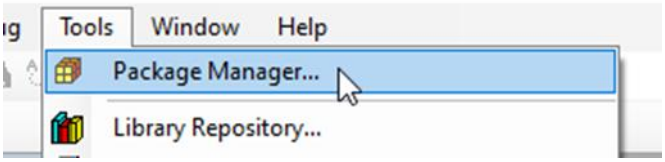
В открывшемся рабочем окне в верхнем левом углу древа проекта будет единственный пункт с названием проекта.



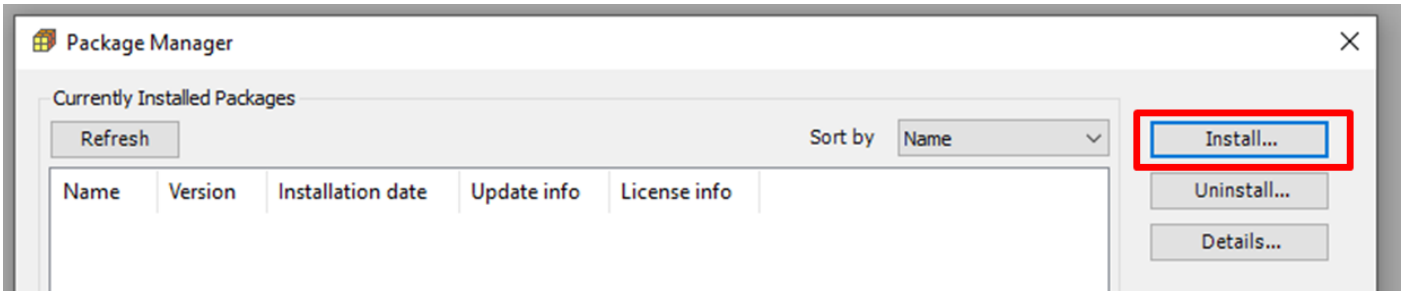
Установка описания устройства для контроллера ME200

Для начала работы с контроллером семейства ME200 необходимо установить в среду программирования файлы с описанием устройства. В данном случае это будет пакет **«CODESYS Package»** с расширением **.package**, который содержит в себе описания контроллера и модулей расширения, библиотеки, USB драйвер и примеры.

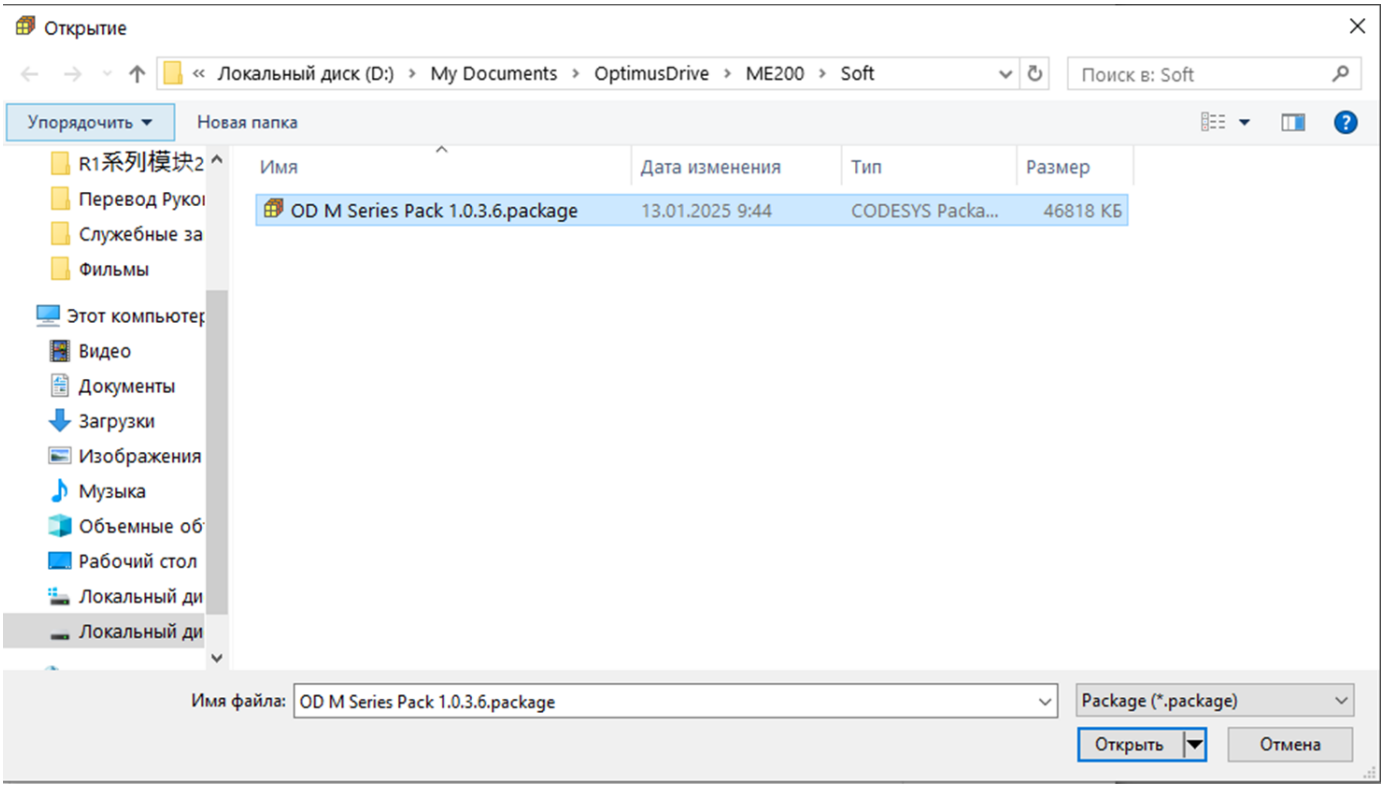
В созданном проекте войдите в меню установки пакетов устройств:



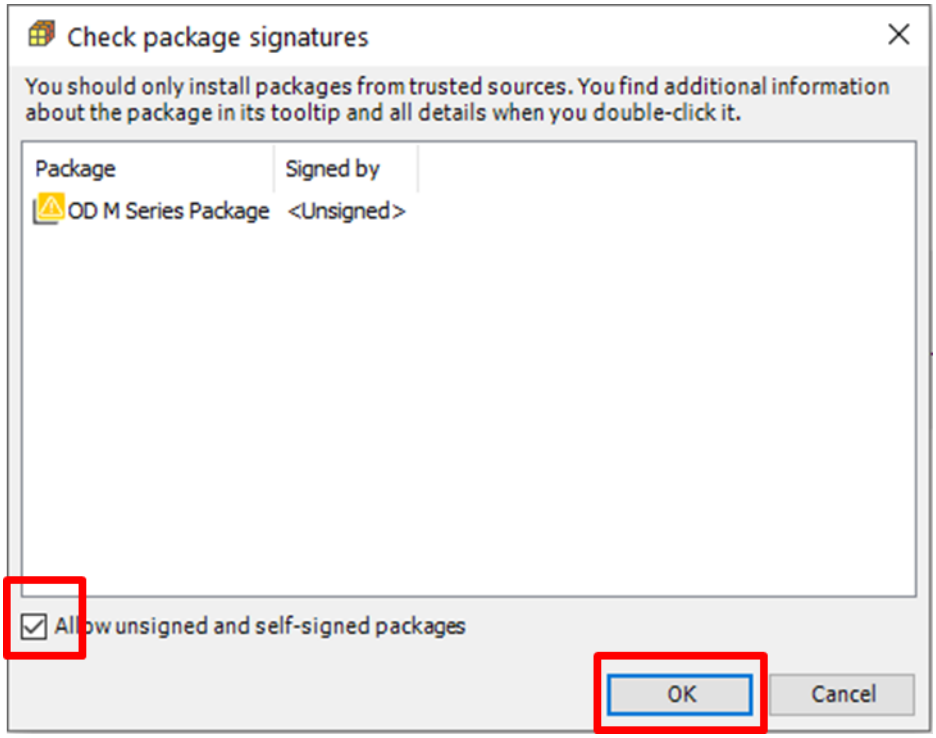
В открывшемся Мастере установки пакетов (Package Manager) нажмите кнопку **Install**:



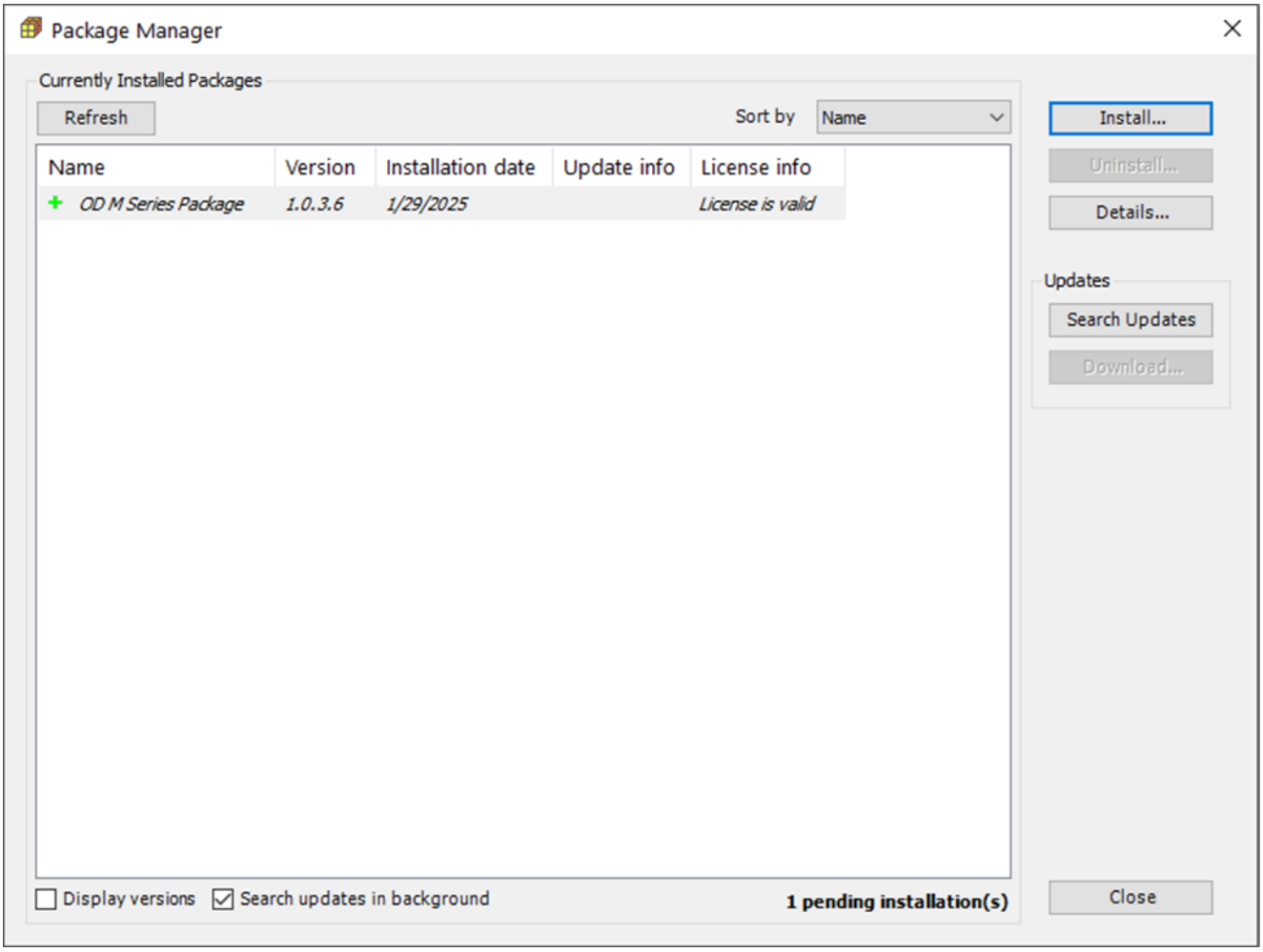
Выберите в проводнике нужный файл:



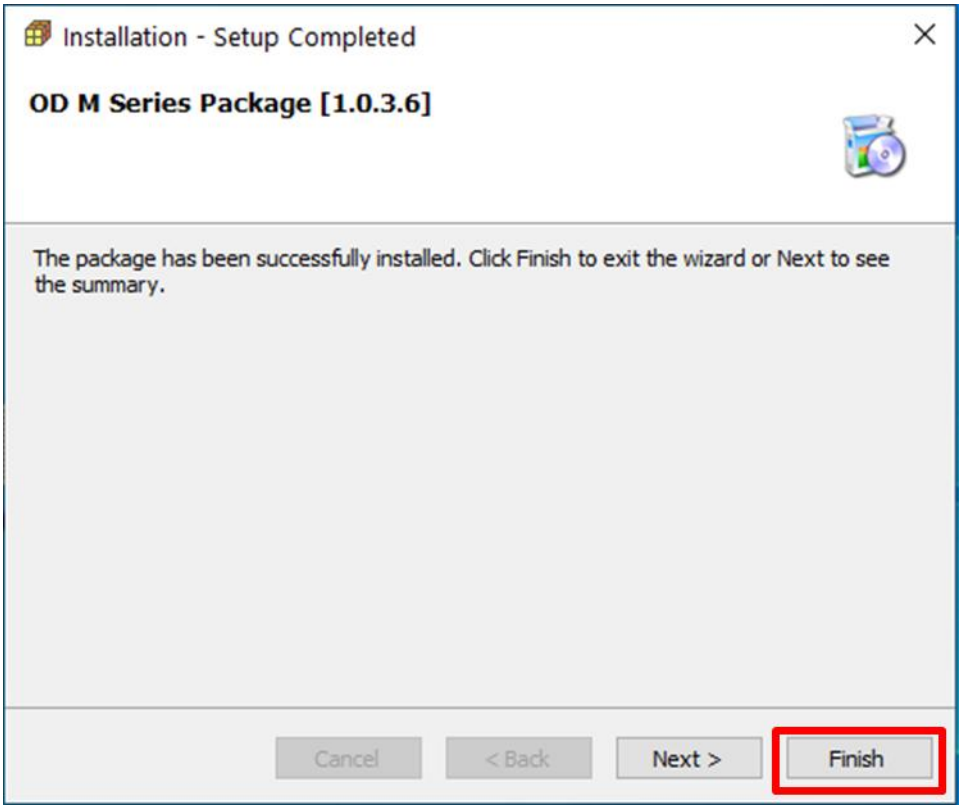
Поставьте флажок и нажмите ОК:



Появится сообщение о готовности к установке пэкиджа:

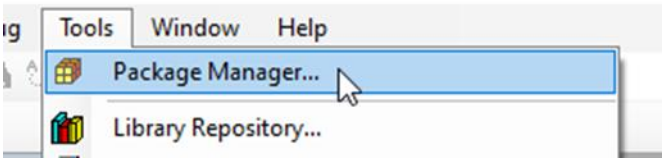


Закройте среду программирования DesignerAX, начнётся установка пэкиджа. Следуйте сообщениям на экране. По окончанию установки нажмите **Finish**:

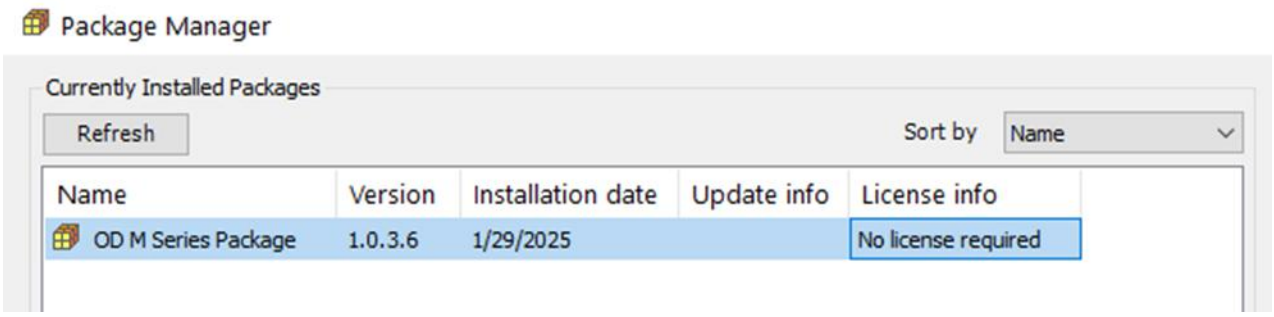


Запустите заново среду программирования DesignerAX.

Войдите в меню установки пакетов устройств:

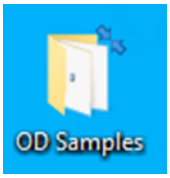


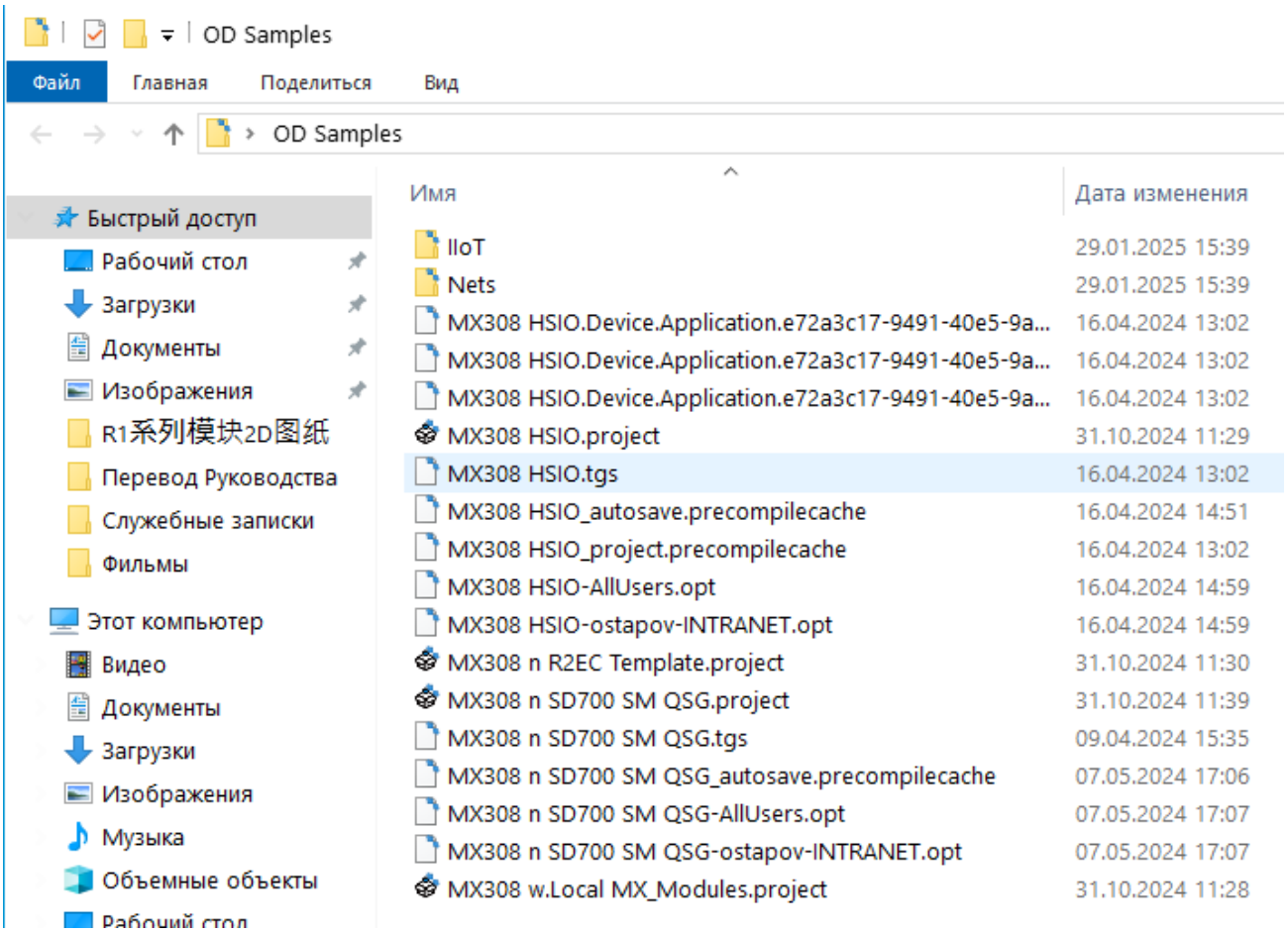
После установки появится запись в списке установленных пакетов:



Закройте менеджер пакетов.

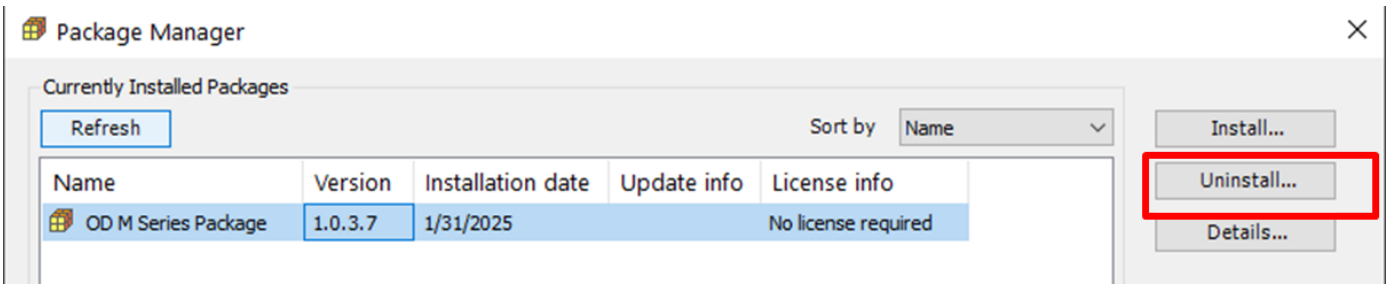
На рабочем столе появится папка с примерами:



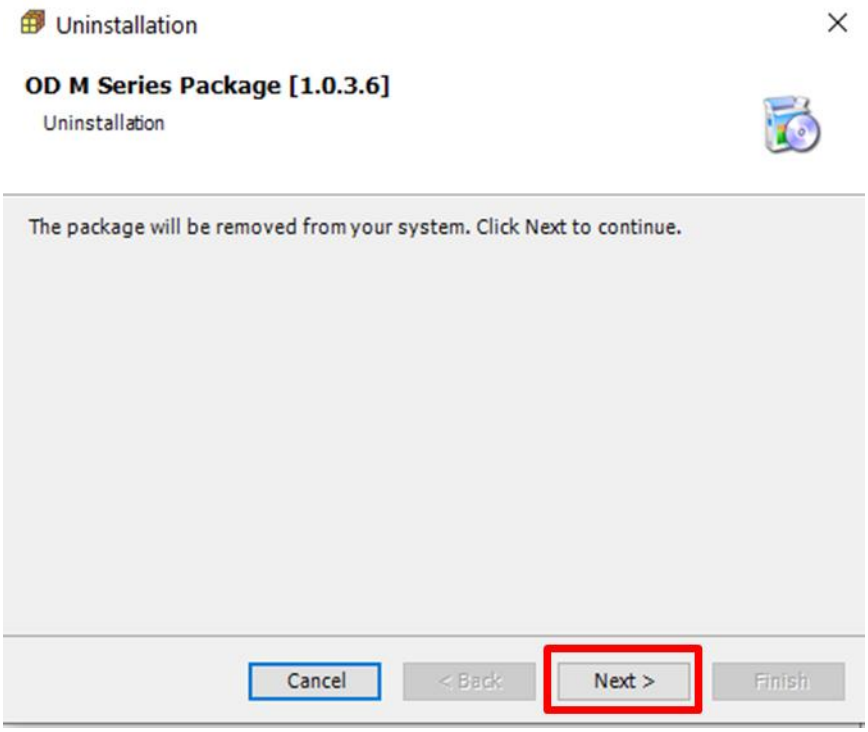


На этом установка пакета закончена и можно переходить к добавлению устройств в проект.

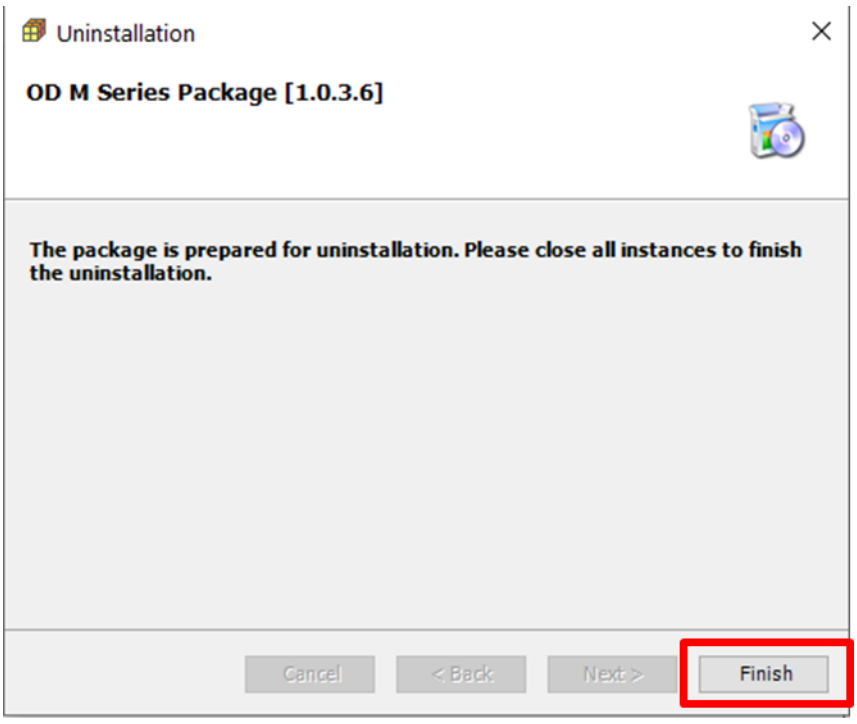
Для удаления пэкиджа необходимо нажать кнопку **Uninstall**:



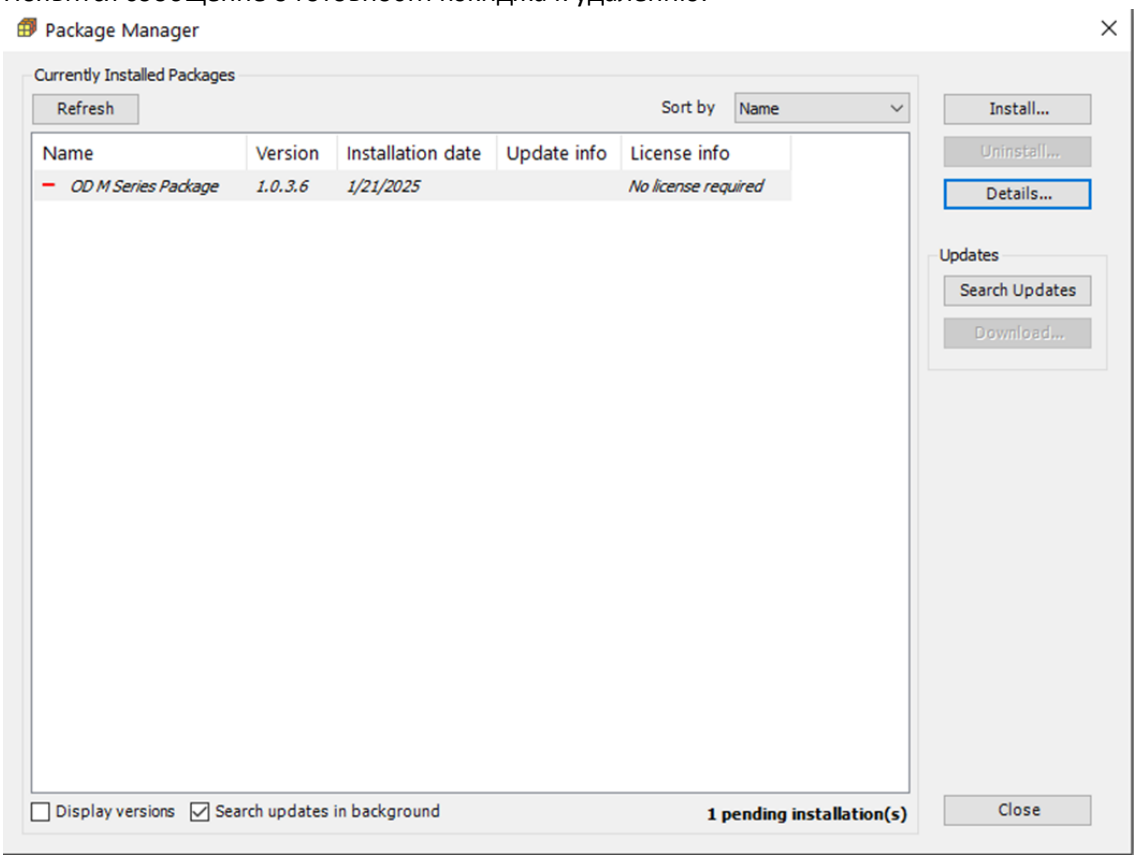
Далее **Next**:



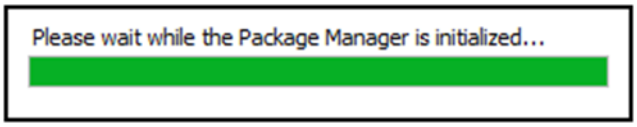
Далее **Finish**:



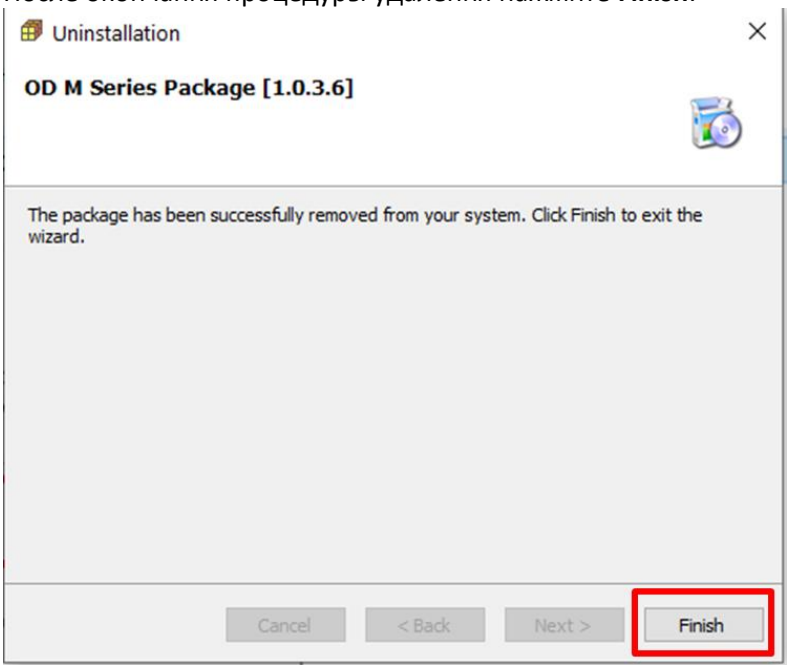
Появится сообщение о готовности пэкиджа к удалению:



После этого необходимо закрыть среду программирования. Начнётся удаление пэкиджа.



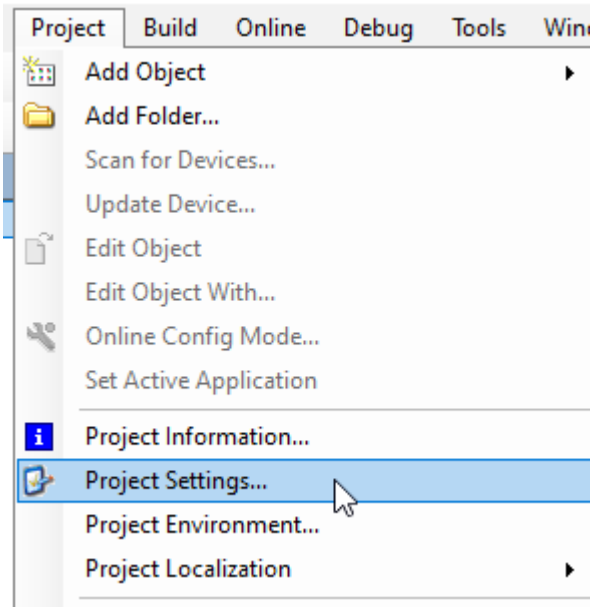
После окончания процедуры удаления нажмите **Finish**:



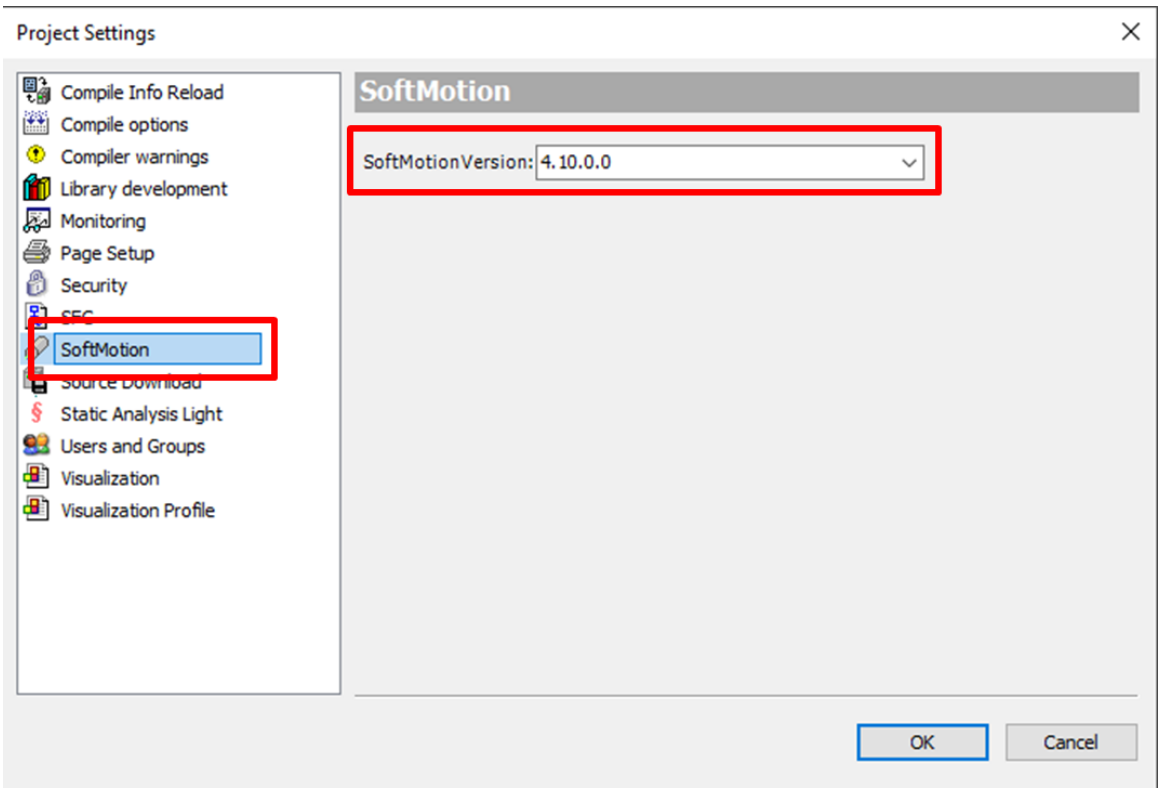
Определение версии библиотеки 3S SoftMotion (SM3)

Для версии DesignerAX 1.7 и выше наиболее подходящей версией библиотеки 3S SoftMotion является 4.16. Но можно выбрать и версию 4.10 при необходимости. Для выбора версии библиотеки 3S SoftMotion выполните следующие действия:

Зайдите в меню **Project – Project Settings**:

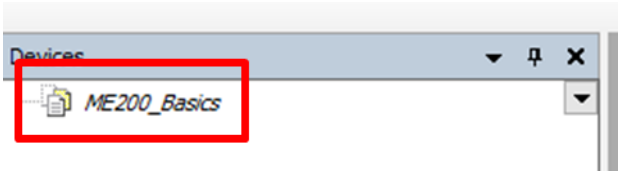


В открывшемся окне выберите пункт SoftMotion и версию 4.10.00:

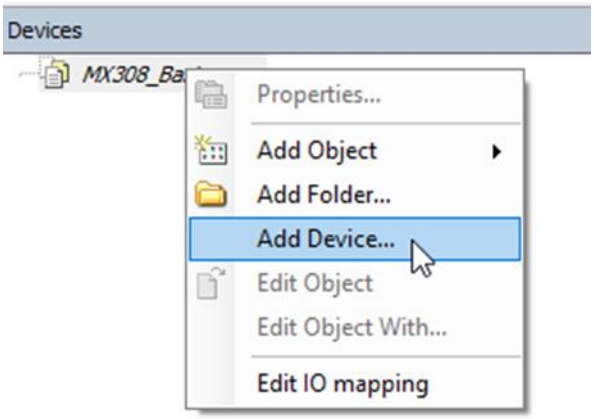


Добавление контроллера в проект

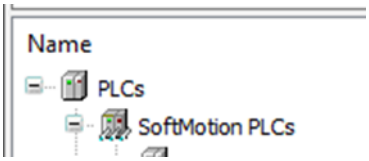
Для добавления контроллера в проект необходимо выполнить следующие действия.
Щёлкните правой кнопкой мышки на названии проекта в левом верхнем углу проекта:



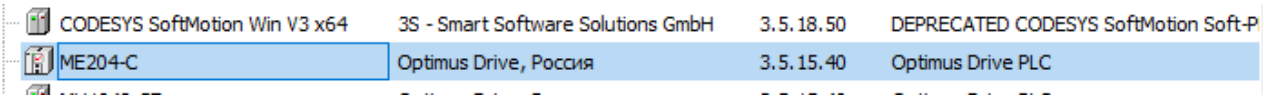
В появившемся меня выберите пункт **Add Device**:



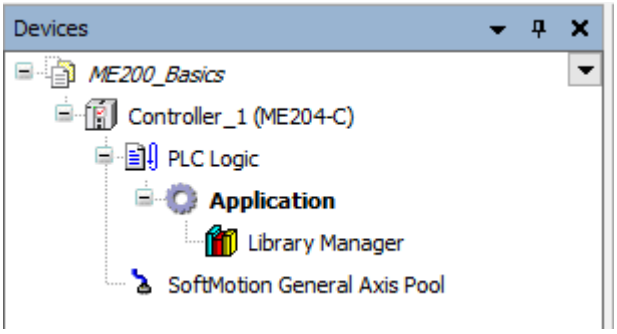
В открывшемся окне выберите раздел **PLC – SoftMotion PLC**:



Далее прокрутите вниз до пункта с контроллером ME200

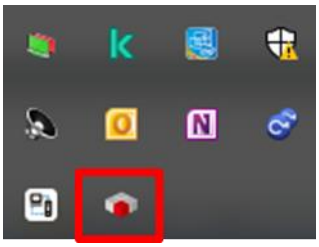


выберите его и нажмите кнопку Add Device и закройте окно. В древе проекта появится устройство – контроллер ME200:

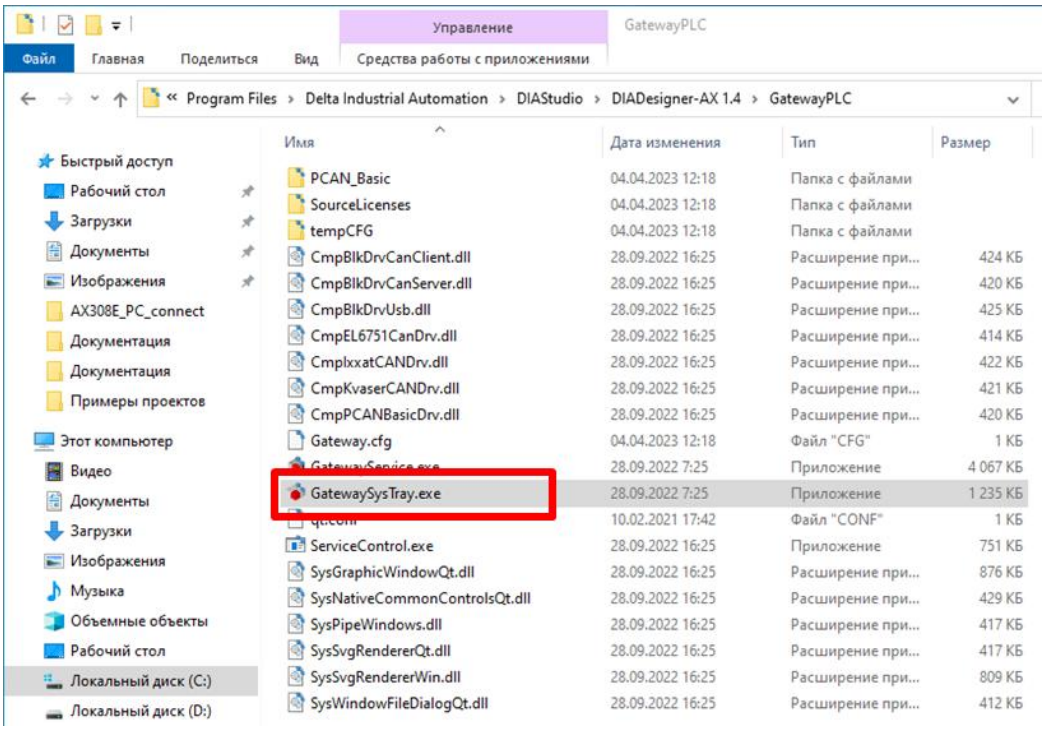


Организация связи контроллеров типа ME200 и среды программирования. Загрузка программы. Онлайн режим

После установки среды программирования в Windows System Tray (правый нижний угол экрана) должна появиться иконка шлюза CODESYS Gateway V3:

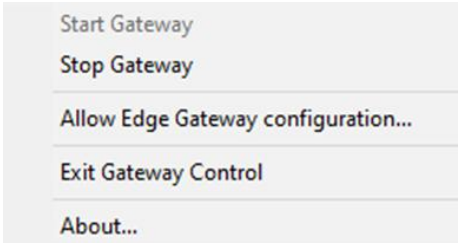


Если шлюз как приложение не запущен, то его можно запустить принудительно через исполнительный файл:



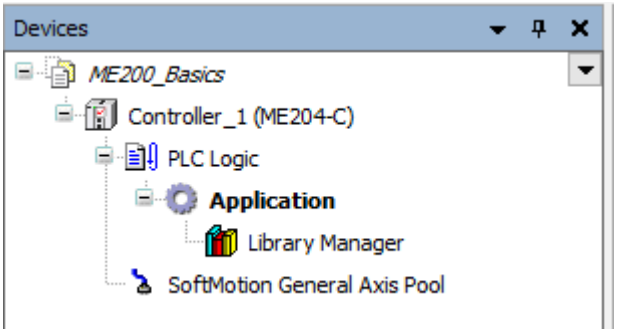
После этого должна появиться иконка.

Если щёлкнуть на иконке шлюза левой кнопки мышки, то откроется меню, позволяющее включить/выключить шлюз и закрыть приложение:

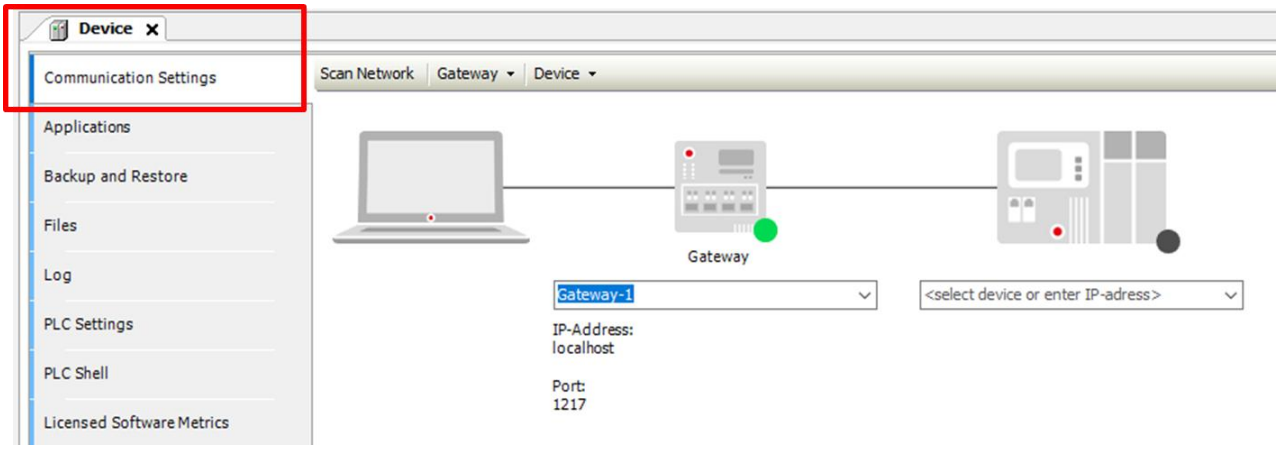


Шлюз используется для организации связи контроллера со средой программирования. Для связи можно использовать USB или Ethernet. Далее рассматриваются оба варианта подключения.

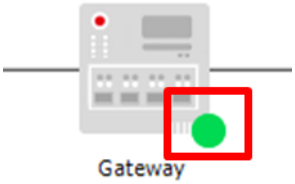
В древе проекта щёлкните дважды на названии контроллера:

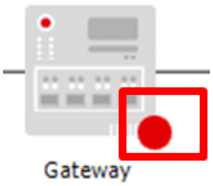


Откроется вкладка **Device** пункт **Communication Settings**:



По индикатору зелёный/красный можно судить от том, включен шлюз или нет.

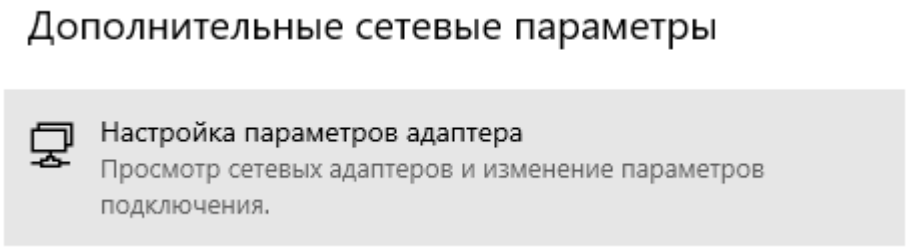




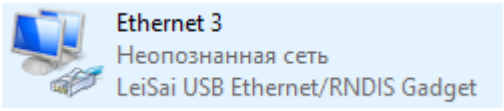
Соединение через порт USB-C

USB драйвер устанавливается на ПК автоматически при инсталляции пэкиджа, см. соответствующий раздел данного Руководства.

Соедините ПК и контроллер через USB-C. В Диспетчере устройств не появляется при этом никаких записей. Драйвер работает через сетевые подключения, поэтому откройте Параметры сети и Интернет Windows – Настройки параметров адаптера:

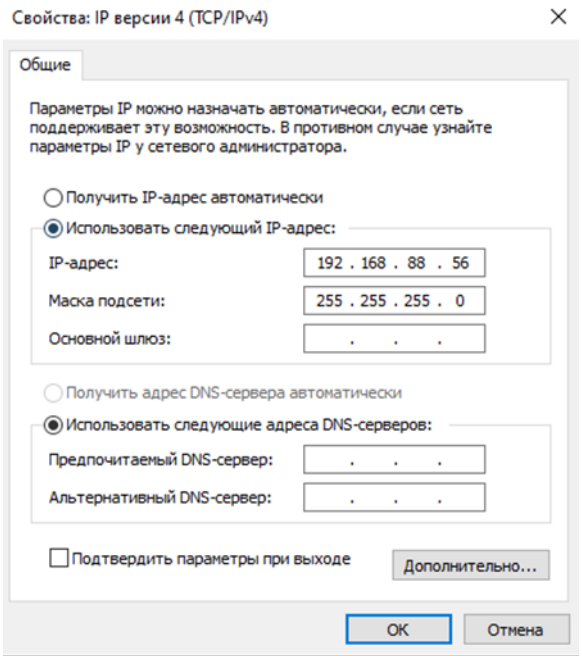


При подключении контроллера к ПК и правильно установленном драйвере должна появиться Неопознанная сеть данного вида:

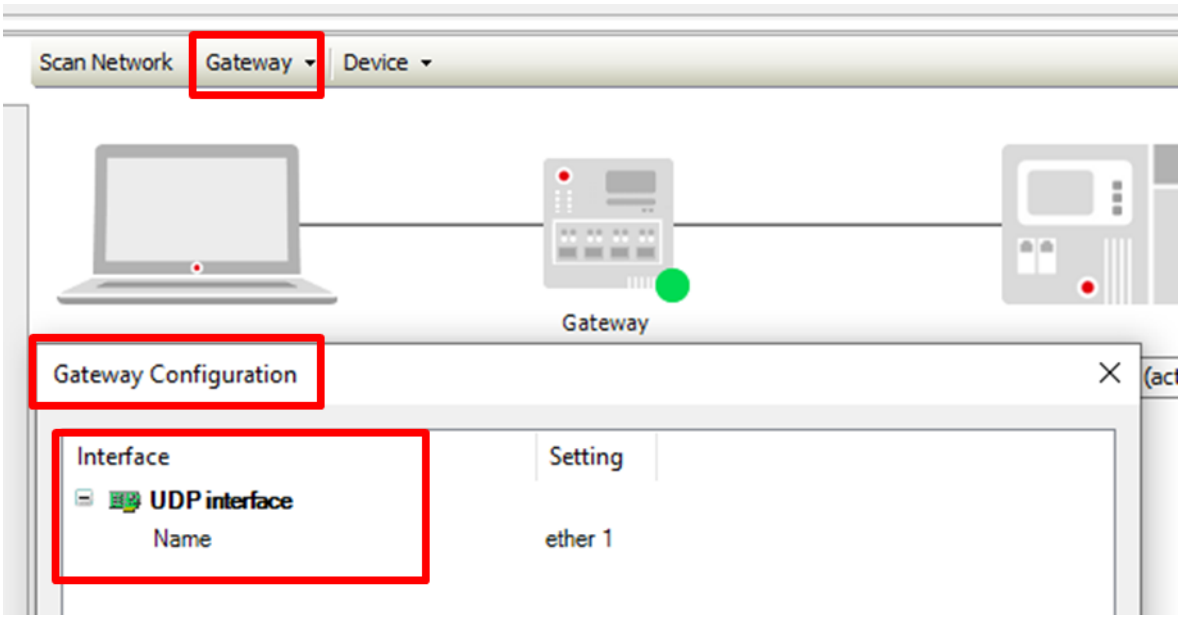


При соединении по USB сетевой контроллера будет фиксированный: **192.168.88.88**

Поэтому адрес ПК в данном виртуальном адаптере должен быть в данной подсети, но не такой же:

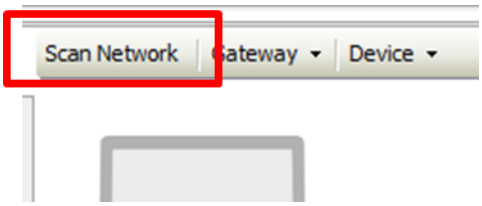


В общем случае для соединения достаточно драйвера по умолчанию:

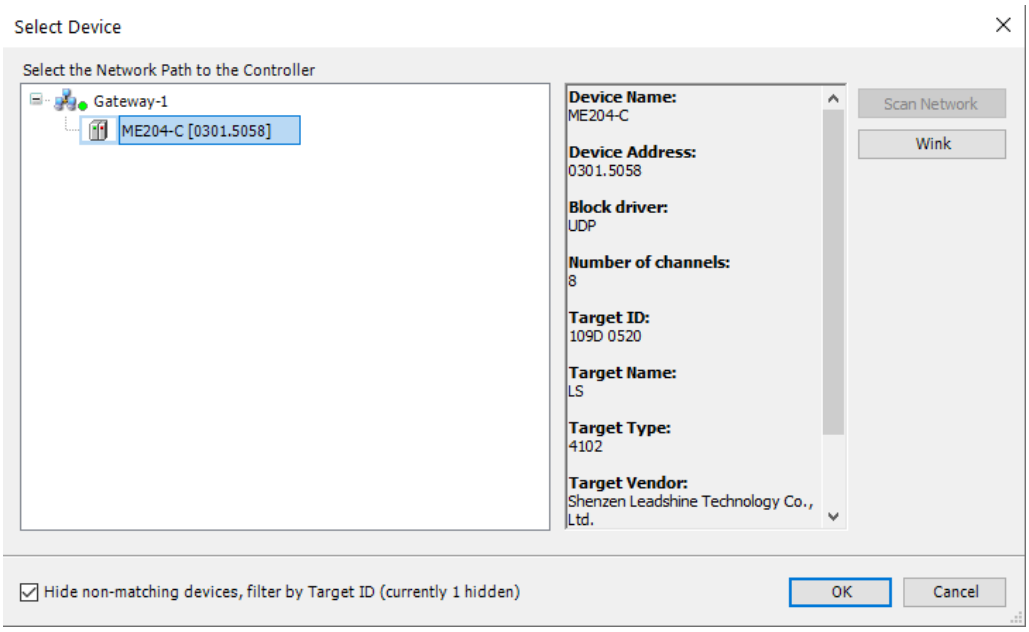


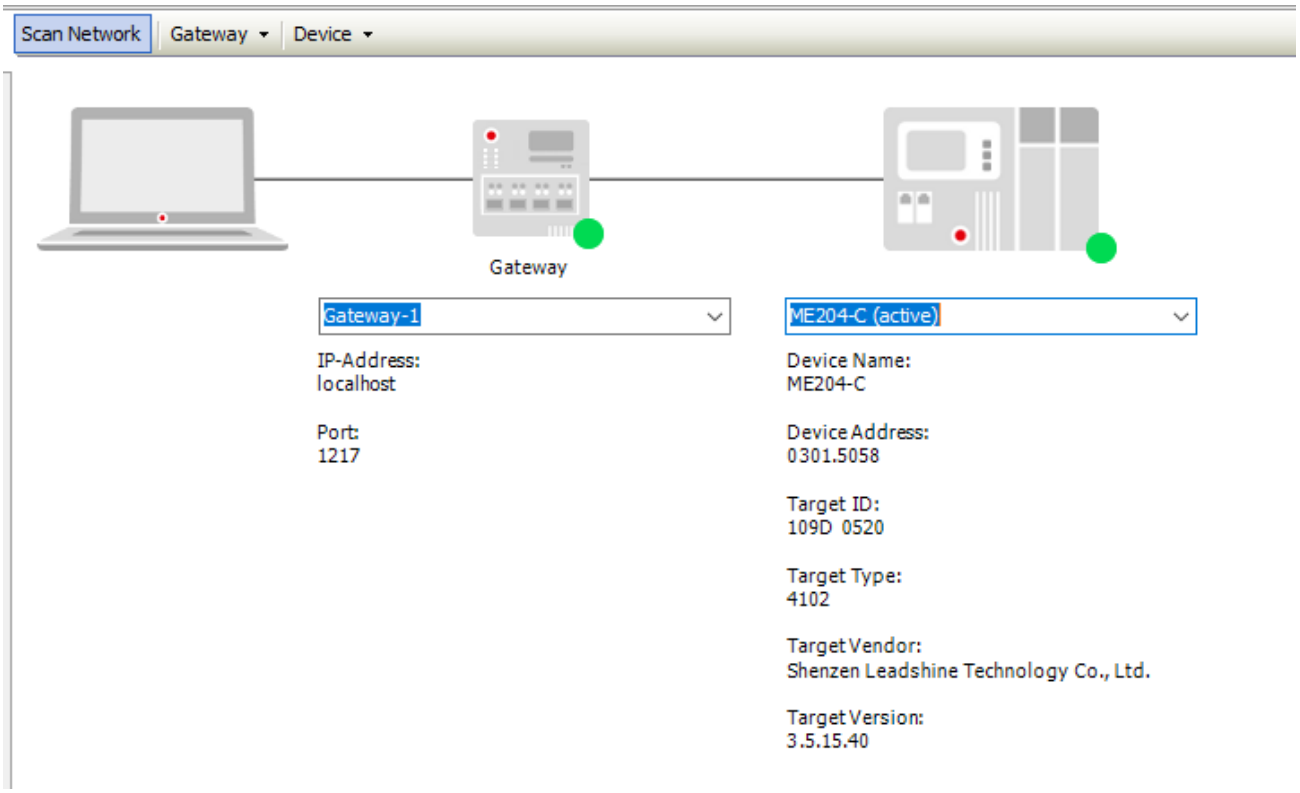
После этого можно приступать к установлению соединения среды программирования и контроллера.

В пункте **Communication Settings** нажмите иконку **Scan Network**:



Появится окно поиска и в итоге должен быть обнаружен контроллер:





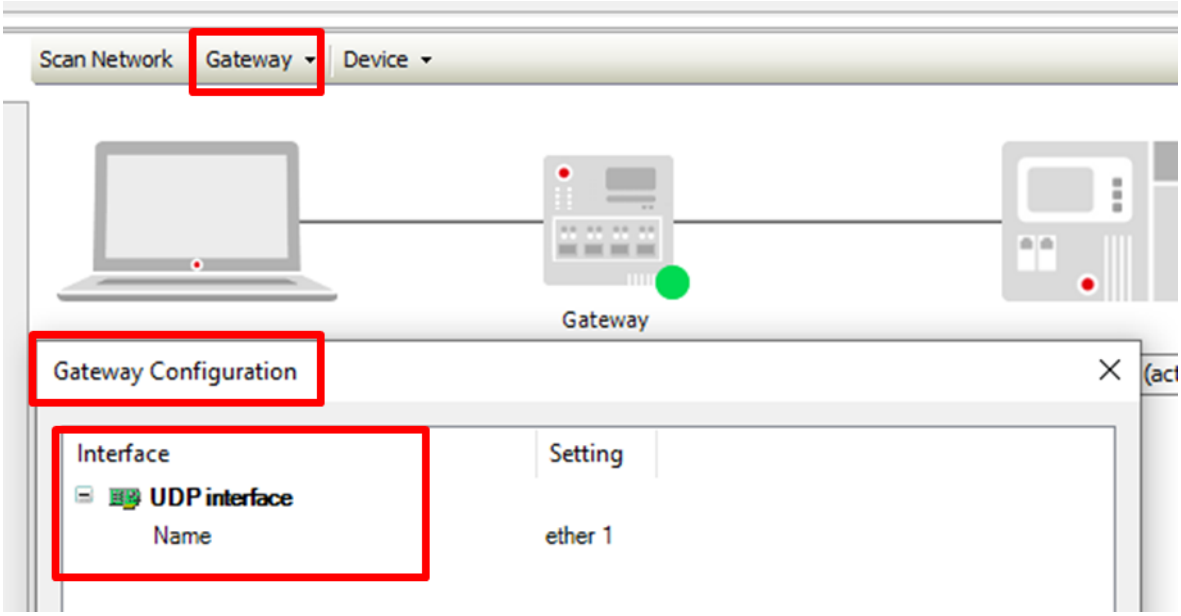
Внимание!
Если соединение не устанавливается, то можно сбросить в контроллере настройки связи на заводские. Для этого необходимо быстро включить-выключить переключатель RUN-STOP на передней панели контроллера. На пятый раз начнёт мигать на некоторое время светодиод RUN. После данной процедуры контроллер будет стоять на заводском IP адресе 192.168.1.3.

Соединение через порт Ethernet

Для установления соединения посредством сети Ethernet соедините патчкордом порты контроллера и ПК напрямую или через неуправляемый коммутатор (свитч).

У контроллера IP адрес по умолчанию - **192.168.1.3**

В общем случае для соединения достаточно драйвера по умолчанию:

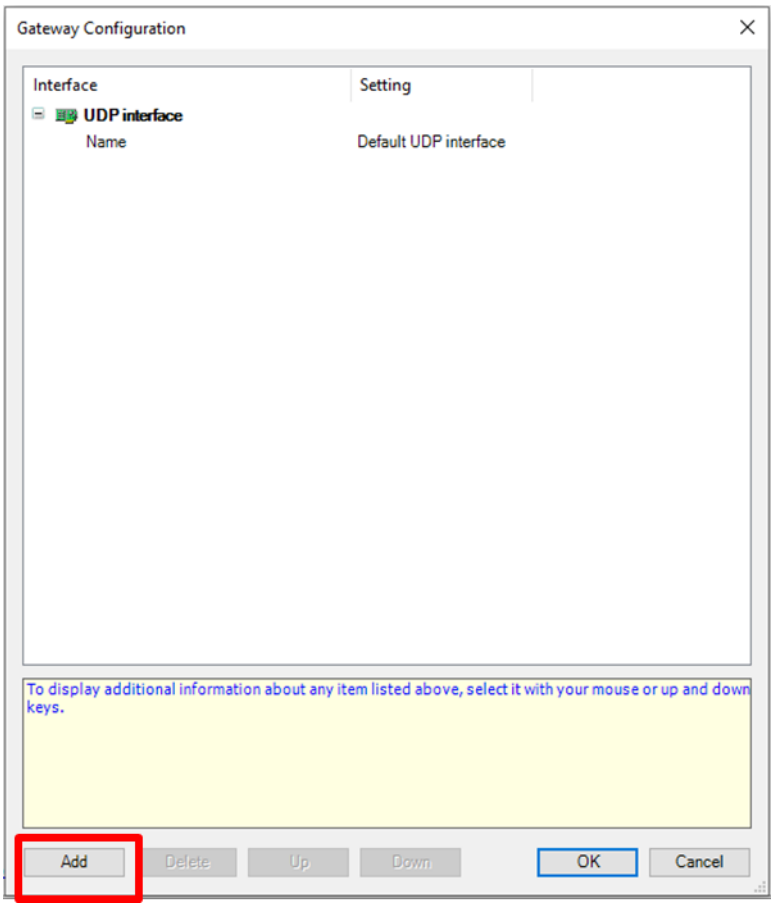


В случае соединения через несколько коммутаторов возможно потребуется создать явное TCP соединение.

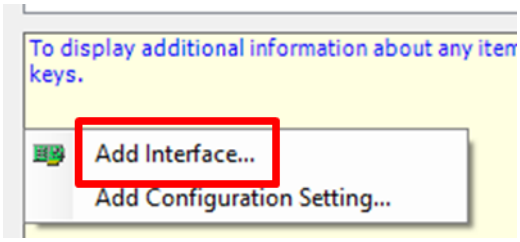
Для этого выберите пункт **Gateway – Configure the Local Gateway**:



Появится окно со списком каналов связи, которое в новом проекте будет пустым:

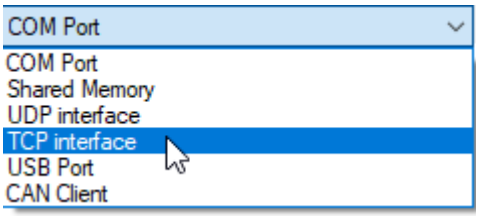


В левом нижнем углу нажмите кнопку **Add**, появится окно создания интерфейса:

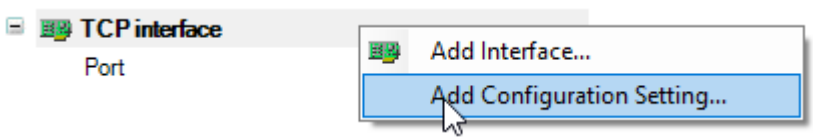


Выберите пункт **Add Interface**.

Появится ещё один пункт, где нужно выбрать **TCP interface**:

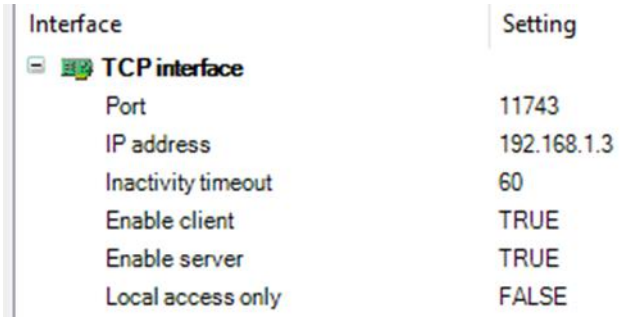


Щёлкните правой кнопкой мышки на надписи **TCP interface** и выберите пункт **Add Configuration Setting**:

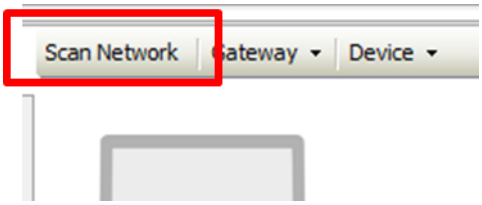


И выберите как минимум поле IP адреса. Введите IP адрес контроллера. При последующих нажатиях будут появляться следующие поля, которые можно не менять и не выводить.

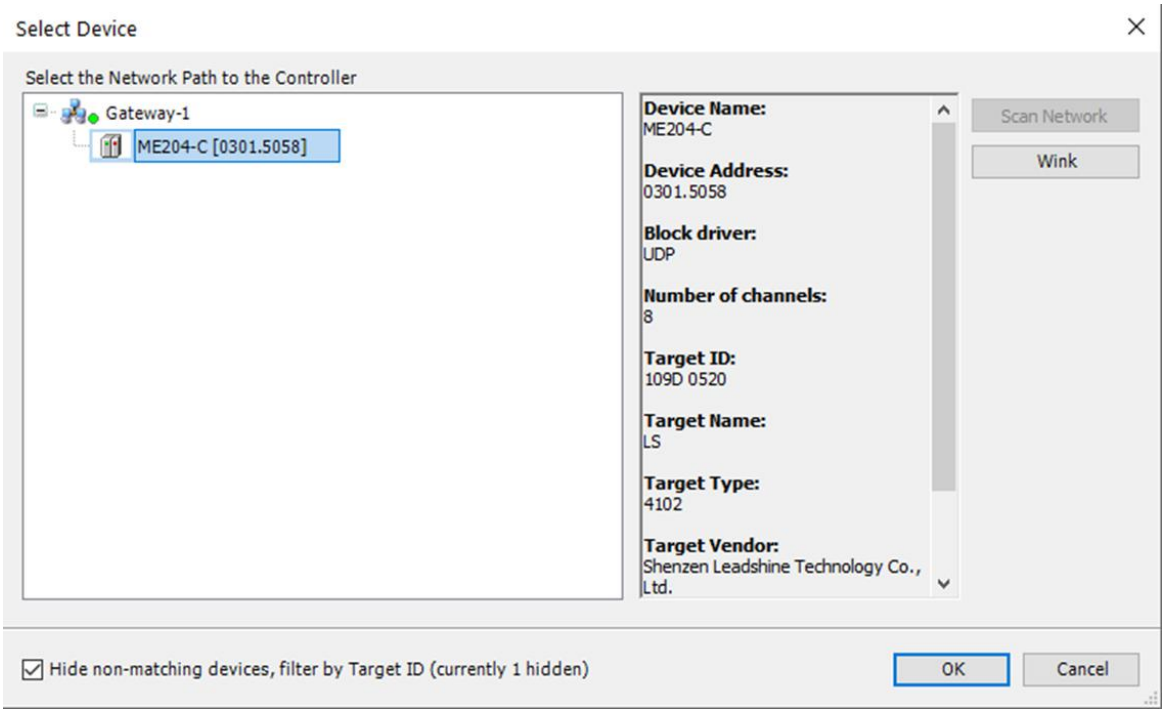
У контроллера IP адрес по умолчанию - **192.168.1.3**



После создания канала связи нажмите иконку **Scan Network**:



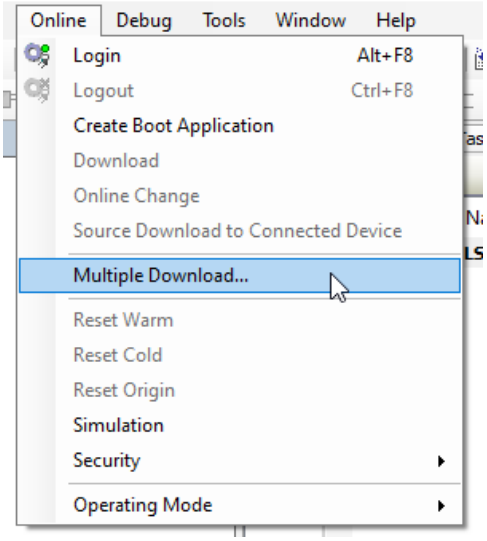
Появиться окно с поиском и найденным контроллером:



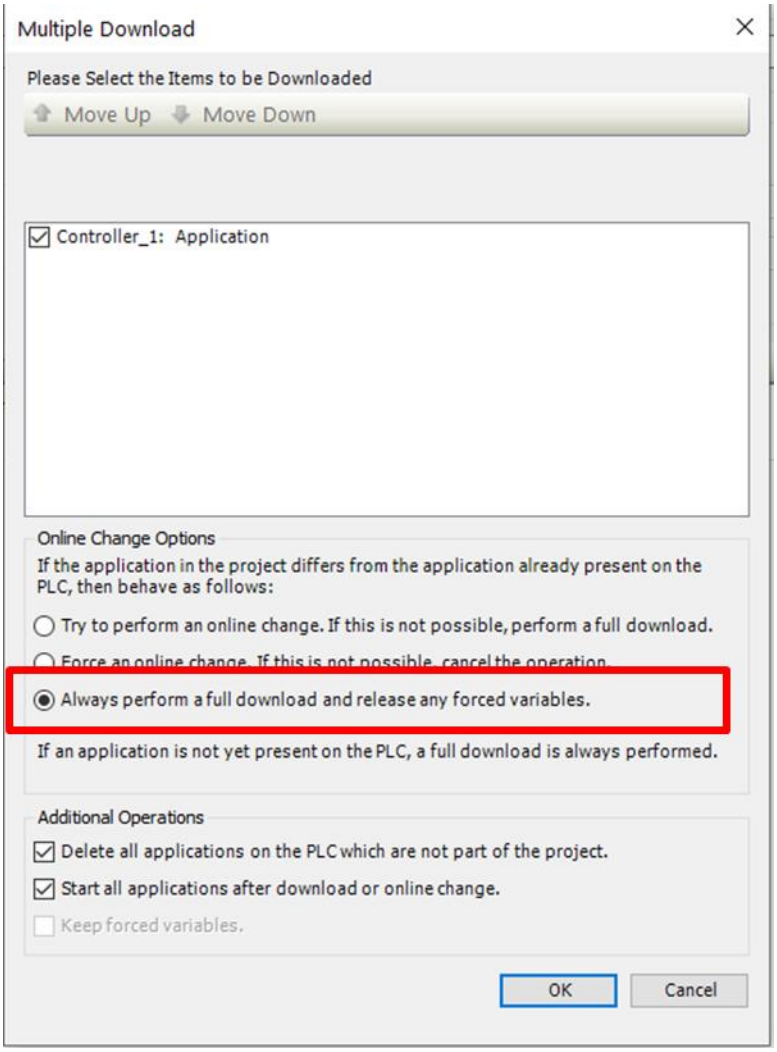
Внимание!
Если соединение не устанавливается, то можно сбросить в контроллере настройки связи на заводские. Для этого необходимо быстро включить-выключить переключатель RUN-STOP на передней панели контроллера. На пятый раз начнёт мигать на некоторое время светодиод RUN. После данной процедуры контроллер будет стоять на заводском IP адресе 192.168.1.3.

Загрузка проекта в контроллер и вход в режим онлайн

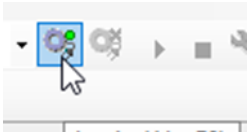
Для загрузки проекта в контроллер необходимо установить связь ПК – контроллер (см. выше), выбрать в меню Online выбрать пункт Multiple Download:



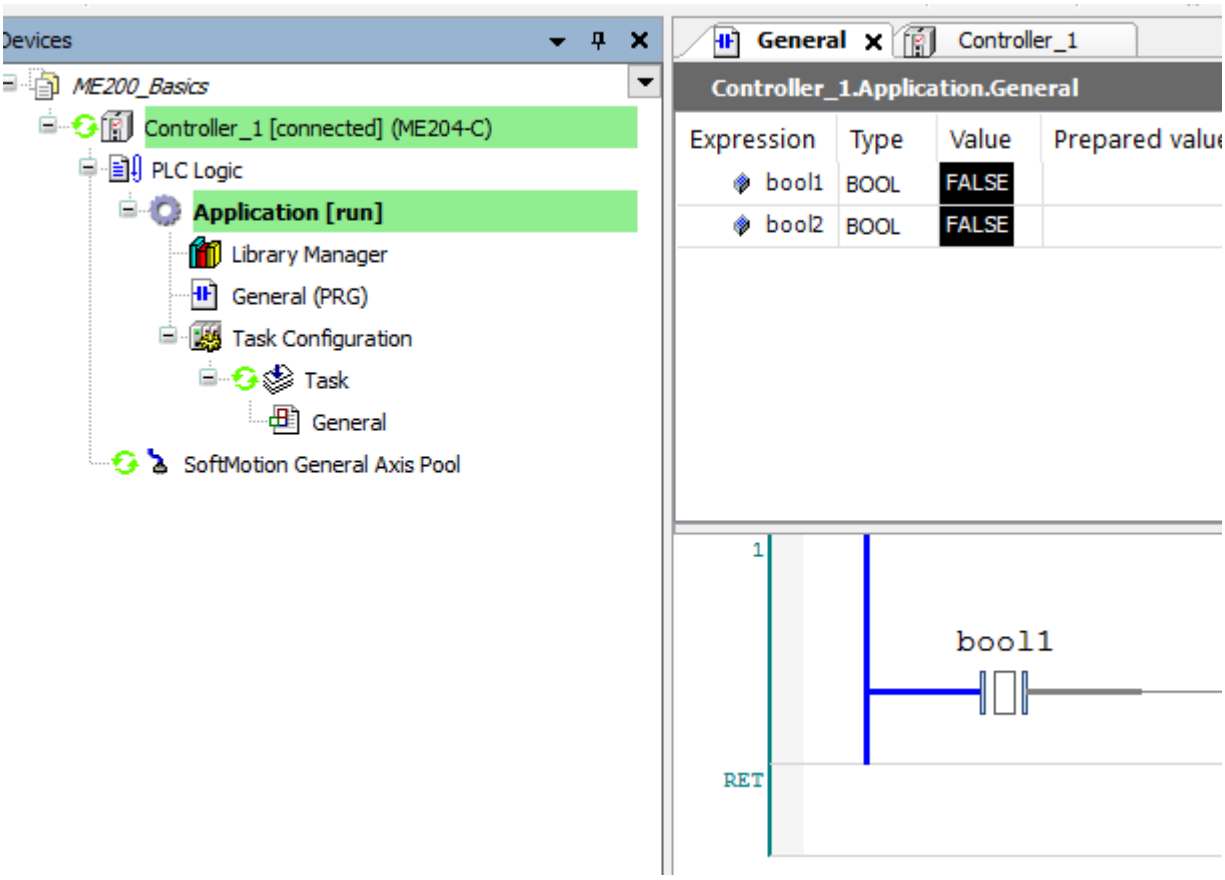
В открывшемся окне выберите вариант полной загрузки и нажмите **ОК**:



Для входа в режим онлайн нажмите кнопку:



Программа подсветится состоянием объектов:

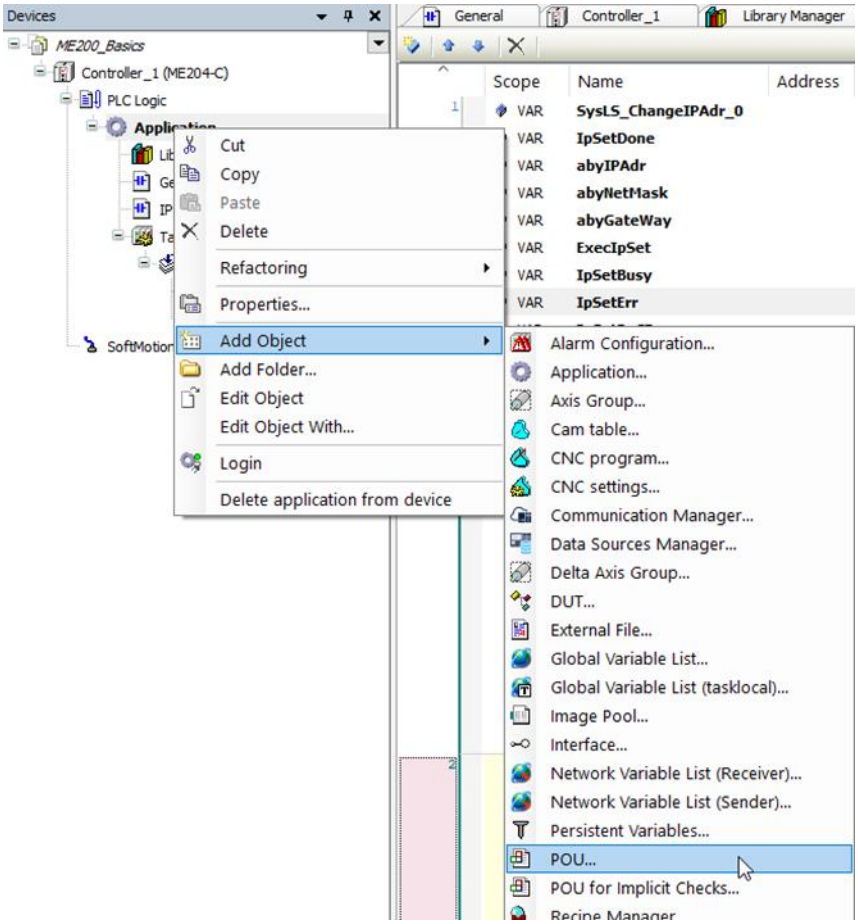


Для выхода из режима онлайн нажмите кнопку с красным крестиком:

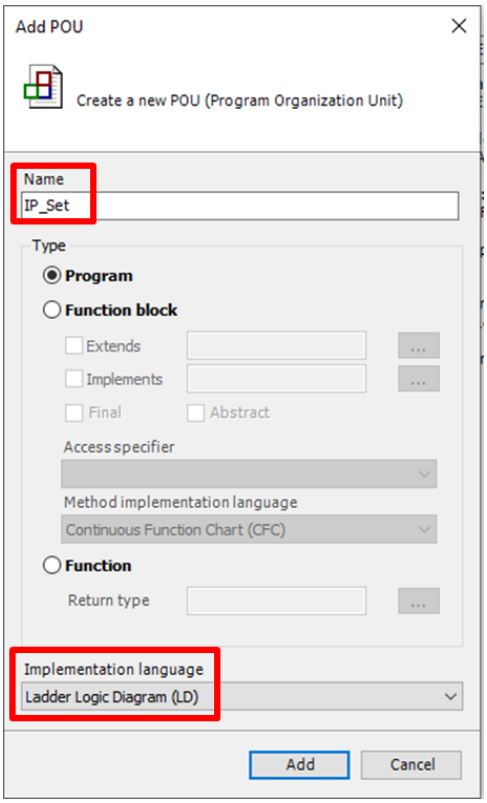


Изменение IP адреса контроллера из программы контроллера

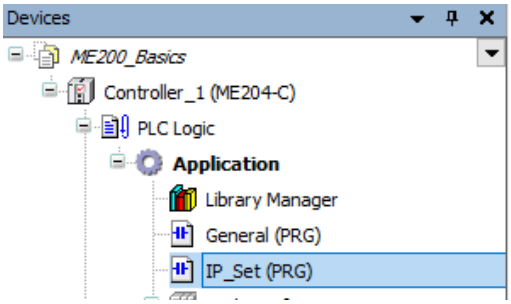
Создайте в проекте контроллера программную единицу (POU). Для этого щёлкните в древе проекта на пункте **Application** правой кнопкой мышки и в открывшемся меню выберите пункт **Add Object – POU:**



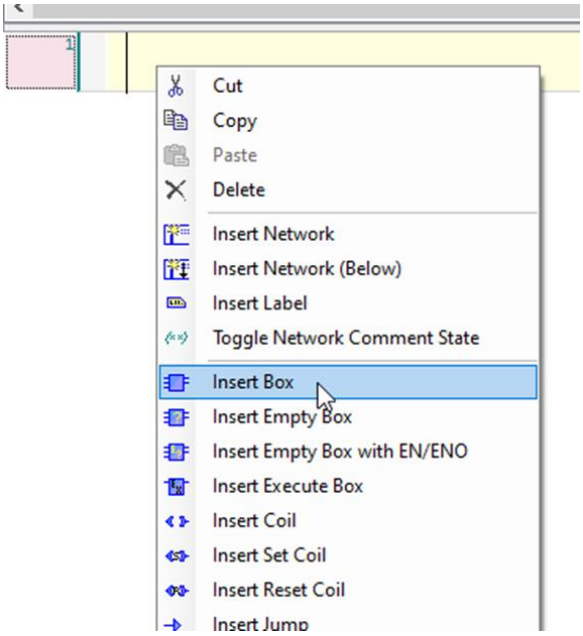
В открывшемся окне задайте название и язык программирования:



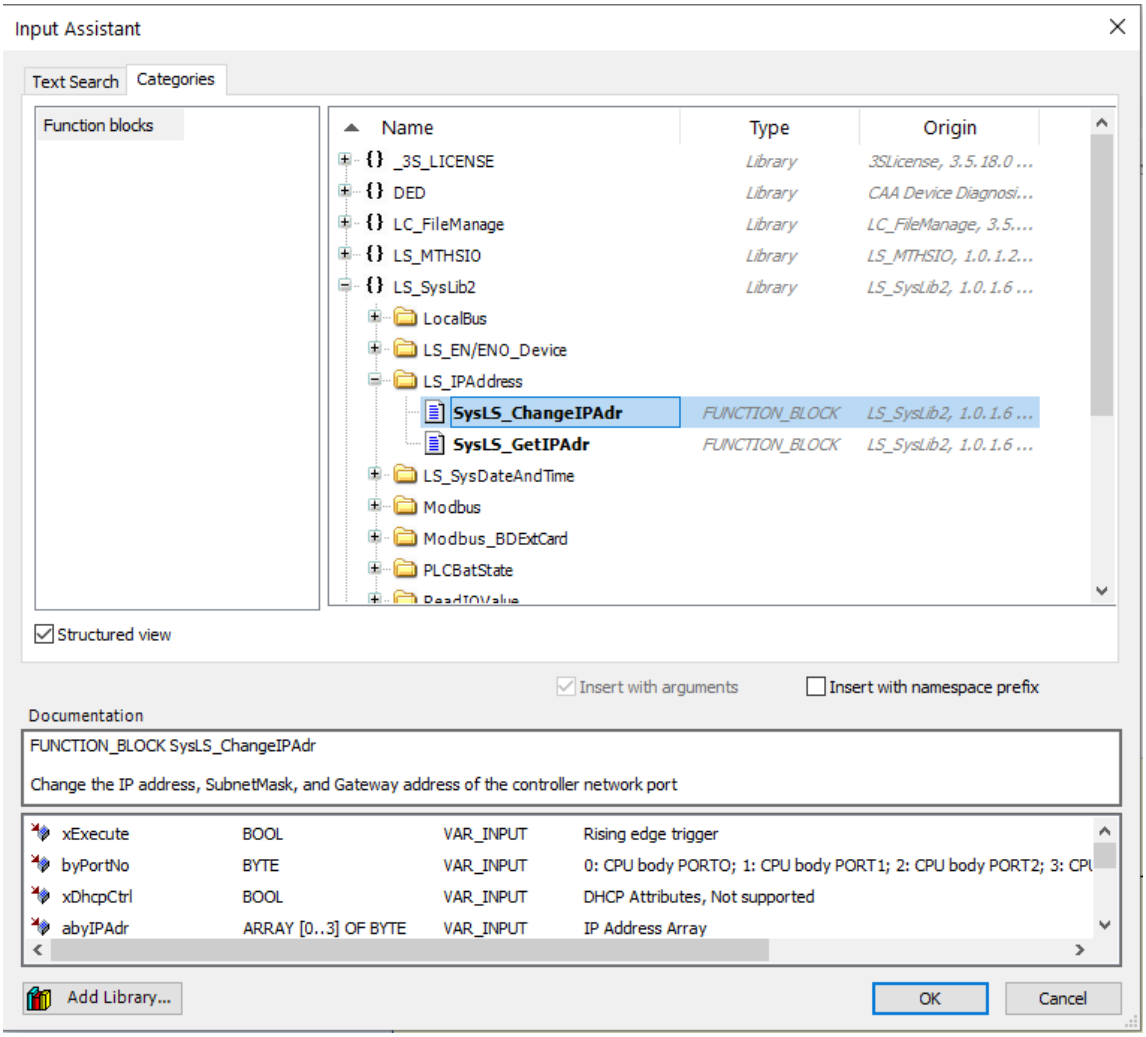
Нажмите кнопку **Add**. В древе проекта появится POU IP_Set



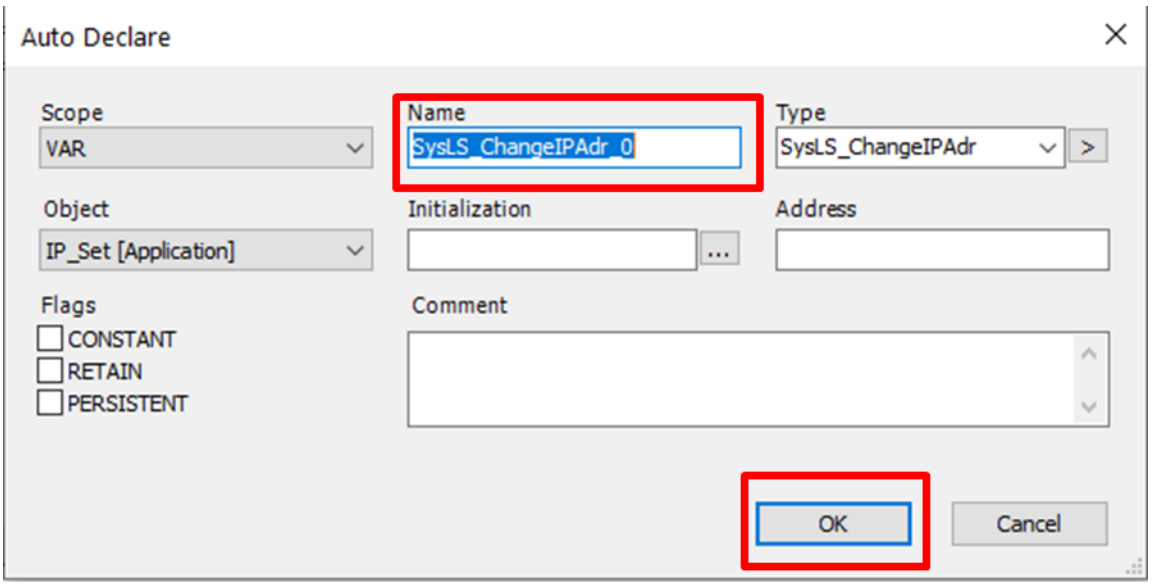
В окне созданного POU в поле ввода кода нажмите правой кнопкой мышки и выберите пункт **Insert Box**:



Откроется Мастер выбора команд. Нужно выбрать команду **SysLS_ChangeIPAdr** (библиотека LS_SysLib2)



Задайте название экземпляра ФБ и нажмите **ОК**:



Появится переменная в таблице, а команда на строке кода:

	Scope	Name	Address	Data type	Initialization
1	VAR	**SysLS_ChangeIPAdr_0**		SysLS_ChangeIPAdr	

Адрес, маска и шлюз записываются в виде байтовых массив, которые необходимо объявить и инициализировать сразу в поле объявления переменных или в программе.

```
3  VAR   abyIPAdr      ARRAY[0..3] OF BYTE
4  VAR   abyNetMask    ARRAY[0..3] OF BYTE
5  VAR   abyGateWay    ARRAY[0..3] OF BYTE
```

Инициализация в программе:

```
1  (* Инициализация массивов с адресом, маской и шлюзом. *)

   TRUE ──┤ ├──┤ EXECUTE ├──
                EN      ENO
```

```
//IP set
abyIPAdr[0]:= 192;
abyIPAdr[1]:= 168;
abyIPAdr[2]:= 1;
abyIPAdr[3]:= 4;

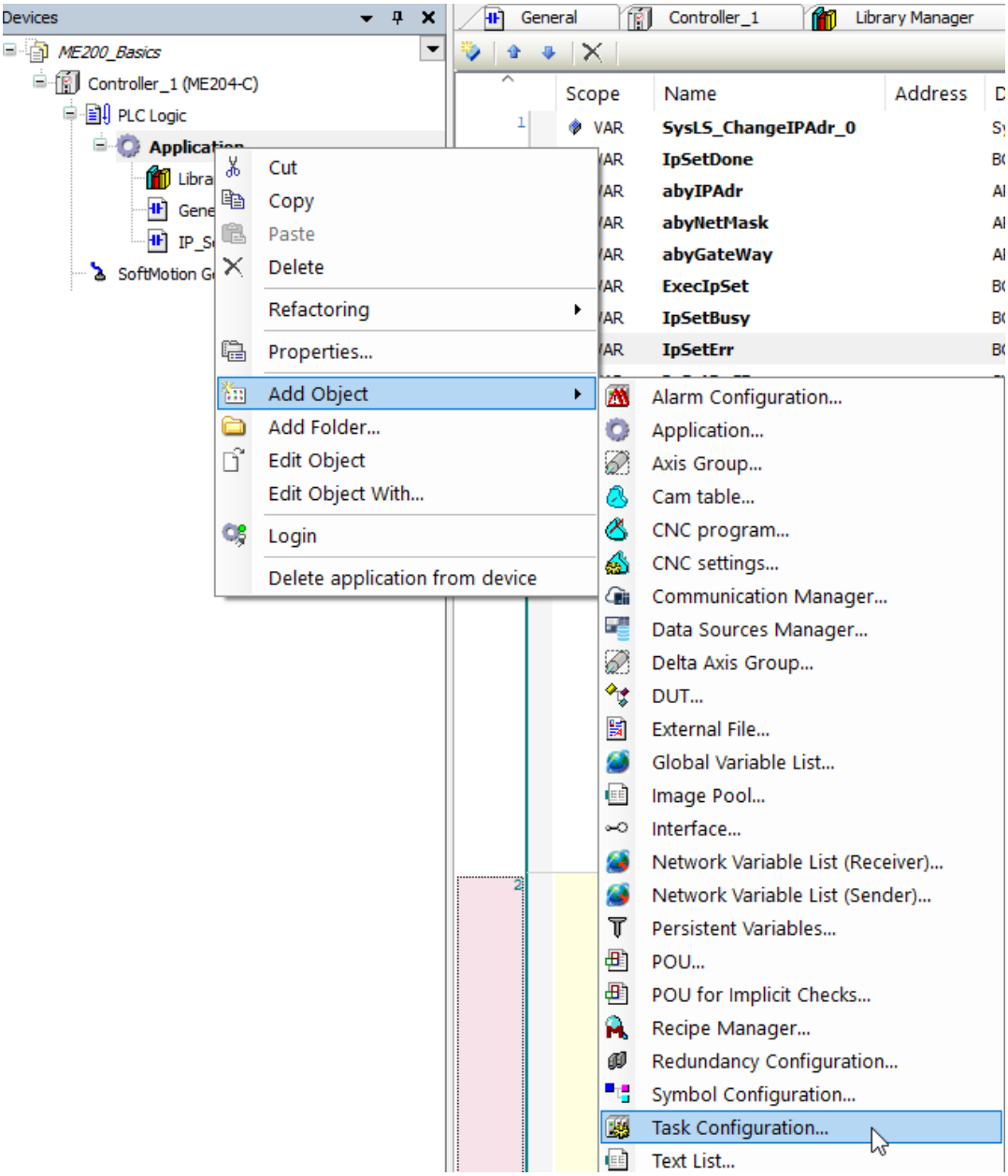
//mask set
abyNetMask[0]:=255;
abyNetMask[1]:=255;
abyNetMask[2]:=255;
abyNetMask[3]:=0;

//gate set
abyGateWay[0]:=192;
abyGateWay[1]:=168;
abyGateWay[2]:=1;
abyGateWay[3]:=1;
```

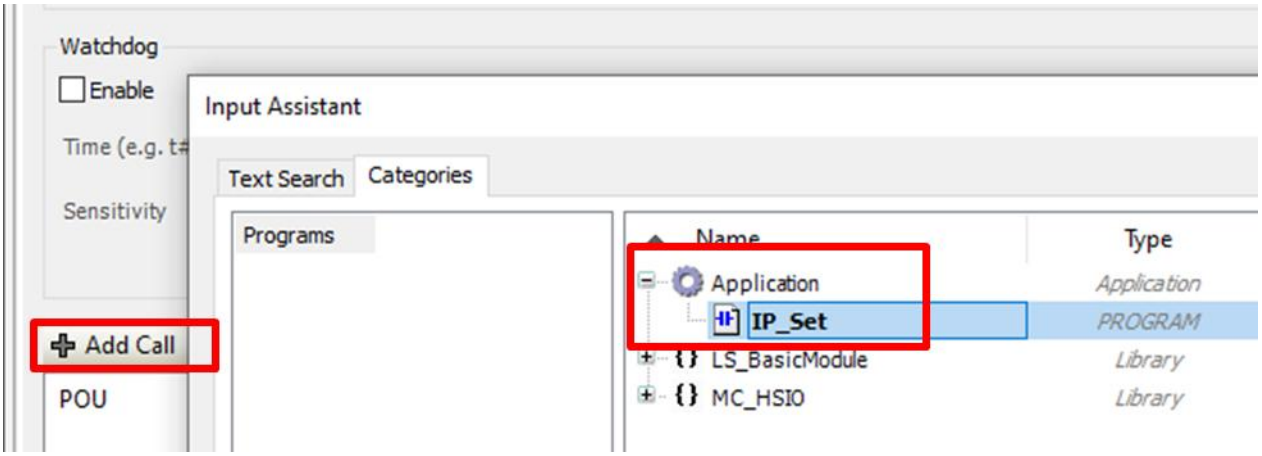
Команда установки IP адреса, маски и шлюза:

```
(* Команда установки IP адреса контроллера из программы. *)
                    SysLS_ChangeIPAdr_0
   TRUE ──┤ ├──┤ SysLS_ChangeIPAdr ├──
                EN                ENO
   ExecIpSet ──xExecute          xDone── IpSetDone
            0 ──byPortNo         xBusy── IpSetBusy
        FALSE ──xDhcpCtrl        xError── IpSetErr
     abyIPAdr ──abyIPAdr         eErrorID── IpSetErrID
  abyNetMask ──abySubnetMask
  abyGateWay ──abyDetaultGateway
```

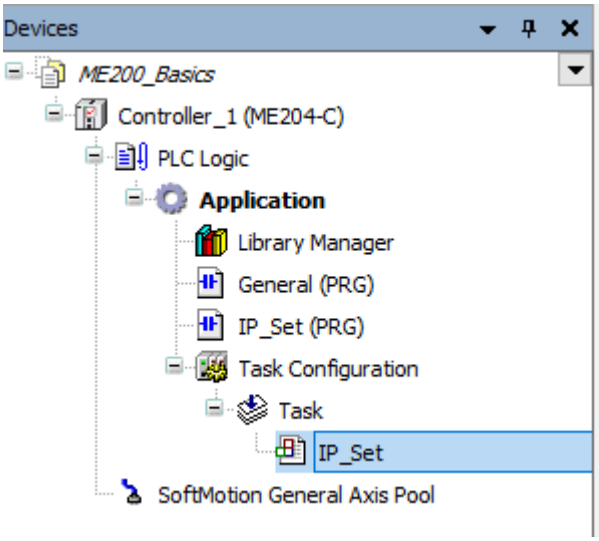
Далее необходимо добавить в проект Менеджер задач (**Task Configuration**):



и добавить в него POU IP_Set:



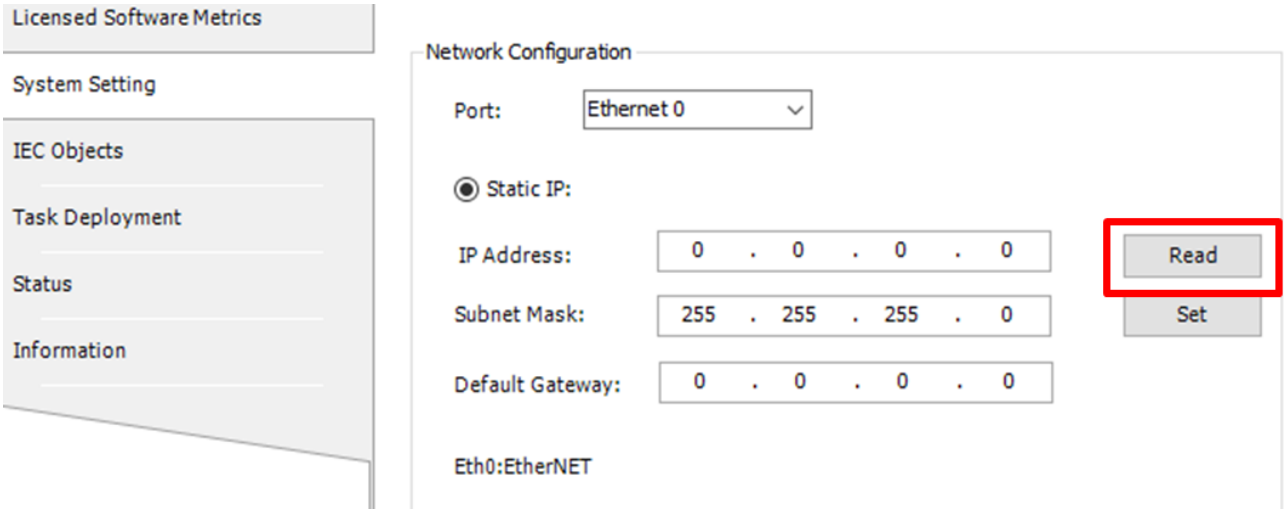
В древе проекта появится соответствующий пункт:



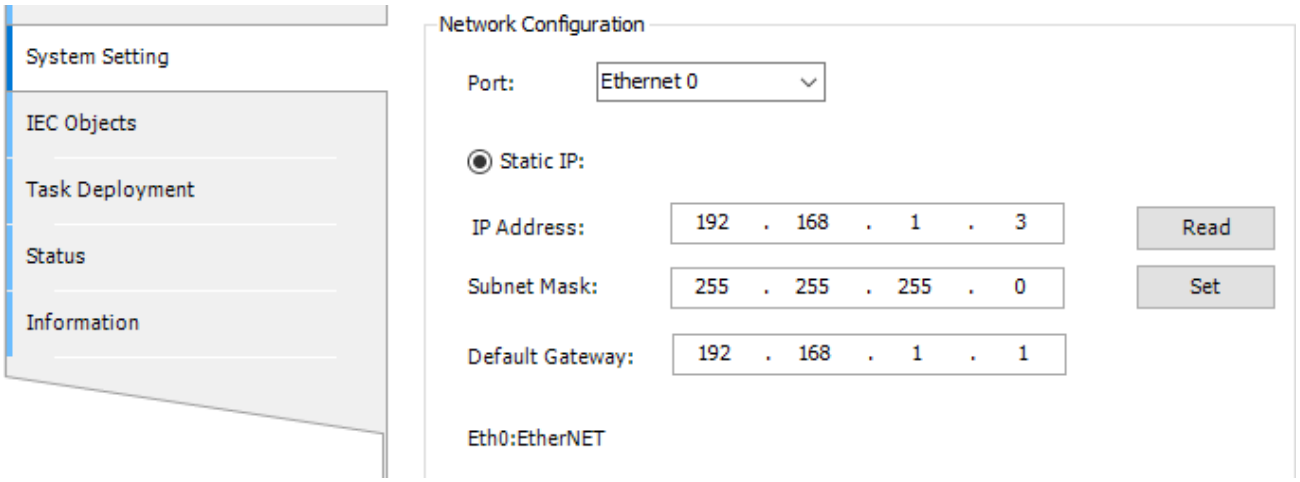
Внимание! Для активации нового IP адреса необходимо загрузить программу в контроллер и кратковременно перевести его в RUN.

Изменение IP адреса контроллера из среды программирования

Для изменения IP адреса контроллера из среды программирования необходимо открыть вкладку Device и выбрать пункт System Setting, далее поле Network Configuration. Установите связь с контроллером (см. соответствующий раздел настоящего Руководства), и нажмите кнопку **Read**:



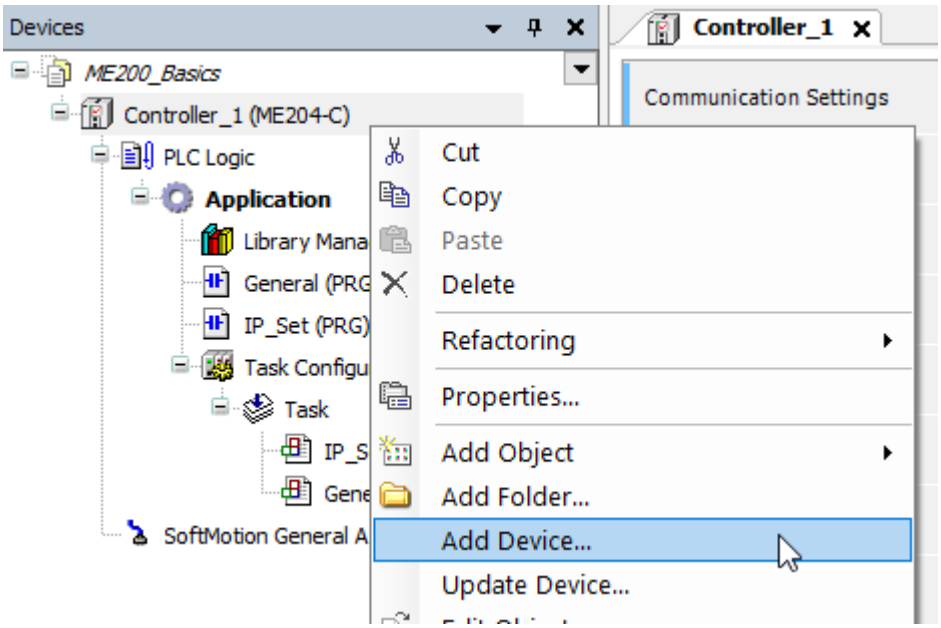
При наличии связи будет прочитан текущий IP адрес:



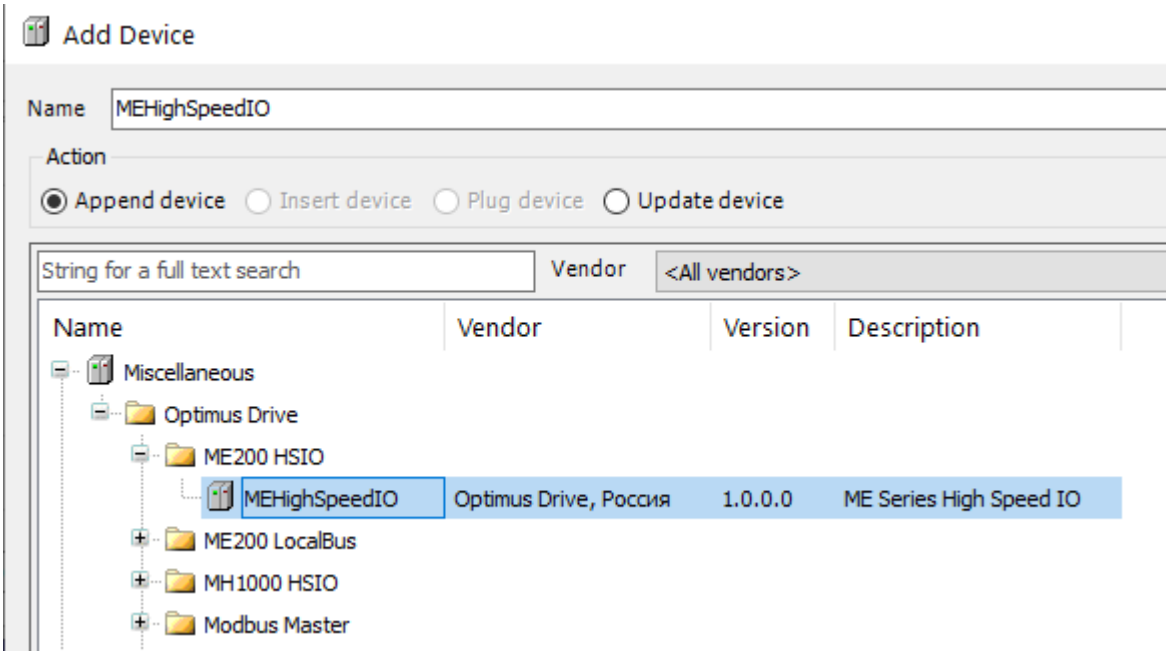
Далее задайте необходимый адрес и нажмете кнопку **Set**. После смены адреса произойдёт потеря связи между ПК и контроллером, поэтому необходимо заново установить связь ПК - контроллер.

Использование встроенных входов-выходов контроллера в обычном режиме

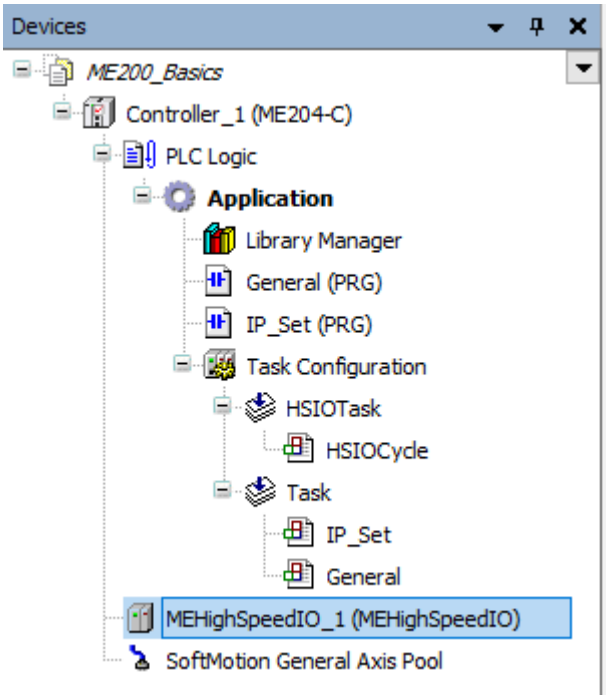
Для использования входов-выходов на ЦПУ в обычном режиме (не в импульсном) достаточно добавить в проект объект типа **High Speed IO** (идёт в составе пэкиджа) для чего щёлкните правой кнопкой мышки на контроллере в древе проекта и в отрывшемся меню выберите пункт **Add Device**:



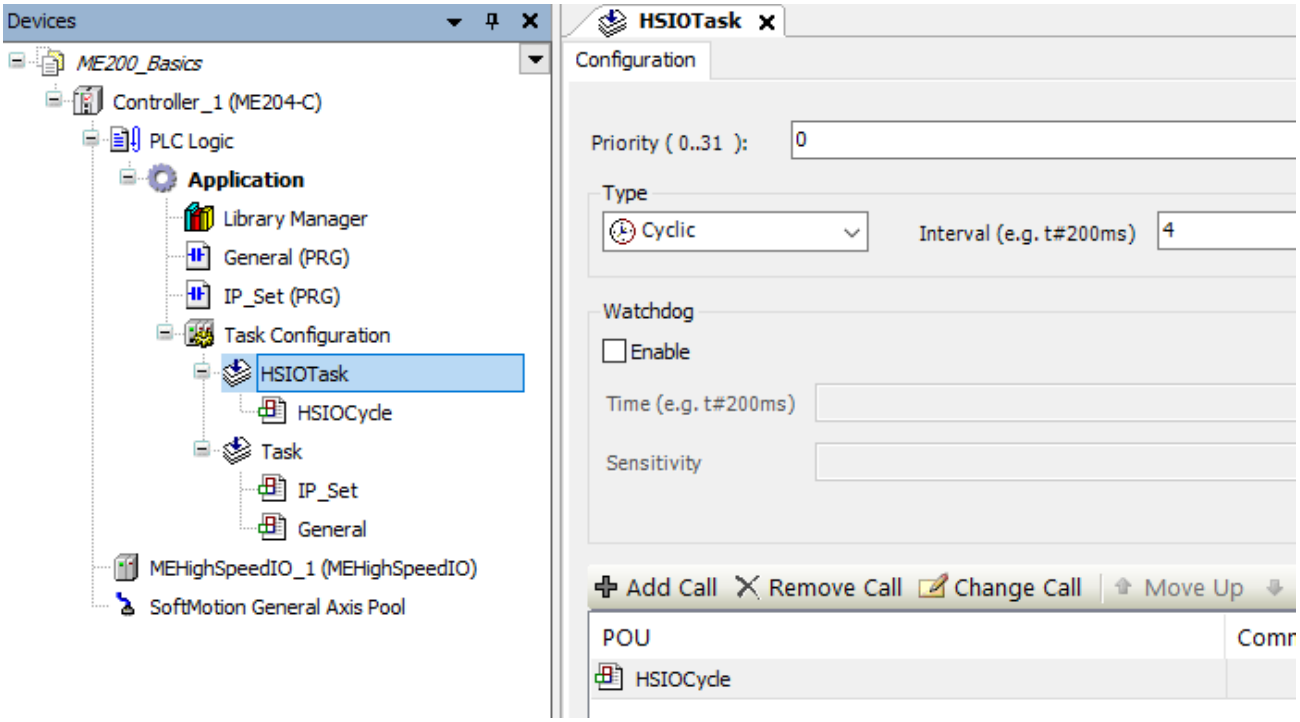
В появившемся окне выберите папку **Miscellaneous – Optimus Drive – ME200 HSIO - MEHighSpeedIO** и нажмите кнопку **Add Device**:



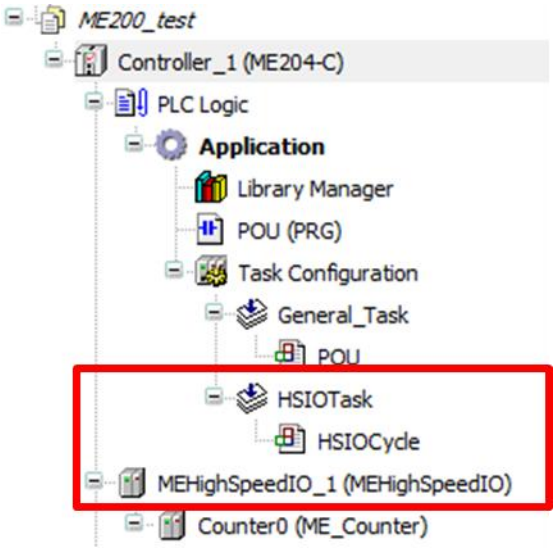
В древе проекта появится пункт **High Speed IO**:

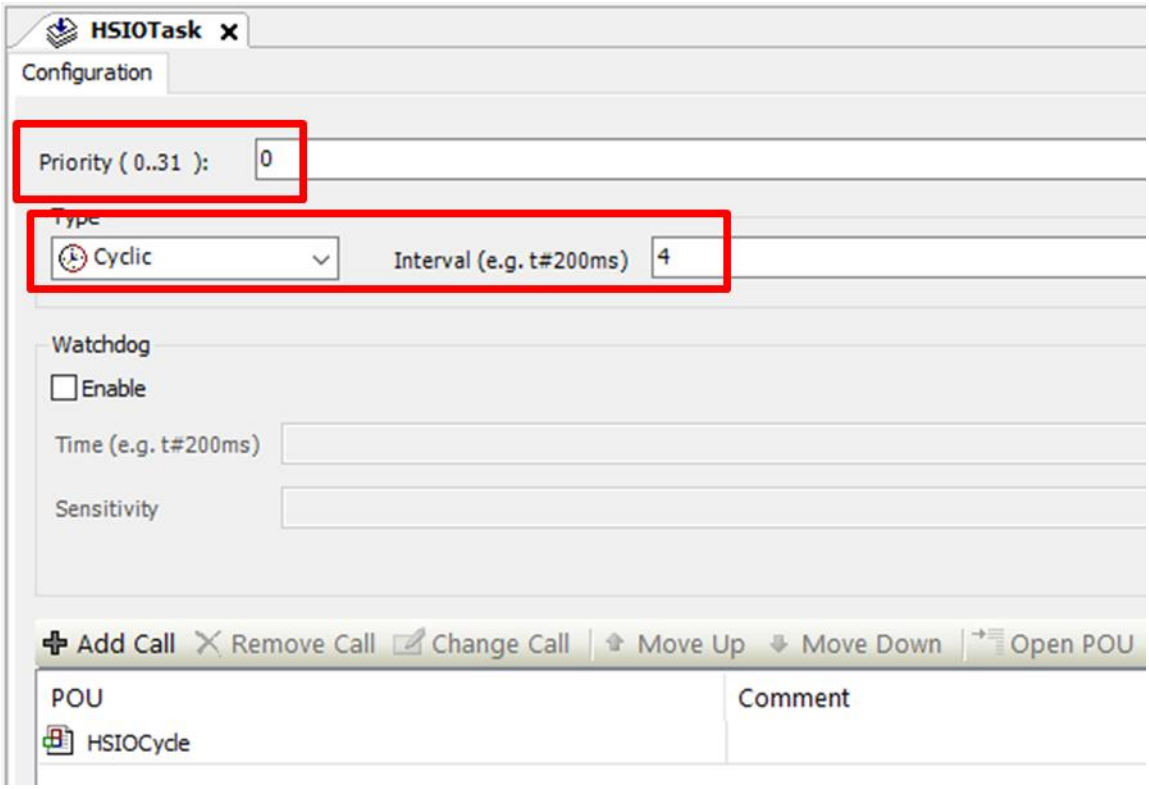


Также, в разделе **Task Configuration** (Менеджер Задач) появится пункт **HSIOTask – HSIOCycle**:

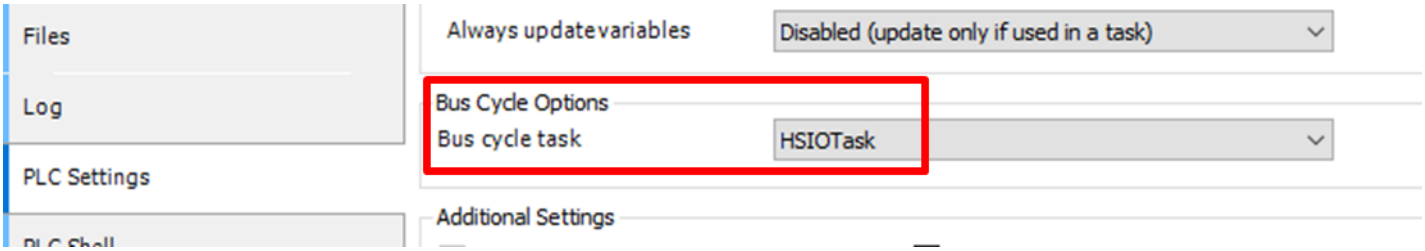


В контроллерах серии ME200 задача **HSIOTask – HSIOCycle** является основной и должна иметь приоритет 0. Время цикла рекомендуется оставить 4 мс.



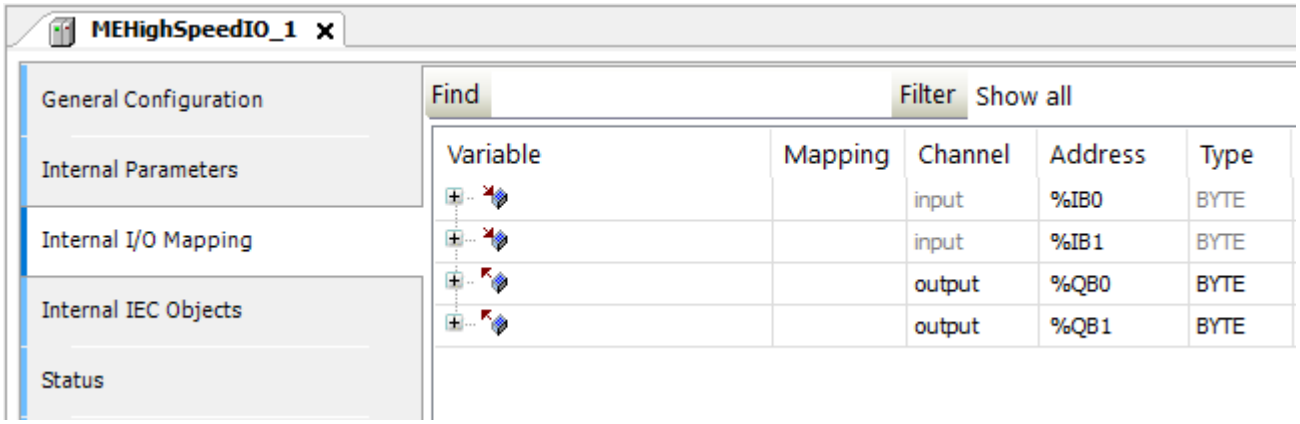


Данную задачу необходимо назначить во вкладке **Device** пункт **PLC Setting**:

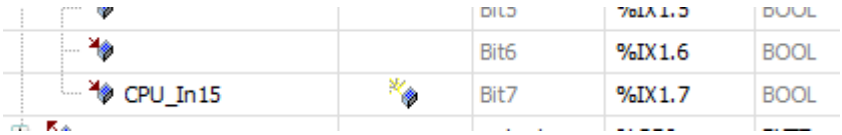


В задаче **HSIOTask – HSIOCycle** необходимо размещать критические ко времени процессы типа высокоскоростных счётчиков, импульсных осей, а также все инструкции управления движением.

Для работы с входами и выходами на ЦПУ в обычном режиме достаточно открыть вкладку **Internal I/O Mapping**:



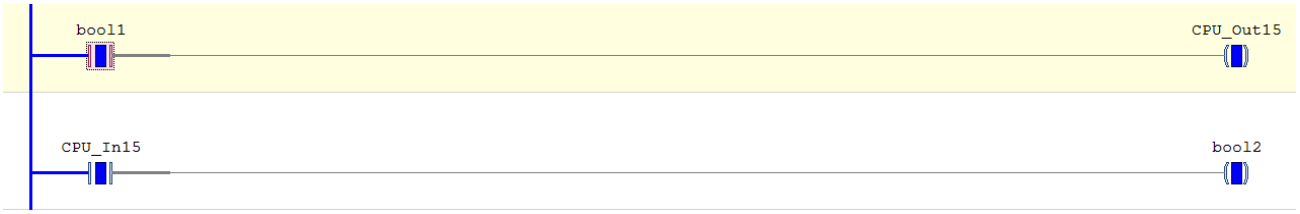
Развернуть нужный список и в колонке **Variable** задать имена переменных для входов и выходов:





Заполните колонку тегов путём подставки по префиксу, или просто вручную своих названий:

В программе состояние входов можно считывать как по тегам, так и по физическим адресам.

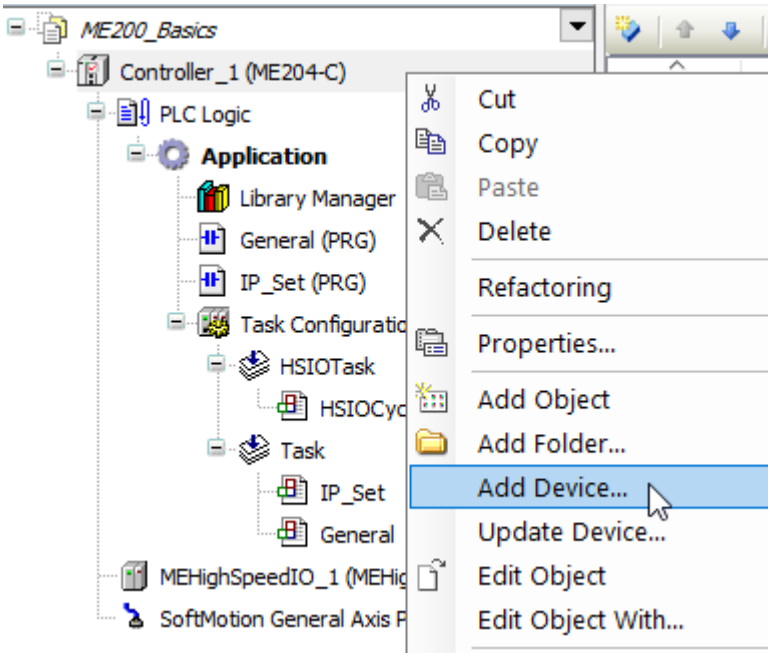


Добавление в проект модулей расширения

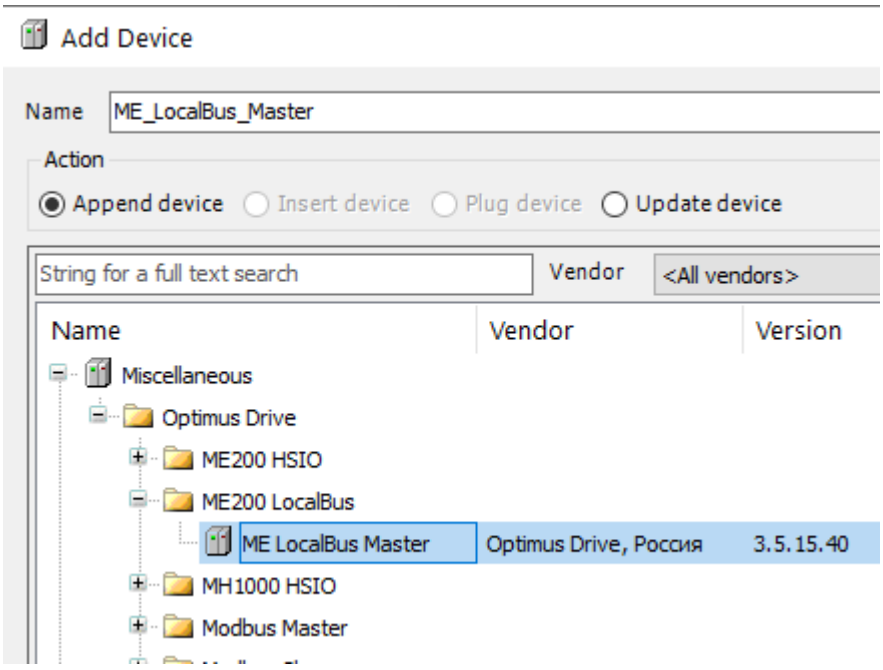
Контроллеры серии ME200 могут расширяться модулями дискретного и аналогового ввода-вывода. Всего к контроллеру по внутренней шине могут быть подключены до 16-ти модулей расширения.

Файлы с описанием устройств (XML) устанавливаются в составе пэкиджа (см. выше).

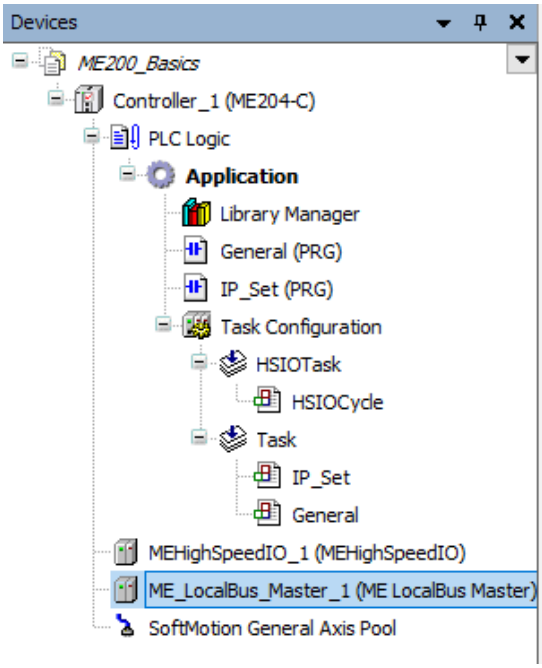
Для добавления модуля расширения в проект щёлкните правой кнопкой мышки на контроллере в древе проекта и в отрывшемся меню выберите пункт **Add Device**:



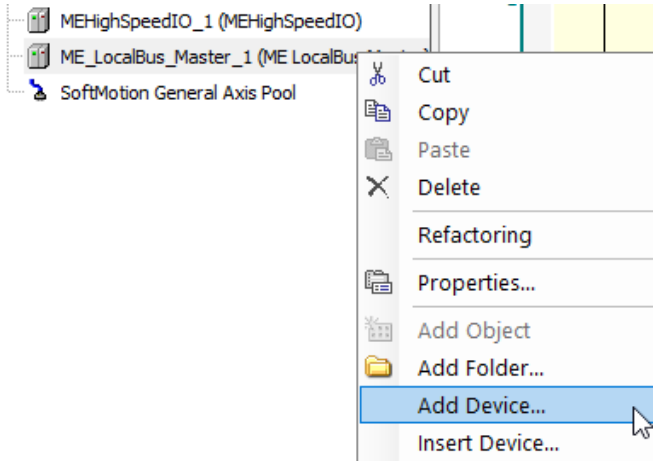
В появившемся окне выберите папку **Miscellaneous – ME200 LocalBus – ME LocalBus Master** и нажмите кнопку **Add Device**:



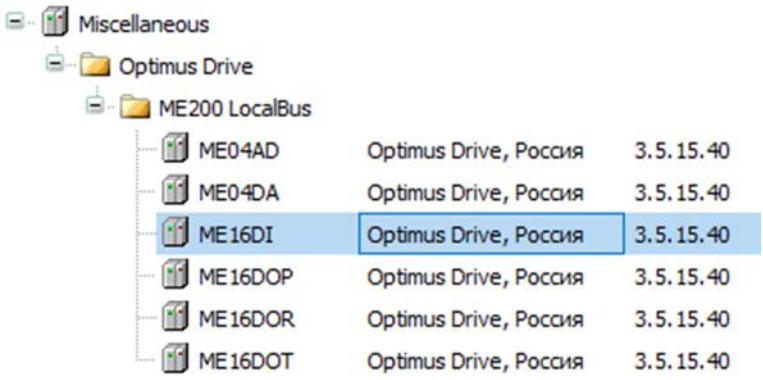
В древе проекта появится пункт **ME_LocalBus Master**:



Щёлкните правой кнопкой мышки на пункте **ME_LocalBus Master** и в отрывшемся меню выберите пункт **Add Device**:

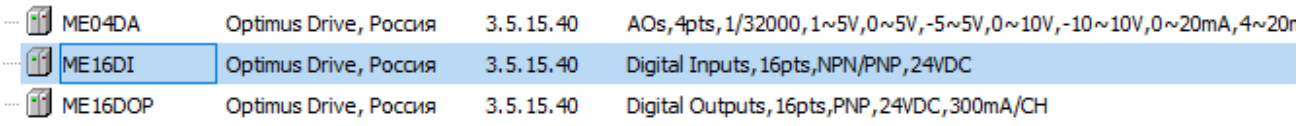


В открывшемся окне выберите нужный модуль:

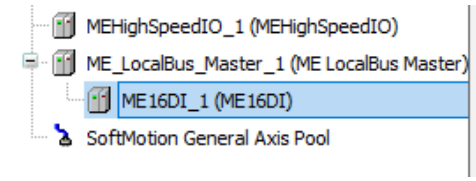


Работа с модулем дискретных входов MX16DI

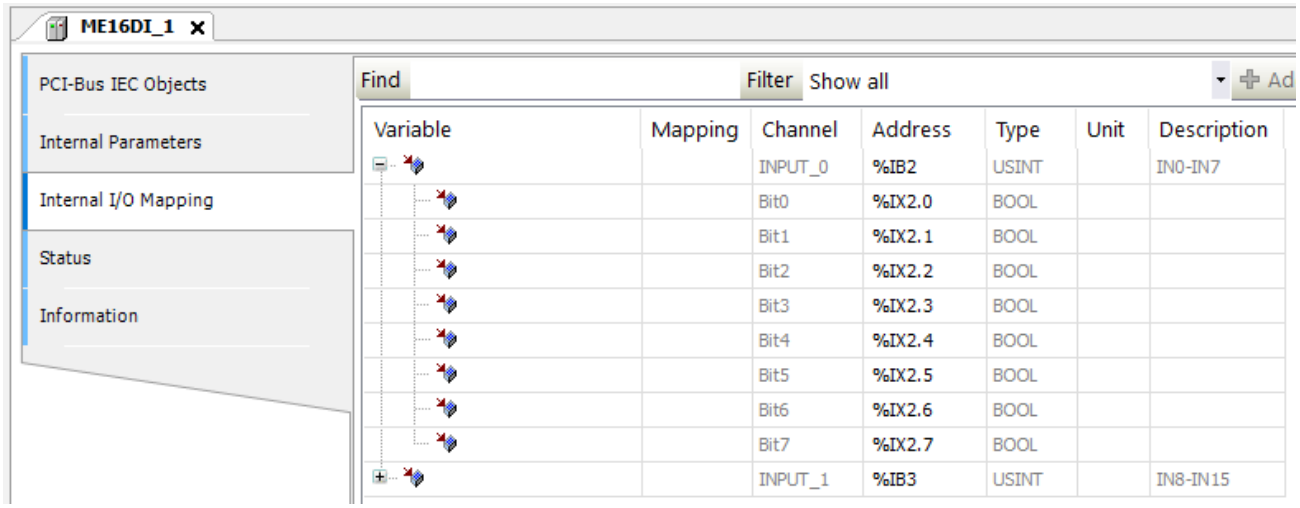
Добавьте в проект по вышеприведённой последовательности модуль MX16DI:



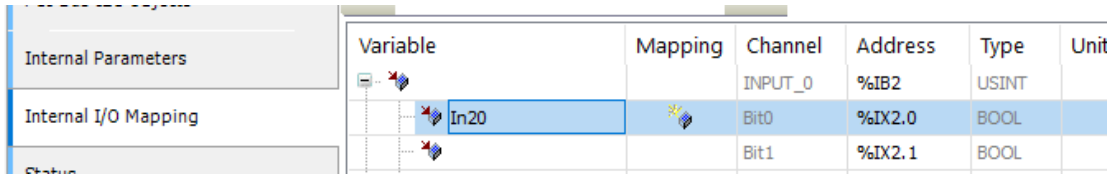
В проекте появится пункт:



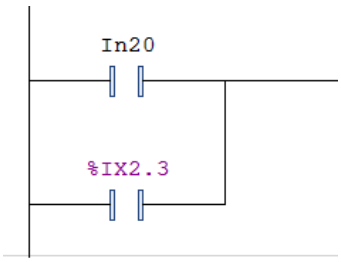
Щёлкните на пункте MX16DI дважды левой кнопкой мышки, отроется вкладка параметров модуля. Выберите в ней пункт Internal I/O Mapping, в которой будут видны физические адреса входов и можно задать тэги для входов.



Для этого разверните список и в колонке **Variable** задайте названия переменным:

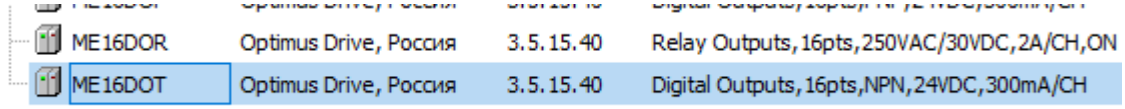


В программе состояние входов можно считывать как по тегам, так и по физическим адресам.

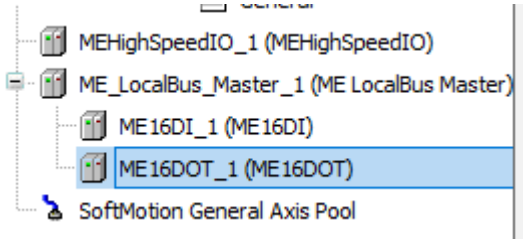


Работа с модулями дискретных выходов ME16DOT/ME16DOP/ME16DOR

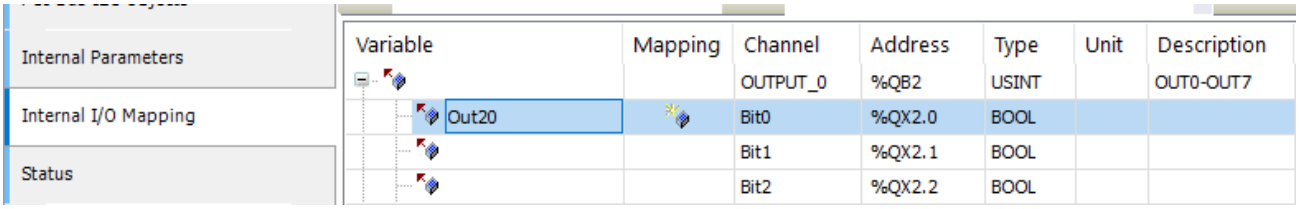
Добавьте в проект по вышеприведённой последовательности модуль ME16DOT:
(модули ME16DOP и ME16DOR настраиваются аналогично)



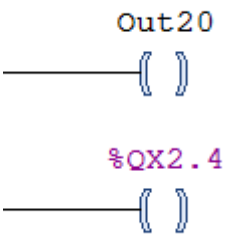
В проекте появится пункт:



Щёлкните на пункте MX16DOT дважды левой кнопкой мышки, отроется вкладка параметров модуля. Выберите в ней пункт Internal I/O Mapping, в которой будут видны физические адреса выходов и можно задать тэги для выходов. Для этого разверните список и в колонке Variables задайте имена переменным.

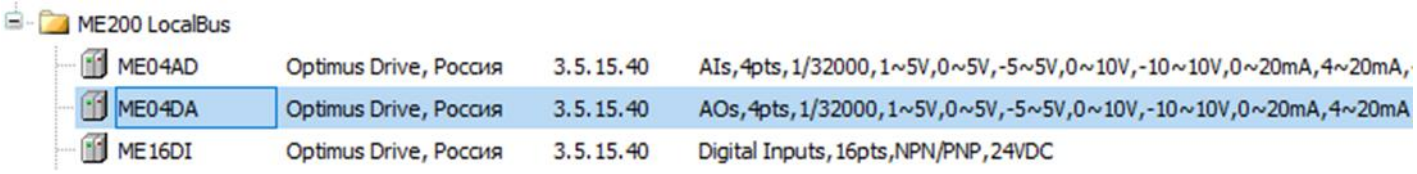


В программе состояние входов можно считывать как по тегам, так и по физическим адресам.

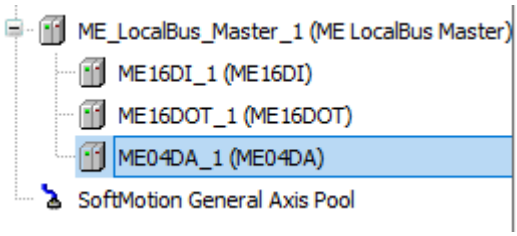


Работа с модулем аналоговых выходов ME04DA

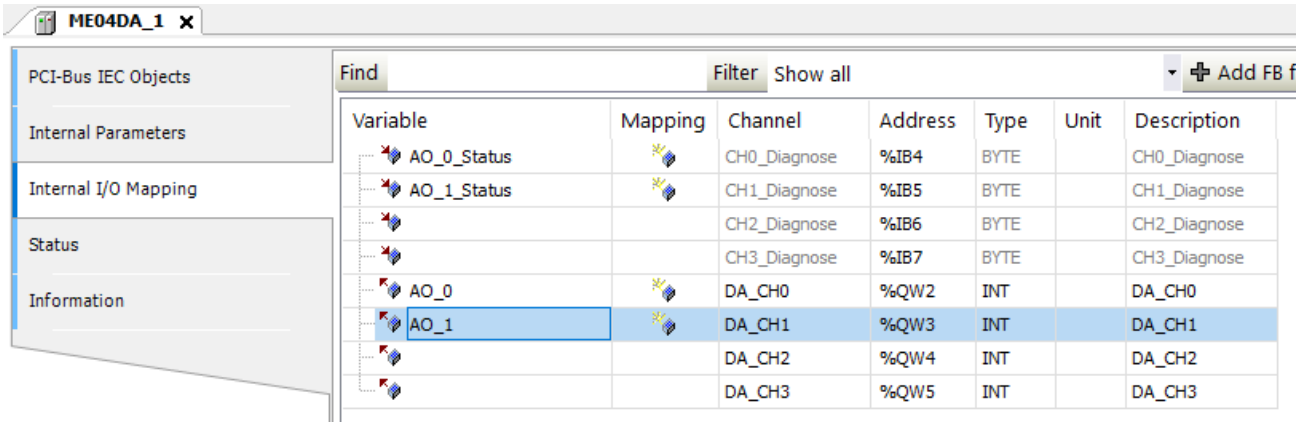
Добавьте в проект по вышеприведённой последовательности модуль ME04DA:



В проекте появится пункт:

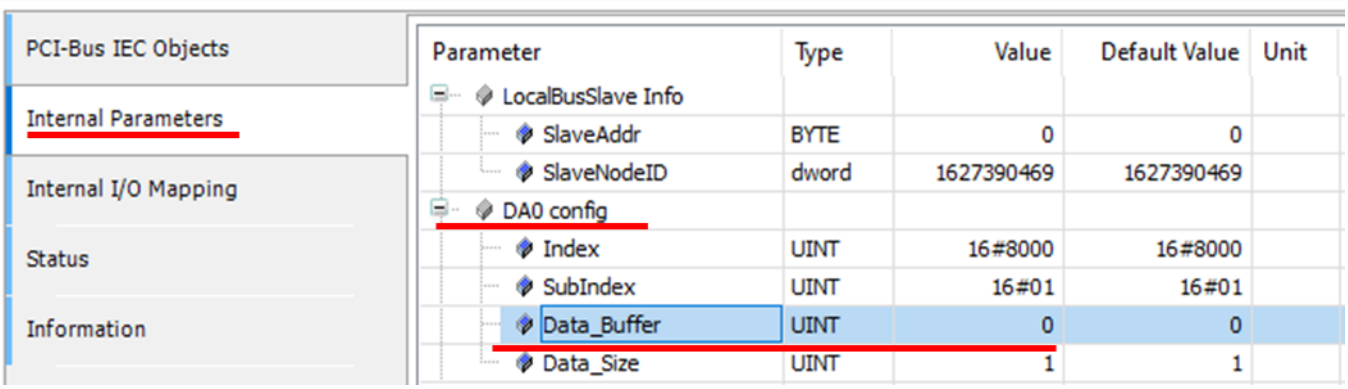


Щёлкните на пункте MX04DA дважды левой кнопкой мышки, отроется вкладка параметров модуля. Выберите в ней пункт Internal I/O Mapping, в которой будут видны физические адреса выходов и можно задать тэги для выходов. Для этого разверните список и введите нужное название тега.



В программе к аналоговым выходам можно обращаться как по тегам, так и по физическому адресу.

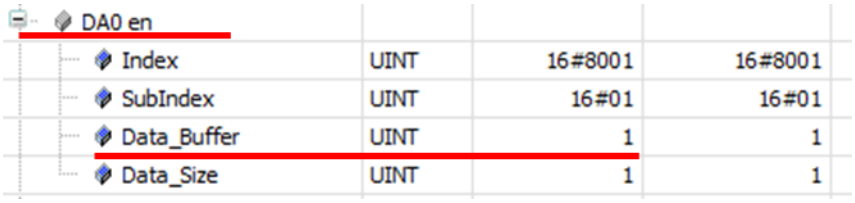
Для выбора режима работы аналогового выхода необходимо открыть вкладку **Internal Parameters**. Выберите раздел типа **DA0 Config**. Для каждого канала DA0 – DA3 предусмотрен свой раздел. В данном разделе выберите пункт **Data Buffer**.



В пункт **Data Buffer** вносится номер режима работы в соответствии с нижеприведённой таблицей:

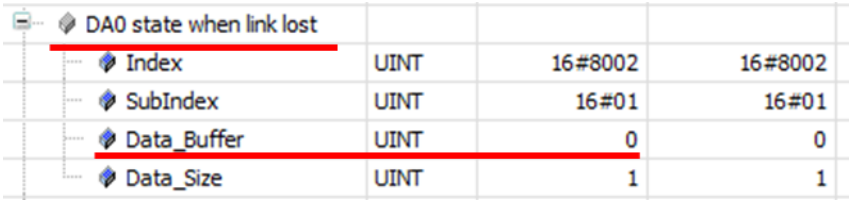
Значение в Data_Buffer	Тип аналогового сигнала	Диапазон цифровой шкалы
0	0~5V	0~32000
1	1~5V,	0~32000
2	-5~5V,	-32000~32000
3	0~10V	0~32000
4	-10~+10V	-32000~32000
5	0~20mA	0~32000
6	4~20mA	0~32000

Далее, в разделе типа **DA0_en** в пункте **Data Buffer** для разрешения работы канала нужно выставить 1, а для запрета 0. По умолчанию работа каналов разрешена.



DA0 en			
Index	UINT	16#8001	16#8001
SubIndex	UINT	16#01	16#01
Data_Buffer	UINT	1	1
Data_Size	UINT	1	1

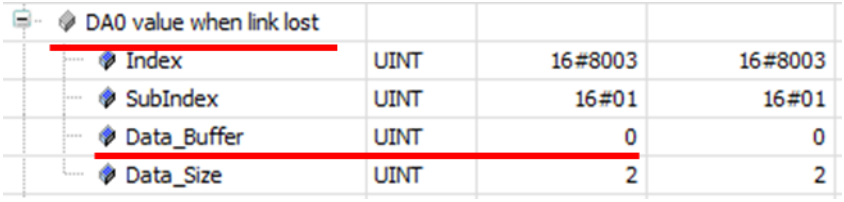
В разделе типа **DA0 state when link lost** в пункте **Data Buffer** можно установить реакцию канала на потерю связи:



DA0 state when link lost			
Index	UINT	16#8002	16#8002
SubIndex	UINT	16#01	16#01
Data_Buffer	UINT	0	0
Data_Size	UINT	1	1

Значение в Data_Buffer	Реакция канала
0	Сохранять текущее значение
1	Сброс на 0
2	Выдача заранее предустановленного значения

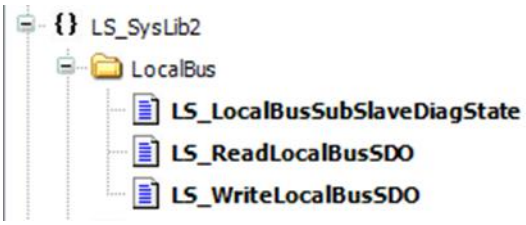
Предустановленное значение выставляется в разделе типа **DA0 value when link lost**:



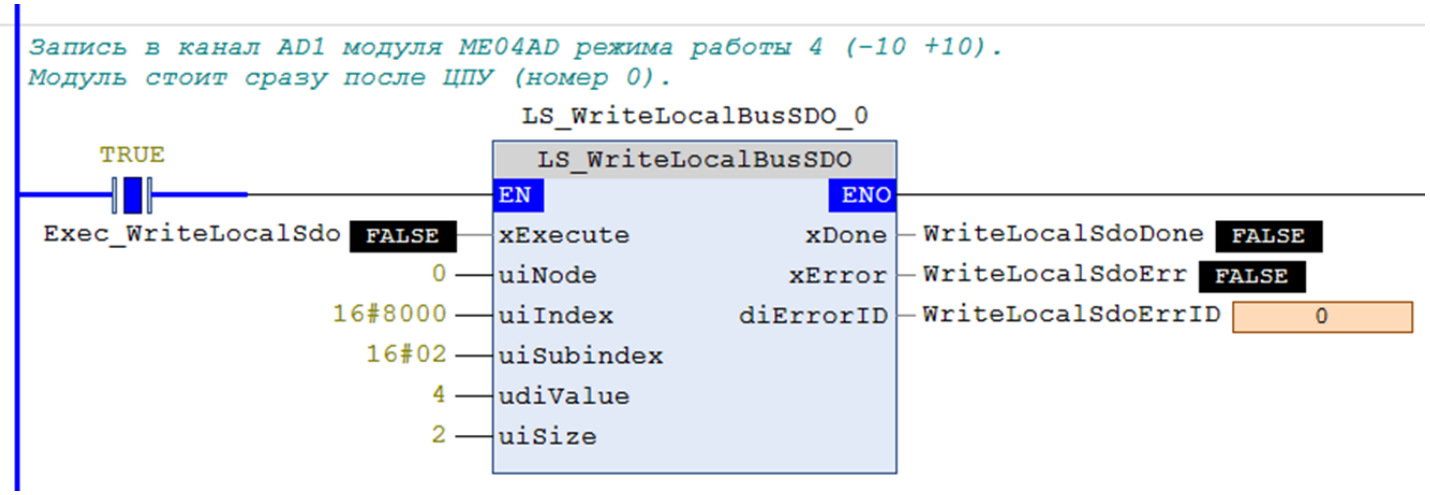
DA0 value when link lost			
Index	UINT	16#8003	16#8003
SubIndex	UINT	16#01	16#01
Data_Buffer	UINT	0	0
Data_Size	UINT	2	2

Например, если для канала выбран режим 0~10 VDC, то при установке в Data_Buffer значения 27200 на аналоговом выходе при пропадании связи с модулем будет 8.50 VDC.

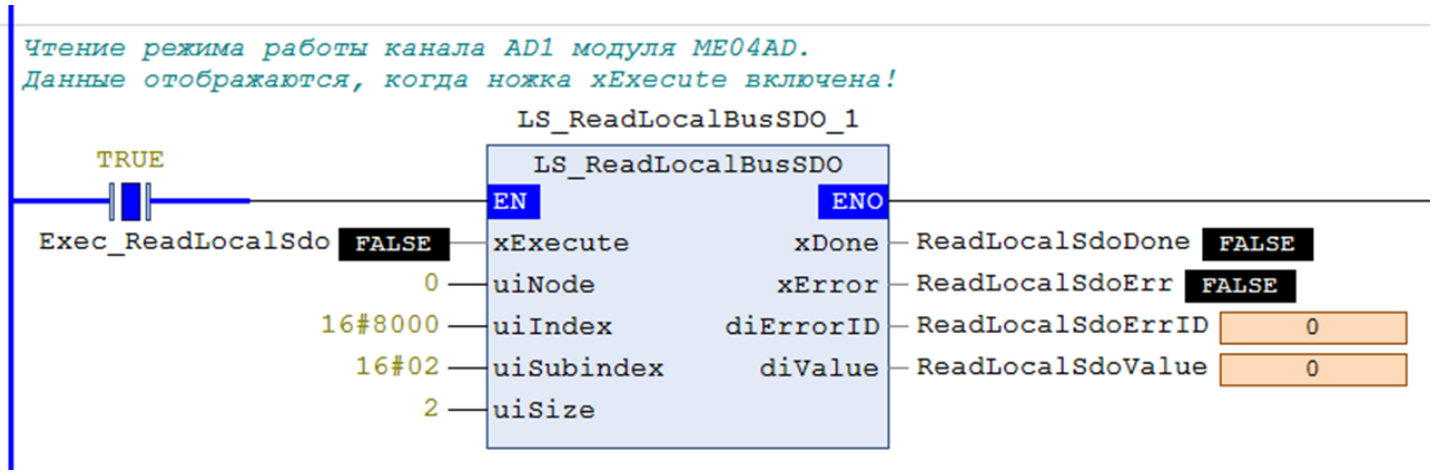
Модуль можно конфигурировать и из программы. Для этого используются команды типа:
LS_ReadLocalBusSDO
LS_WriteLocalBusSDO

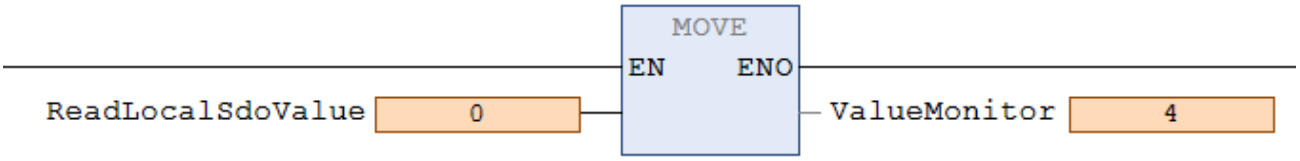


Например, запись режима 4 и чтение обратно для проверки:
(нумерация модулей (uiNode) начинается с нуля и далее до 15 по степени удаления от ЦПУ на шине)

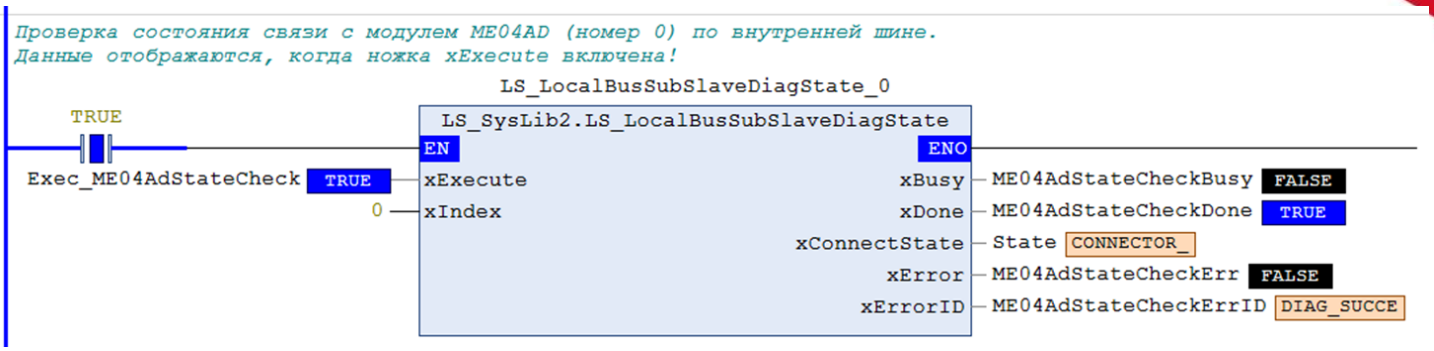


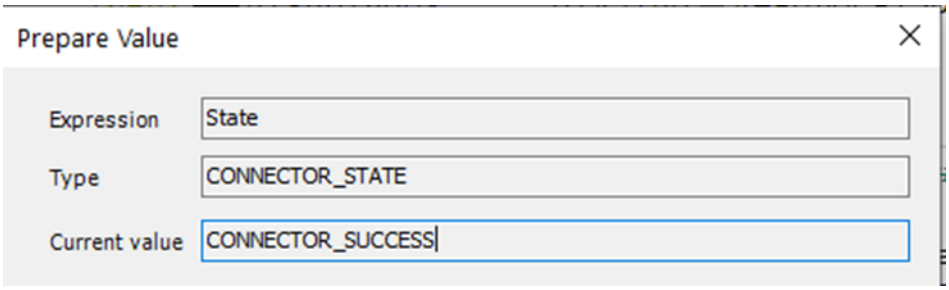
Чтение (данные отображаются, когда ножка xExecute включена):





Для контроля состояния связи с модулем по внутренней шине используется команда
LS_LocalBusSubSlaveDiagState (данные отображаются, когда ножка xExecute включена)

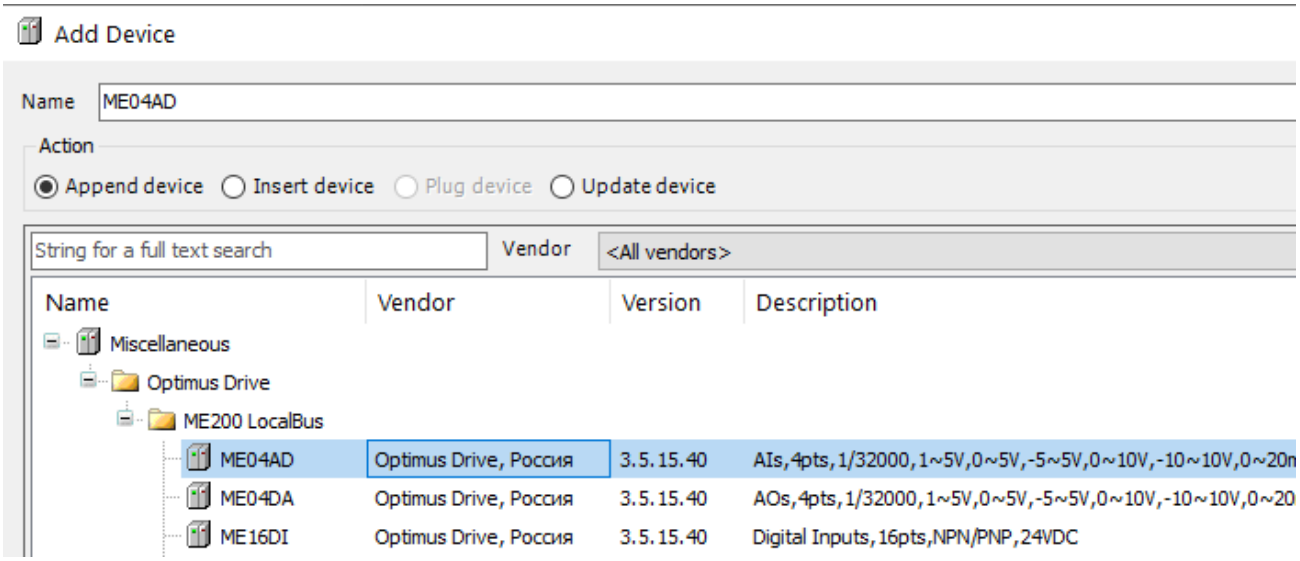




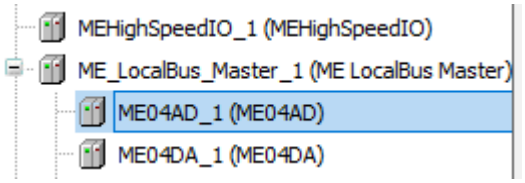
Внимание!
*В текущей версии программного обеспечения при нахождении в онлайн режиме параметры кнопкой **Write Parameters** не записываются. Т.е. необходимо загрузить проект с нужными настройками из среды программирования, или использовать команды записи из программы.*

Работа с модулем аналоговых входов ME04AD

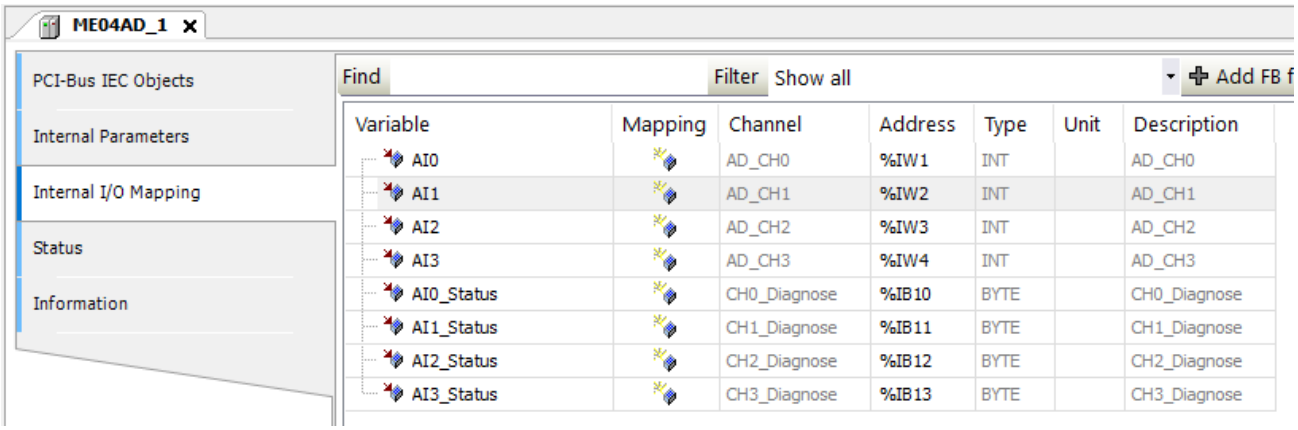
Добавьте в проект по вышеприведённой последовательности модуль MX04AD:



В проекте появится пункт:

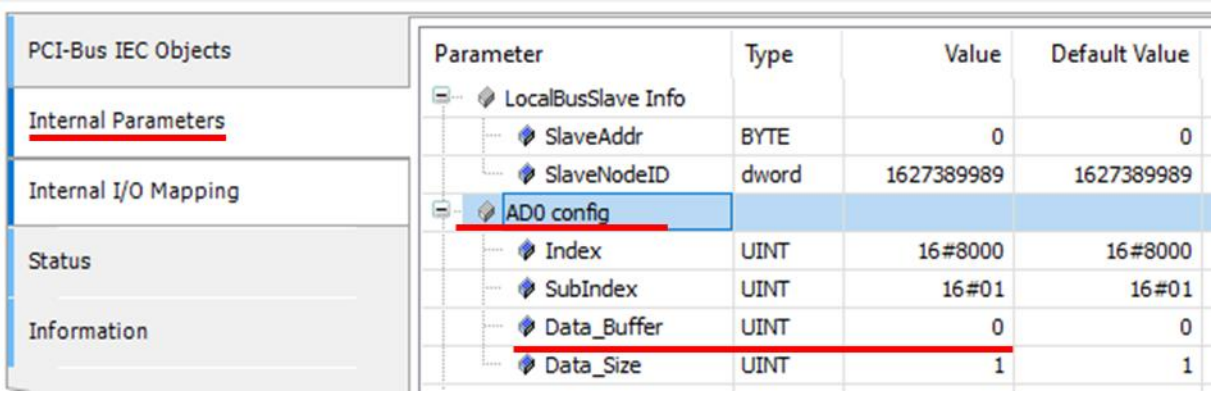


Щёлкните на пункте ME04AD дважды левой кнопкой мышки, отроется вкладка параметров модуля. Выберите в ней пункт Internal I/O Mapping, в которой будут видны физические адреса выходов и можно задать тэги для входов. Для этого разверните список и введите в колонке **Variable** нужное название тега:



В программе к аналоговым входам можно обращаться как по тегам, так и по физическому адресу.

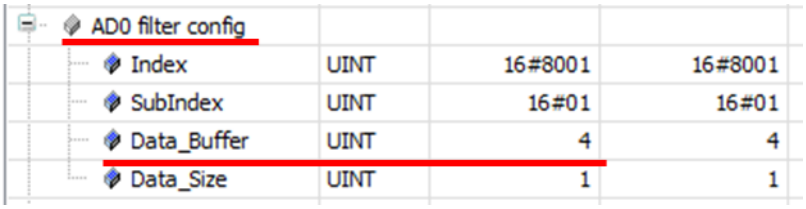
Для выбора режима работы аналогового входа необходимо открыть вкладку **Internal Parameters**. Выберите раздел типа **AD0 Config**. Для каждого канала AD0 – AD3 предусмотрен свой раздел. В данном разделе выберите пункт **Data Buffer**.



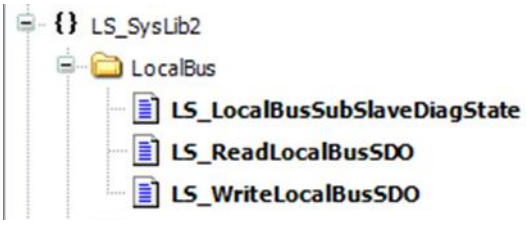
В пункт **Data Buffer** вносится номер режима работы в соответствии с нижеприведённой таблицей:

Значение в Data_Buffer	Тип аналогового сигнала	Диапазон цифровой шкалы
0	-5~5V,	-32000~32000
1	1~5V,	0~32000
2	-10~+10V	-32000~32000
3	0~10V	0~32000
4	0~20mA	0~32000
5	4~20mA	0~32000
6	0~5V	0~32000
7	-20mA~20mA	-32000~32000

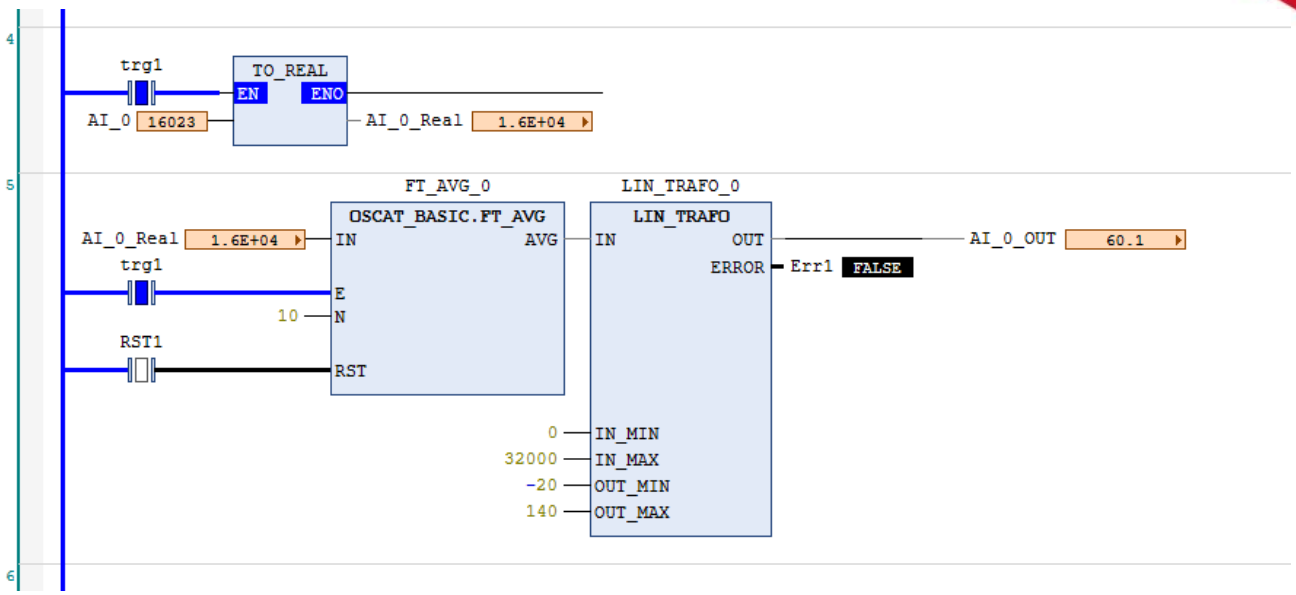
Далее, в разделе типа **AD0 filter config** в пункте **Data Buffer** нужно выставить цикл опроса входа в мс. По умолчанию стоит 4 мс.



Модуль можно конфигурировать и из программы. Для этого используются команды типа:
LS_ReadLocalBusSDO
LS_WriteLocalBusSDO
(см. пример в разделе модуля аналоговых выходов)



Для осреднения входных значений по принципу «бегущее среднее» можно использовать ФБ **FT_AVG** из библиотеки **OSCAT_BASIC**. Для масштабирования шкалы АЦП а шкалу требуемой физической величины можно использовать ФБ **LIN_TRAFO** из библиотеки **Util**.



Состав библиотек можно посмотреть в Library Manager (древо проекта).

Name	Namespace	Effective Version
3SLicense = 3SLicense, 3.5.18.0 (3S - Smart Software Solutions GmbH)	_3S_LICENSE	3.5.18.0
BreakpointLogging = Breakpoint Logging Functions, 3.5.17.0 (3S - Smart Software Solutions GmbH)	BPLog	3.5.17.0
CAA Device Diagnosis = CAA Device Diagnosis, 3.5.15.0 (CAA Technical Workgroup)	DED	3.5.15.0
CAA NetBaseSrv = CAA Net Base Services, 3.5.15.0 (CAA Technical Workgroup)	NBS	3.5.15.0
IoStandard = IoStandard, 3.5.15.0 (System)	IoStandard	3.5.15.0
LS_BasicModule = LS_BasicModule, 1.0.0.5 (Leadshine Technology Co.Ltd)	LS_BasicModule	1.0.0.5
MC_HSIO = MC_HSIO, 1.0.2.3 (Leadshine Technology Co.Ltd)	MC_HSIO	1.0.2.3
MC_SysLib = MC_SysLib, 1.0.0.8 (Leadshine Technology Co.Ltd)	MC_SysLib	1.0.0.8
OSCAT_BASIC = BASIC, 3.3.4.0 (OSCAT)	OSCAT_BASIC	3.3.4.0
SM3_Basic = SM3_Basic, 4.10.0.0 (3S - Smart Software Solutions GmbH)	SM3_Basic	4.10.0.0
SM3_CNC = SM3_CNC, 4.10.0.0 (3S - Smart Software Solutions GmbH)	SM3_CNC	4.10.0.0
SM3_Robotics = SM3_Robotics, 4.10.0.0 (3S - Smart Software Solutions GmbH)	SM3_Robotics	4.10.0.0
SM3_Robotics_Visu = SM3_Robotics_Visu, 4.10.0.0 (3S - Smart Software Solutions GmbH)	SM3_Robotics_Visu	4.10.0.0
SM3_Transformation = SM3_Transformation, 4.10.0.0 (3S - Smart Software Solutions GmbH)	TRAFO	4.10.0.0
Util = Util, 3.5.18.0 (System)	Util	3.5.18.0

При отсутствии данных библиотек их необходимо инсталлировать и подключить к проекту.

Поддерживаемые базовые типы данных

Категории данных	Тип данных	Ключевые слова	Количество бит занимаемой памяти	Диапазоны
Логический тип	логическое значение	BIT	1	0 или 1
	логическое значение	BOOL	8	ЛОЖЬ(0) или ИСТИНА(1)
Целочислен-ный тип	байт	BYTE	8	0 ~ 255
	слово	WORD	16	0 ～ 65535
	двойное слово	DWORD	32	0 ~ 4294967295
	длинное слово	LWORD	64	0 ~ (2^64-1)
	короткий	SINT	8	- 128 ~ 127
	короткий беззнаковый	USINT	8	0 ~ 255
	целое число	INT	16	- 32768 ~ 32767
	целое число без знака	UINT	16	0 ～ 65535
	двойное целое число	DINT	32	- 2147483648 ~ 2147483647
	беззнаковое двойное целое	UDINT	32	0 ~ 4294967295
	длинное целое	LINT	64	-2^63 ～ (2^63- 1)
	длинное целое число без знака	ULINT	64	0 ~ (2^64-1)
Тип с плавающей точкой	одинарная точность	REAL	32	1.175494351e- 38 ~ 3.402823466e+38
	двойная точность	LREAL	64	2.2250738585072014e- 308 ~ 1.7976931348623158e+308
Строка	строка	STRING	8*N бит	
Строка	строка Юникод	WSTRING	16*N бит	
Время		TIME	32	T#0ms~T#71582m47s295ms
Время дня		TIME_OF_DAY	32	TOD#0:0:0~TOD#1193:02:47.295
Дата		DATE	32	D#1970-1-1~D#2106-02-06
Дата и Время		DATE_AND_TIME	32	DT#1970-1-1-0:0:0~DT#2106-02-06-06: 28:15

Примечание.
DesignerAX и CODESYS 3.5.18.30 и выше версией поддерживают также новые типы данных и интерфейсов. В новых версиях среды программирования ряд инструкций поддерживают только новые типы данных

Список наиболее употребительных команд

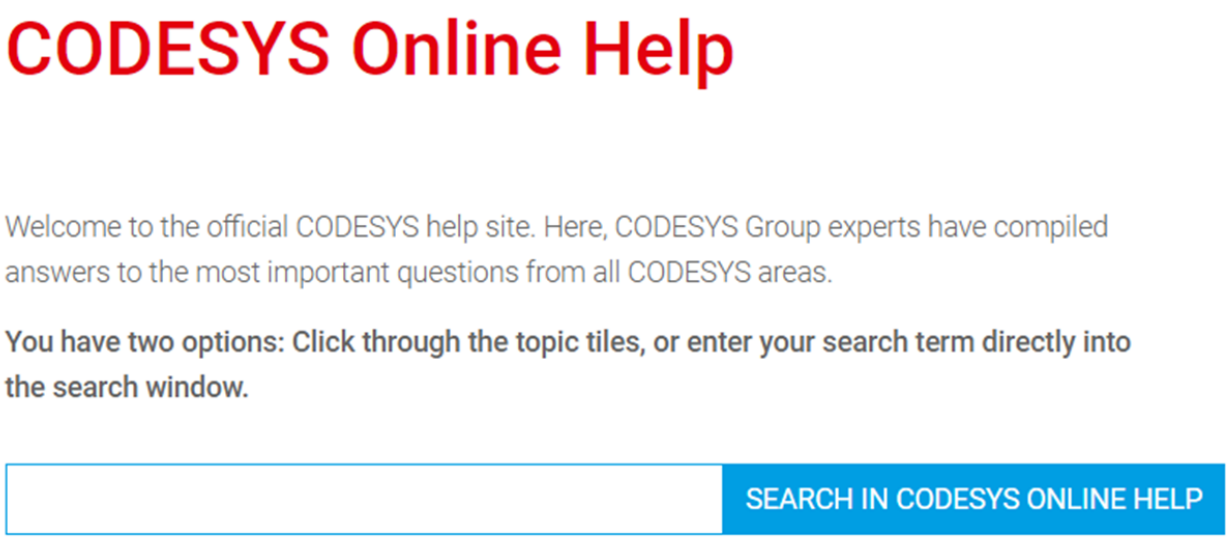
Среда программирования CODESYS является развитым, но достаточно сложным продуктом. В состав данного программного входит большое количество различных библиотек, каждая из которых содержит определённый набор команд. Существует несколько сообществ, разрабатывающих библиотеки независимо друг от друга. Кроме того, каждый производитель контроллеров разрабатывает и свой набор команд, работающих только с контроллерами данного производителя.

Контроллеры серии ME200 могут работать с библиотеками основного разработчика компании 3S, сообществ CAA и OSCAT, а также с специализированной библиотекой производителя контроллеров. Команды из библиотеки производителя контроллеров начинаются в с префикса LS_ и устанавливаются в составе пэкиджа.

Для использования конкретной команды необходимо в состав включить проекта включить соответствующую библиотеку. Найти название библиотеки по имени команды можно в справке среды программирования (Help) или на сайте:

https://www.helpme-codesys.com/

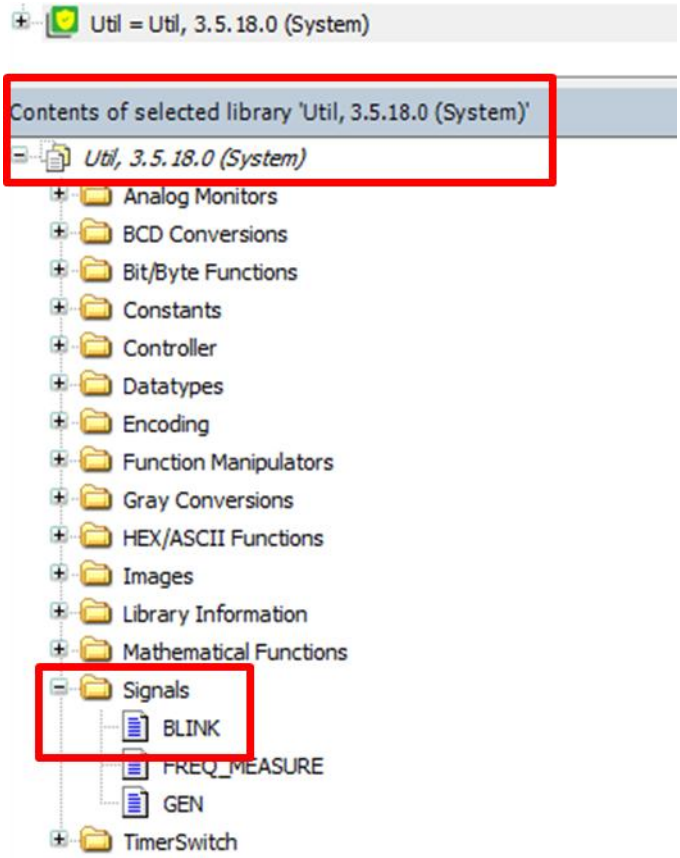
Например, Вы хотите использовать команду BLINK (фликер). Изначально данная команда не находится при попытке её использовать, так как библиотека, в состав которой она входит, не подключена к проекту. Вы заходите на сайт https://www.helpme-codesys.com/, открывается страничка, на которой есть поле поиска, куда необходимо ввести название команды и нажать Enter.



В ответ будет показано описание команды и сверху будет видно название библиотеки:



Далее через Мастер установки библиотек подключите данную библиотеку (Signals в нашем примере) к своему проекту. Часто библиотека может быть в составе более общей библиотеки. В нашем примере библиотеке **Signals** входит в состав библиотеки **Utils.** В итоге библиотека и команда должны появиться в списке подключенных библиотек:



Для удобства работы с контроллерами серии ME200 далее приводится список наиболее употребительных команд.

Категория инструкции	Название инструкции	FB/FC	Описание
Сравнение	GT	FC	больше, чем
	LT	FC	меньше, чем
	GE	FC	больше или равно
	LE	FC	меньше или равно
	EQ	FC	равно
	NE	FC	не равно
Выбор	SEL	FC	выбор из 2-х значений по логическому состоянию
	MUX	FC	выбор по номеру из набора значений
	MAX	FC	выбор большего из 2-х чисел
	MIN	FC	выбор меньшего из 2-х чисел
	LIMIT	FC	Ограничение по верхнему и нижнему значению
Счётчики общего назначения	CTD	FB	Счёт вверх
	CTU	FB	Счёт вниз
	CTUD	FB	Счёт вверх-вниз
Таймеры	TP	FB	Таймер с импульсным запуском
	TON	FB	Таймер с задержкой на включение
	TOF	FB	Таймер с задержкой на выключение
	RTC	FB	Часы реального времени
Логические операции	AND	FC	побитовое И
	OR	FC	побитовое ИЛИ
	NOT	FC	Логическое НЕ
	XOR	FC	побитовое ЛИБО
	SR	FB	Бистабильное реле с приоритетом пуска
	RS	FB	Бистабильное реле с приоритетом стопа
	R_TRIG	FC	Обнаружение нарастающего фронта
	F_TRIG	FC	Обнаружение спадающего фронта
	SHR	FC	Побитовый сдвиг

Сдвиг данных			вправо
	SHL	FC	Побитовый сдвиг влево
	ROR	FC	Побитовое вращение вправо
	ROL	FC	Побитовое вращение влево

Категория инструкций	Название инструкции	FB/FC	Описание
Преобразование типов данных	BOOL_TO_<TYPE>	FC	BOOL в другой тип
	BYTE_TO_<TYPE>	FC	BYTE в другой тип
	WORD_TO_<TYPE>	FC	WORD в другой тип
	DWORD_TO_<TYPE>	FC	DWORD в другой тип
	INT_TO_<TYPE>	FC	INT в другой тип
	SINT_TO_<TYPE>	FC	SINT в другой тип
	DINT_TO_<TYPE>	FC	DINT в другой тип
	UDINT_TO_<TYPE>	FC	UDINT в другой тип
	REAL_TO_<TYPE>	FC	REAL в другой тип
	STRING_TO_<TYPE>	FC	STRING в другой тип
	TIME_TO_<TYPE>	FC	TIME в другой тип
	TOD_TO_<TYPE>	FC	TOD в другой тип
	DATE_TO_<TYPE>	FC	DATEв другой тип
	DT_TO_<TYPE>	FC	DT в другой тип
Инструкции конвертации данных	MOVE	FC	Присвоение значения
	HEXinASCII_TO_BYTE	FC	Шестнадцатеричное значение, записанное ASCII кодами, преобразует в двоичное число размером байт
	BYTE_TO_HEXinASCII	FC	Преобразует двоичное число размером байт в шестнадцатеричное число, записанное ASCII кодами
	WORD_AS_STRING	FC	ASCII коды в string
	BYTE_TO_HEXSTRING	FC	Байт в шестнадцатеричный стринг

	WORD_TO_HEXSTRING	FC	Слово в шестнадцатеричный стринг
	DWORD_TO_HEXSTRING	FC	Двойное слово в шестнадцатеричный стринг

Категория инструкций	Название инструкции	FB/FC	Описание
Математические инструкции	ADD	FC	Сложение
	SUB	FC	Вычитание
	MUL	FC	Умножение
	DIV	FC	Деление
	MOD	FC	Остаток от деления
	ABS	FC	Абсолютная величина
	SQRT	FC	Корень квадратный
	LN	FC	Натуральный логарифм
	LOG	FC	Десятичный логарифм
	EXP	FC	Возведение в степень числа e (e в степени x)
	EXPT	FC	Возведение в степень
	SIN	FC	Синус угла в радианах
	COS	FC	Косинус угла в радианах
	TAN	FC	Тангес угла в радианах
	ASIN	FC	Арксинус угла в радианах
	ACOS	FC	Арккосинус угла в радианах
	ATAN	FC	Арктангенс угла в радианах
	XSIZEOF	FC	Определяет необходимое количество байтов для хранения данных

Категория инструкций	Название инструкции	FB/FC	Описание
Строковые инструкции	LEN	FC	Количество символов в строке
	LEFT	FC	Возвращает заданное количество символов при отсчёте слева
	RIGHT	FC	Возвращает заданное количество символов при отсчёте справа
	MID	FC	Возвращает заданное количество символов с заданной позиции в строке
	CONCAT	FC	Конкатенация (соединение) двух строк
	INSERT	FC	Вставка строки в заданную позицию другой строки
	DELETE	FC	Удаление заданного количества символов с заданной позиции в строке
	FIND	FC	Определение наличия одной строки в другой
	REPLACE	FC	Заменяет заданное количество символов с указанного места в строке

Категория инструкций	Название инструкции	FB/FC	Описание
Операции с файлами CAA File Library	FILE.Open	FB	Открыть файл
	FILE.Close	FB	Закрыть файл
	FILE.Write	FB	Записать файл
	FILE.Read	FB	Прочитать файл
	FILE.Delete	FB	Удалить файл
	FILE.Copy	FB	Копировать файл
	FILE.Rename	FB	Переименование файла
	FILE.SetPos	FB	Установить местоположение чтения и записи файла
	FILE.GetPos	FB	Получить местоположение чтения и записи файла
	FILE.GetSize	FB	Получить размер файла

Категория инструкций	Название инструкции	FB/FC	Описание
Регуляторы	PD	FB	ПД регулятор
	PID	FB	ПИД регулятор
	PID_FIXCYCLE	FB	ПИД регулятор с установкой цикла опроса
Преобразование BCD	BCD_TO_INT	FC	Преобразование числа в формате BCD в тип INT
	INT_TO_BCD	FC	Преобразование числа типа INT в формат BCD
	BCD_TO_BYTE	FC	Преобразование числа в формате BCD в тип BYTE
	BYTE_TO_BCD	FC	Преобразование числа типа BYTE в формат BCD
	BCD_TO_WORD	FC	Преобразование числа в формате BCD в тип WORD
	WORD_TO_BCD	FC	Преобразование числа тип WORD в формат BCD
	BCD_TO_DWORD	FC	Преобразование числа в формате BCD в тип DWORD
	DWORD_TO_BCD	FC	Преобразование числа тип DWORD в формат BCD
Программные фликеры	BLINK	FB	Программный генератор импульсов
	GEN	FB	Программный генератор периодических сигналов
	FREQ_MEASURE	FB	Измерение частоты входных импульсов

Категория инструкций	Название инструкции	FB/FC	Описание
Состояние оси	MC_Power	FB	Включение оси
	MC_Reset	FB	Сброс оси
	MC_ReadStatus	FB	Чтение статуса оси
	MC_ReadAxisError	FB	Чтение кода ошибки оси
	MC_ReadParameter	FB	Чтение параметров
	MC_ReadBoolParameter	FB	Чтение логических параметров
	MC_WriteParameter	FB	Запись параметров
	MC_WriteBoolParameter	FB	Запись логических параметров
	MC_ReadActualPosition	FB	Чтение положения оси
	MC_ReadActualVelocity	FB	Чтение скорости оси
	MC_ReadActualTorque	FB	Чтение момента на оси
	MC_SetPosition	FB	Задание позиции

	SMC_ReadSetPosition	FB	Чтение заданной позиции
	SMC_ReadFBError	FB	Чтение исторической информации об ошибках
	SMC_ClearFBError	FB	Очистить исторические сообщения об ошибках
	SMC_ErrorString	FB	Текст ошибки

Категория инструкций	Название инструкции	FB/FC	Описание
Управление одноосевым движением	MC_Home	FB	Возврат в ноль
	MC_MoveAbsolute	FB	Абсолютное позиционирование
	MC_MoveRelative	FB	Относительное позиционирование
	MC_MoveVelocity	FB	Движение с фиксированной скоростью
	MC_Stop	FB	Стоп движения оси
	MC_Halt	FB	Пауза движения оси
	MC_Jog	FB	Джог-движение
	MC_MoveAdditive	FB	Аддитивное позиционирование
	MC_MoveSuperImpose	FB	Наложенное движение
	MC_PositionProfile	FB	Плановое движение
	MC_VelocityProfile	FB	Плановое движение в режиме скорости
	MC_AccelerationProfile	FB	Плановое ускорение
	SMC_Homing	FB	Возврат в нулевую позицию
	SMC_Inch	FB	Пошаговое движение оси
Управление групповым движением осей	MC_GearIn	FB	Электронный редуктор
	MC_GearInPos	FB	Электронный редуктор с учётом соотношения позиций осей
	MC_GearOut	FB	Расцепление осей (прекращение MC_GearIn)
	MC_CamTableSelect	FB	Выбор таблицы E-CAM
	MC_CamIn	FB	Запуск конкретного E-CAM
	MC_CamOut	FB	Отключение конкретного E-CAM
	SMC_GetTappetValue	FB	Отображение статуса текущего сегмента E-CAM
	LS_2AxisLine	FB	Двухосевая линейная интерполяция

Управление интерполированным движением осей LS_IpoLib Library	LS_3AxisLine	FB	Трёхосевая линейная интерполяция
	LS_4AxisLine	FB	Четырёхосевая линейная интерполяция
	LS_5AxisLine	FB	Пятиосевая линейная интерполяция
	LS_6AxisLine	FB	Шестиосевая линейная интерполяция
	LS_2AxisLineA_Ratio	FB	Двухосевая линейная интерполяция с регулируемой скоростью
	LS_LineFollow	FB	Отслеживание
	LS_2AxisCircle	FB	Двухосевая круговая интерполяция
	LS_3AxisCircle	FB	Трёхосевая круговая интерполяция
	LS_2AxisEllipses	FB	Двухосевая эллиптическая интерполяция
	LS_2AxisCircle_Helical	FB	Спиральная интерполяция
	LS_3AxisMoveSequence	FB	Трехосное непрерывное интерполяционное движение
	LS_4AxisMoveSequence	FB	Четырехосное непрерывное интерполяционное движение
	LS_6AxisMoveSequence	FB	Шестиосное непрерывное интерполяционное движение

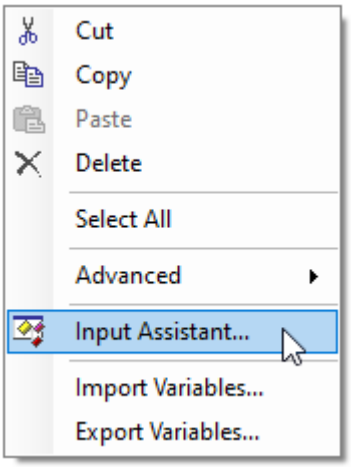
Категория инструкций	Название инструкции	FB/FC	Описание
TCP сокет	TCP_Client	FB	Открыть TCP сокет в режиме клиента
	TCP_Write	FB	Отправка данных через TCP сокет
	TCP_Read	FB	Приём данных через TCP сокет
	TCP_Connection	FB	Инкапсуляция TCP соединения между клиентом и сервером
	TCP_Server	FB	Открыть TCP сокет в режиме сервера
UDP сокет	UDP_Peer	FB	Открыть UDP сокет
	UDP_Receive	FB	Приём данных через UDP сокет
	UDP_Send	FB	Отправка данных через UDP сокет
	IoDrvEtherNetIP_Diag	FB	Расширение для IoDrvEtherNetIP

Инструкции Ethernet/IP	RemoteAdapter	FB	Расширения для LAT.Element
	Generic_Service	FB	Предоставляет generic service для EtherNet/IP Adapter
	Get_Attributes_All	FB	Получить все свойства экземпляра объекта
	Get_Attribute_Single	FB	Получить один атрибут экземпляра объекта
	Set_Attributes_All	FB	Установить все свойства экземпляра объекта
	Set_Attribute_Single	FB	Установите одно свойство экземпляра объекта

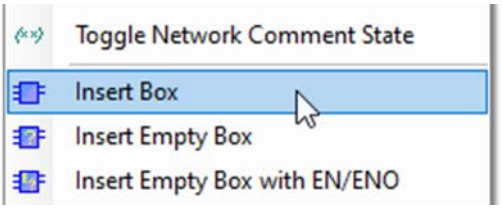
Категория инструкций	Название инструкции	FB/FC	Описание
Импульсное управление осью LS_MTHSIO	MC_Home_P	FB	Импульсная ось возвращается в исходное положение
	MC_MoveFeed	FB	
	LS_Counter	FB	Инструкция высокоскоростного счетчика
	LS_PresetValue	FB	Команда предустановленного значения счетчика
	LS_Compare	FB	Включение выхода по сравнению
	LS_EnableInterrupt	FB	Разрешение прерывания по входу
	LS_PresetValue	FB	Предустановленное значение счётчика
	LS_PWM	FB	Выдача ШИМ импульсов
	LS_TouchProbe	FB	Захват текущей позиции

Категория инструкций	Название инструкции	FB/FC	Описание
Системные команды LS_SysLib2	SysLS_GetIPAdr	FB	Прочитать IP адрес контроллера
	SysLS_ChangeIPAdr	FB	Установить IP адрес контроллера
	GetBatState	FC	Прочитать уровень заряда батареи
	ColdResetApp	FC	«Холодная» перезагрузка контроллера
	WarmResetApp	FC	«Тёплая» перезагрузка контроллера

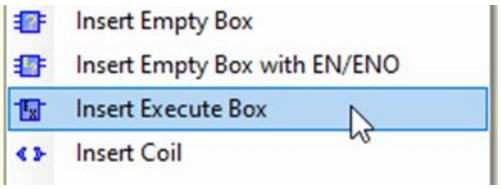
Для добавления в проект команды типа FB или FC на языке ST необходимо просто вызвать нажатием правой кнопки мышки и выбрать пункт **Input Assistant**:

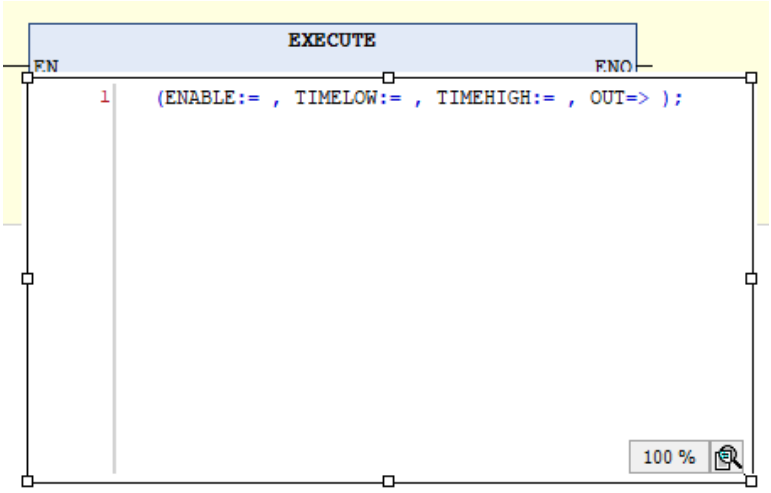


На языке LD команды типа FB вводятся через меню



а команды типа FC вводятся через

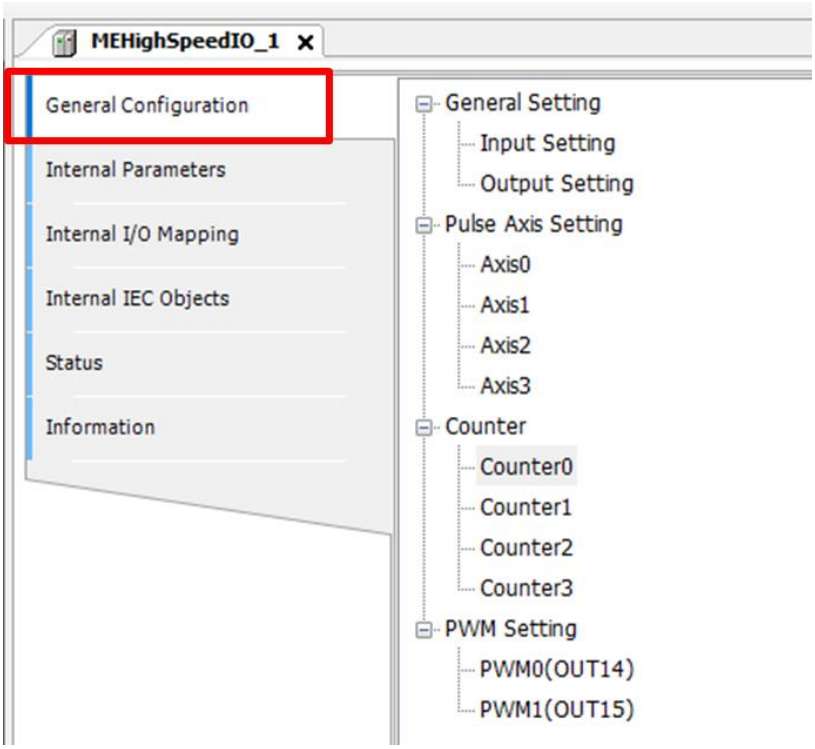




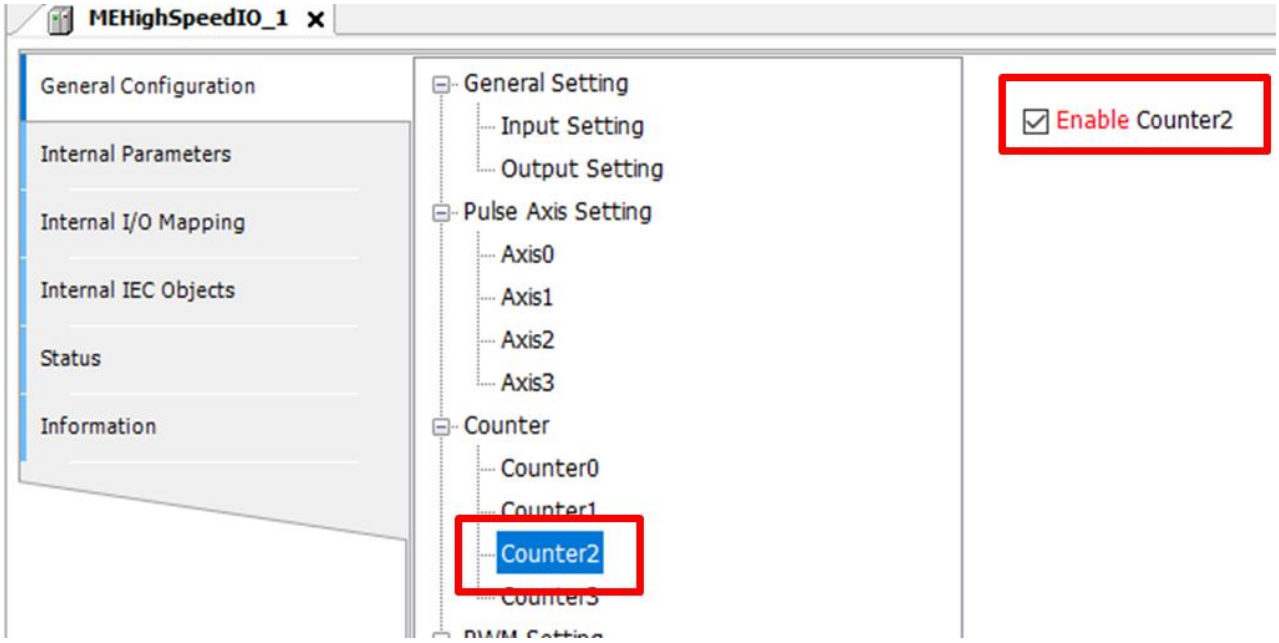
Работа с высокоскоростными счётчиками

Контроллер имеет на борту 4 AB-канала высокоскоростного счёта (входы 00-07). Далее описывается последовательность настройки высокоскоростных счётчиков.
(в проект должен быть добавлен заранее объект **MEHighSpeedIO**, см. данное Руководство).

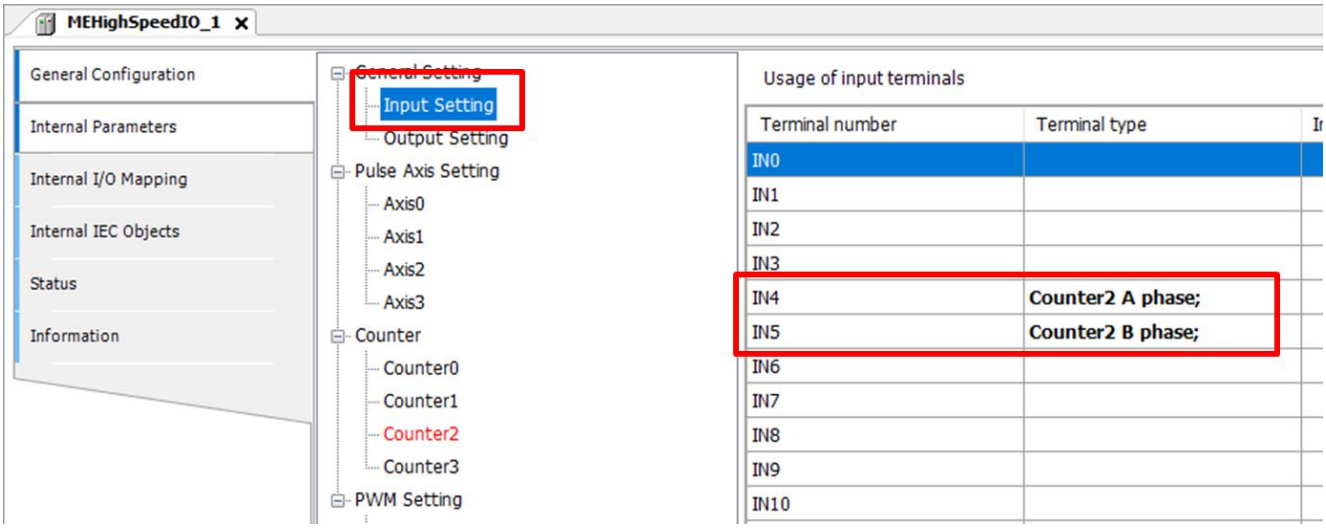
Задание параметров работы высокоскоростных счётчиков
Щёлкните дважды на пункте“**MEHighSpeedIO**” в древе проекта и в открывшейся вкладке выберите раздел “**General Configuration**”:



Выберите для примера пункт **Counter2** и поставьте флажок:



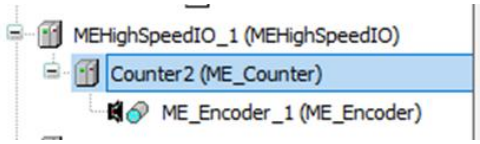
В пункте Input Setting появятся обозначения счётчика на входах 4 и 5.



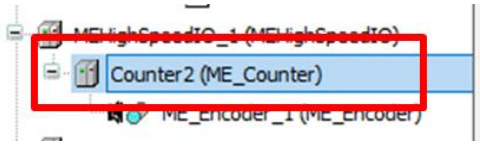
Здесь же можно выставить входной фильтр:

IN4	**Counter2 A phase;**		2	Unit 100ns, high speed input (range 1-65535)
IN5	**Counter2 B phase;**		2	Unit 100ns, high speed input (range 1-65535)

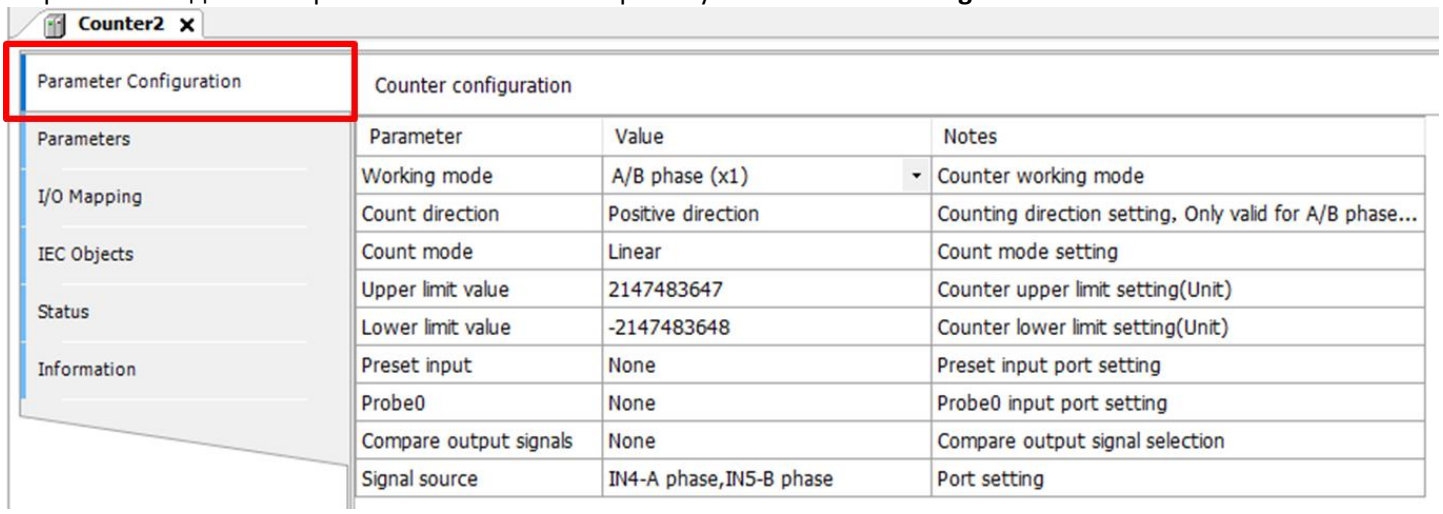
Также, в древе проекта под адаптером **MEHighSpeedIO** появится счётчик и привязанная к нему энкодерная ось:



Щёлкните на пункте **Counter2(ME_Counter)** два раза левой кнопкой мышки:



откроется вкладка с настройками счётчика. Выберите пункт **Parameter Configuration**:

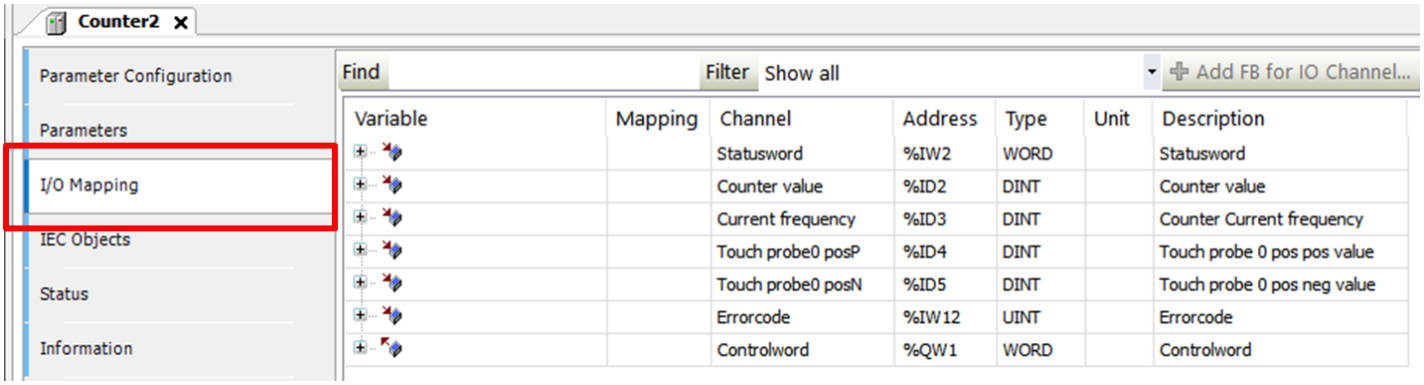


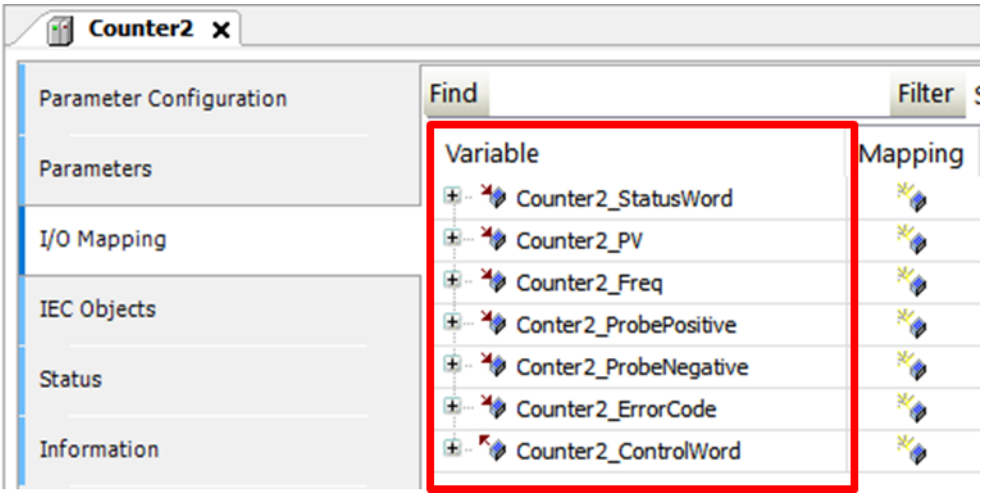
В данной вкладке можно сделать настройки высокоскоростного счётчика:

Working mode – Рабочий режим (AB, импульс/направление и т.д.)
Count direction – Positive – счёт вверх, Negative – счёт вниз
Count mode – Linear – линейный счёт (до предела), Rotary – круговой счёт
Upper Limit Value – верхний предел счёта
Lower Limit Value – нижний предел счёта
Preset input – вход для аппаратного сброса счётчика (вход для инструкции LS_PresetValue)
Probe – вход для инструкции LS_TouchProbe (привязка по названию энкодерной оси)
Compare Output Signals – выход для инструкции LS_Compare (привязка по названию энкодерной оси)
Signal Source – назначение входов на ЦПУ для счётчика (можно изменить установленные по умолчанию)

Пункт **Parameters** служит для информативных целей в нём ничего менять не надо. Он сам модифицируется в зависимости от настроек в пункте **Parameter Configuration**.

В пункте **I/O Mapping** в колонке **Variable** можно задать названия переменным. Однако, данные переменные являются внутренними для команды **LS_Counter**, поэтому их можно использовать только после запуска данной команды в программе:



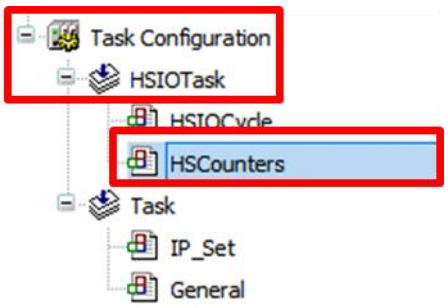


Для обращения к счётчикам в программе используйте команды:

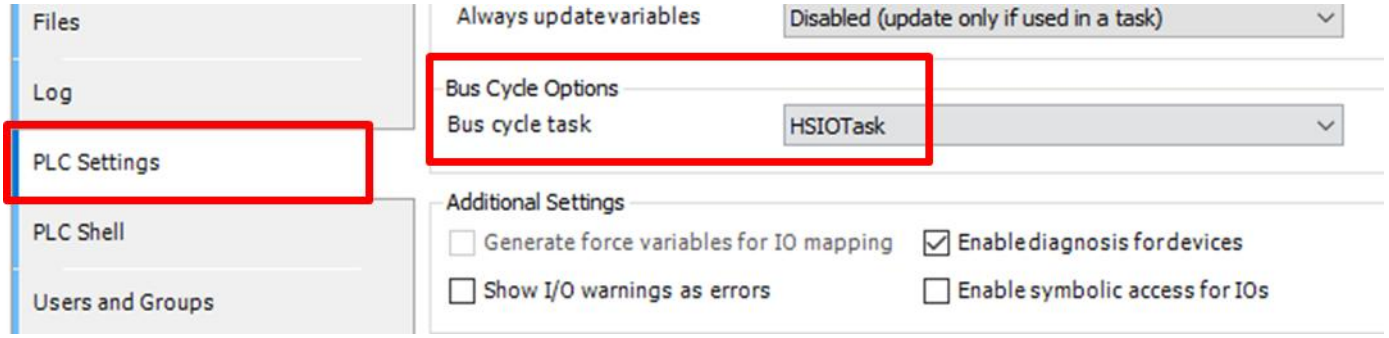
LS_Counter - для получения текущего значения счётчика

LS_PresetValue - для обнуления счётчика

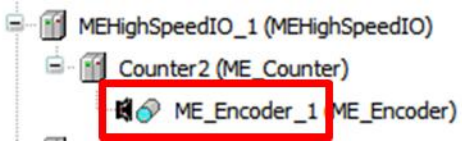
Программный модуль (POU, в нашем примере HSCounters), где будет использоваться команды работы с высокоскоростным счётчиком, должен быть привязан в Менеджере Задач (**Task Configuration**) к основной задаче (приоритет 0) **HSIOTask**:

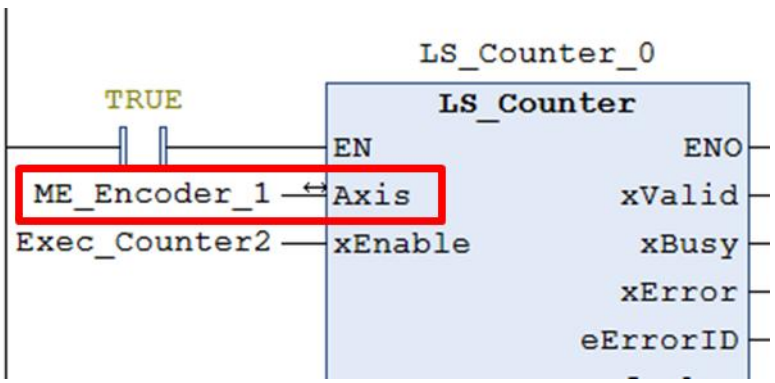


Также, во вкладке Device в пункте PLC Settings нужно выбрать в качестве такта шины (Bus cycle task) привязать основную задачу (приоритет 0) **HSIOTask**:

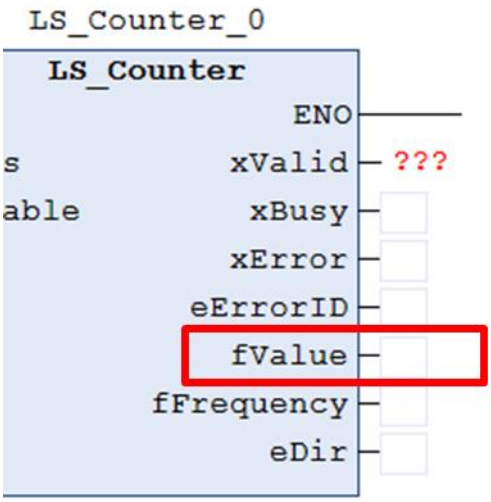


Для обращения к счётчику в командах необходимо использовать имя энкодерной оси в древе проекта (в нашем проекте ME_Encoder_1):

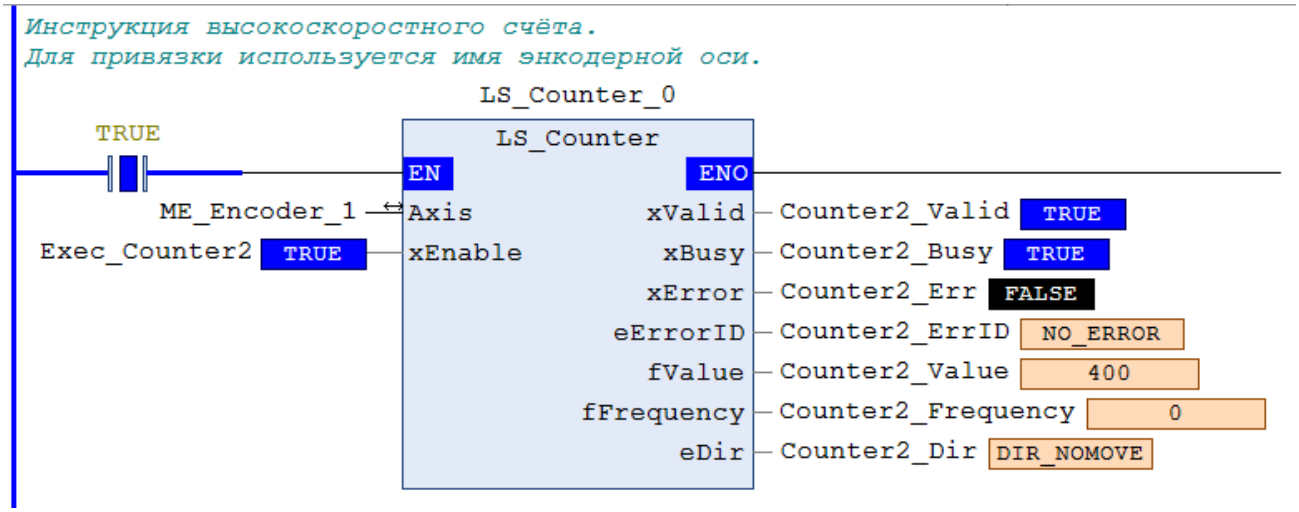




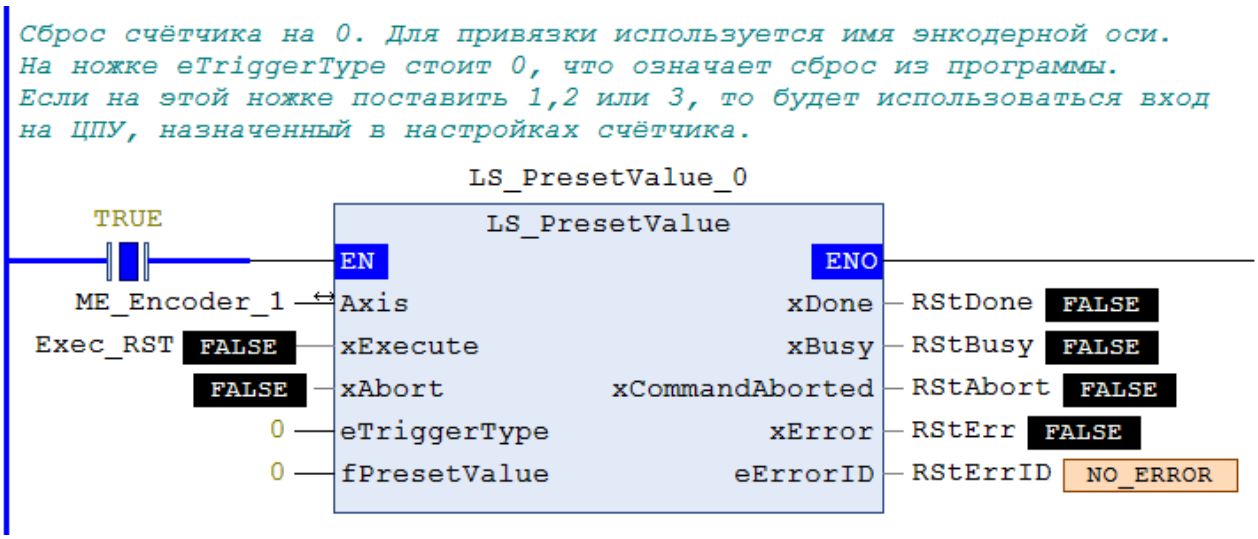
Для получения значения счётчика нужно использовать ножку инструкции **fValue**:



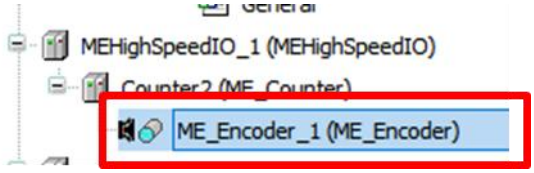
Пример инструкции высокоскоростного счёта:



Пример инструкции сброса счётчика на ноль из программы:



Как было показано выше при назначении высокоскоростного счётчика в древе проекта к нему автоматически привязывается энкодерная ось:

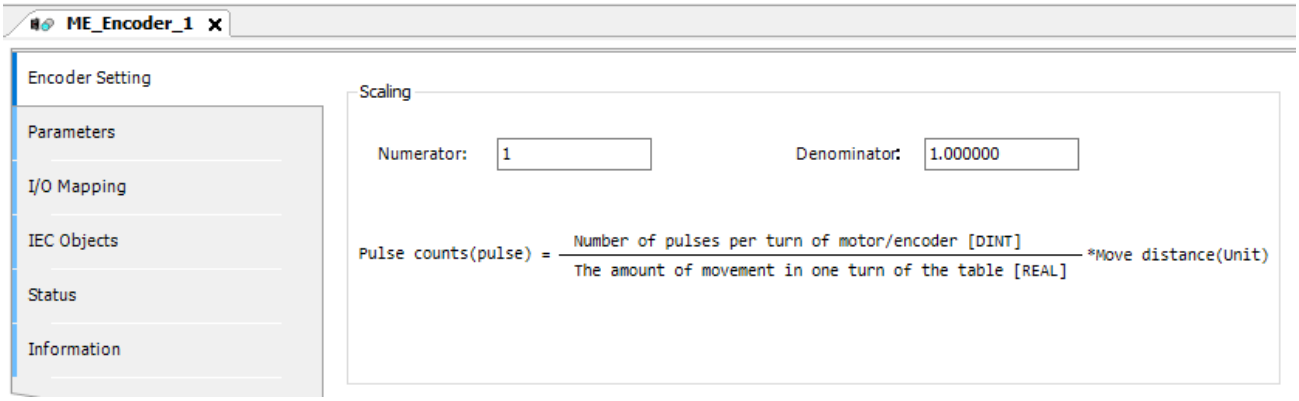


Энкодерная ось может использоваться в качестве Мастер оси в стандартных инструкциях управления движением типа MC_Gear и т.п.

Текущее положение можно получить из параметра **fActPosition** оси структуры оси:

ME_Encoder_1.fActPosition

Настройки коэффициента редукции осуществляется в пункте **Encoder Setting**:



Если коэффициенты **Numerator** и **Denominator** равны 1, то позиция оси и значение счётчика будут прямо соответствовать количеству физически полученных импульсов на счётном входе:

ME_Encoder_1.fActPosition = HSCounters.LS_Counter_0.fValue = количество полученных импульсов

Например, если подключить стандартный штурвальный энкодер на 100 импульсов/оборот в режиме 4xA/B, то на 1 оборот будет принято 400 импульсов, значение счётчика и позиция оси также будет 400:

ME_Encoder_1.fActPosition	Controller_1.Application	LREAL	400
HSCounters.LS_Counter_0.fValue	Controller_1.Application	LREAL	400

Если внести редукцию, то принятое количество импульсов и позиция оси/значение счётчика будут отличаться. Коэффициент может быть только повышающим. Формула сводится к следующему:

Позиция оси/значение счётчика = входные импульсы * Denominator/Nominator

ME_Encoder_1.fActPosition = HSCounters.LS_Counter_0.fValue = количество полученных импульсов * Denominator/Nominator

Внимание!

Denominator/Numerator >= 1 (больше или равно единицы)

Например, если сделать Denominator = 4 (Numerator=1), то позиция оси и значение счётчика будет в 4 раза больше, чем физически принятых импульсов на счётном входе.

Например, если подключить стандартный штурвальный энкодер на 100 импульсов/оборот в режиме 4xA/B, то на 1 оборот будет принято 400 импульсов, а значение счётчика и позиция оси будут 1600:

ME_Encoder_1.fActPosition	Controller_1.Application	LREAL	1600
HSCounters.LS_Counter_0.fValue	Controller_1.Application	LREAL	1600

Принятое количество импульсов можно пересчитать в линейное перемещение по следующей формуле:

Перемещение (user units) = количество принятых импульсов * линейное перемещение (user units) на 1 оборот энкодера /количество импульсов на 1 оборот энкодера

Предположим у нас 1000 импульсов на 1 оборот энкодера, механизм проходит 10 мм (user units) на 1 оборот энкодера. Контроллер принял 4000 импульсов. Следовательно пройдённое расстояние механизмом будет:

S (user units) = 4000 * 10 / 1000 = 40 мм

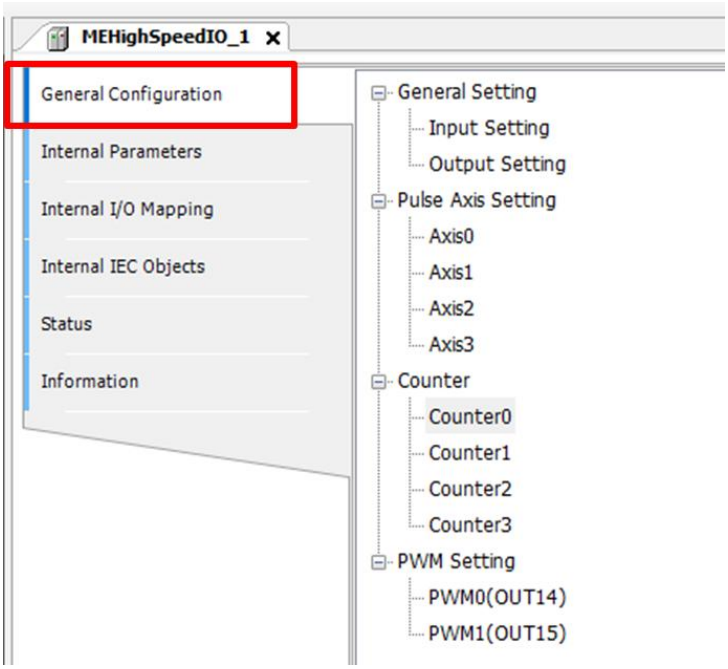
Импульсная ось движения

Контроллеры серии ME200 имеют 8 импульсных выходов 00-07 по 200 кГц, сгруппированных в 4 оси. Каждая ось соответственно занимает по 2 импульсных выхода: импульс/направление или AB режим. Остальные 8 выходов 08-15 являются обычными.

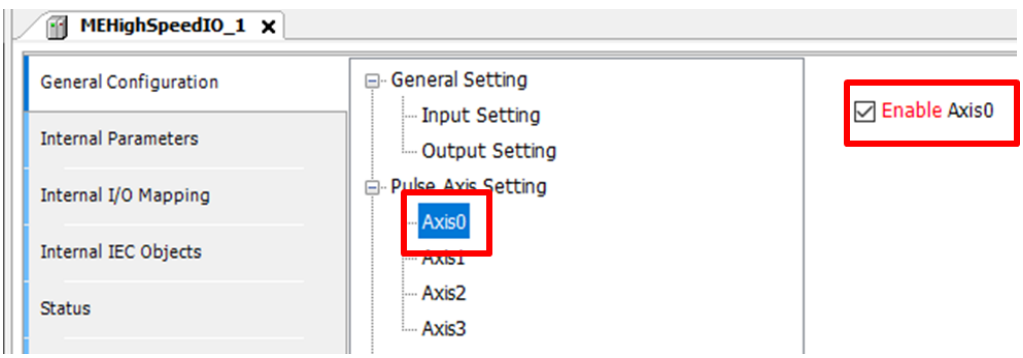
0	Импульсный выход	AXIS 0 PUL	8	Обычный выход
1	Импульсный выход	AXIS 0 DIR	9	Обычный выход
2	Импульсный выход	AXIS 1 PUL	10	Обычный выход
3	Импульсный выход	AXIS 1 DIR	11	Обычный выход
4	Импульсный выход	AXIS 2 PUL	12	Обычный выход
5	Импульсный выход	AXIS 2 DIR	13	Обычный выход
6	Импульсный выход	AXIS 3 PUL	14	Обычный выход/выход под ШИМ
7	Импульсный выход	AXIS 3 DIR	15	Обычный выход/выход под ШИМ

Для конфигурирования импульсной оси необходимо сначала добавить в проект адаптер входов-выходов **MEHighSpeedIO** (см. данное Руководство, раздел входов-выходов).

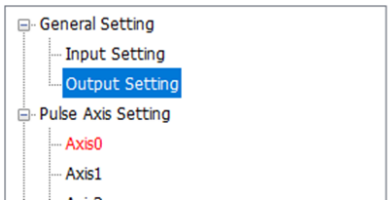
Щёлкните дважды на пункте“**MEHighSpeedIO**” в древе проекта и в открывшейся вкладке выберите раздел “**General Configuration**”:



Выберите для примера пункт **Axis0** и поставьте флажок **Enable**:



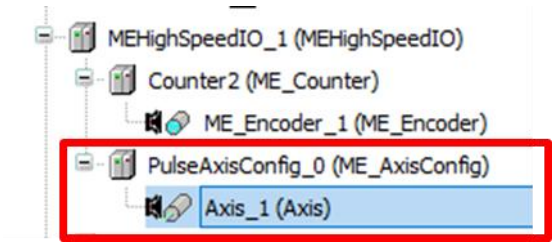
В пункте **Output Setting** появится назначение двух выходов для оси **Axis0**:



Usage of output terminals

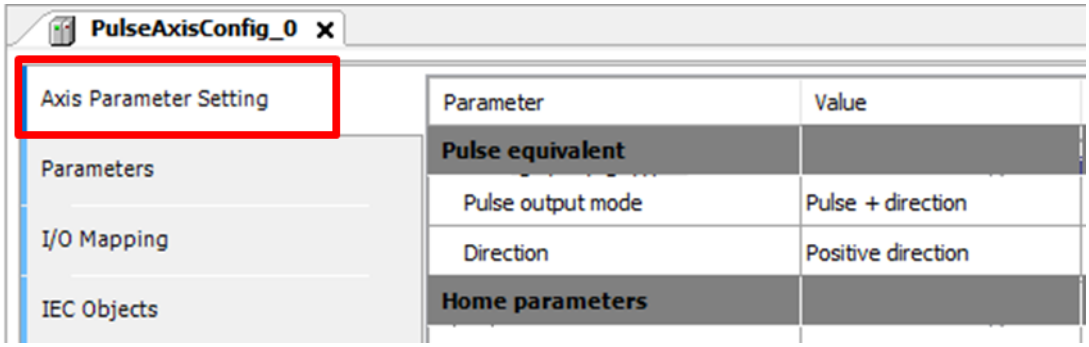
Terminal number	Terminal type	Notes
OUT0	**Axis0;**	High speed output 200KHz
OUT1	**Axis0;**	High speed output 200KHz
OUT2		General Output
OUT3		General Output

Также, в древе проекта появится адаптер импульсной оси и автоматически к нему будет притянута стандартная виртуальная ось:



Адаптер обеспечивает доступ к импульсным выходам на контроллере, а виртуальная ось обеспечивает стандартные настройки на пользовательские единицы длины, скорости и ускорения.

Щёлкните дважды левой кнопкой мышки на пункте **PulseAxisConfig_0(ME_AxisConfig),** откроется вкладка настройки режимов работы импульсной оси. Выберите пункт **Axis Parameter Setting**:



Параметры:

Pulse Output mode – Режим выдачи импульсов
Direction – Прямой и обратный вариант выдачи импульсов
Home Mode – Режим выхода в ноль
Home Signal – Сигнал датчика нуля
Home Signal Logic – НО/НЗ исходное состояние датчика нуля
Home Direction – Направление выхода в ноль
Home Position Offset – Смещение позиции нуля (user units)
Home Velocity – Скорость выхода в ноль (user units)
Creed Speed – «Ползучая» скорость подхода к нулю (user units)
Home Acceleration – Ускорение при выходе в ноль (user units)
Home Deceleration – Замедление при выходе в ноль (user units)
Home Latch Source – Источник положения оси при выходе в ноль (Command – задание)
Home Trigger Edge – Передний/Задний фронт сигнала датчика нуля

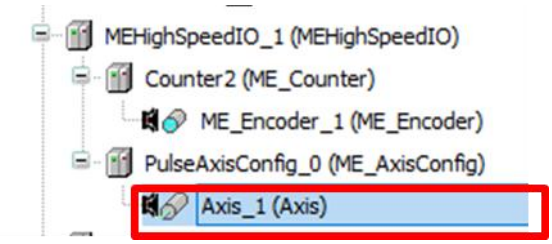
Stop Mode – Метод останова после выхода в ноль
Positive Limit Signal – датчик крайнего положения в положительном направлении движения
Positive Limit Signal Logic – НО/НЗ исходное состояние датчика крайнего положения ПНД
Negative Limit Signal – датчик крайнего положения в отрицательном направлении движения
Negative Limit Signal Logic – НО/НЗ исходное состояние датчика крайнего положения ОНД
Probe0 – Вход на ЦПУ для захвата позиции оси по внешнему сигналу. Работает с LS_TouchProbe FB

Настройки, связанные с выходом в ноль, используются инструкцией **MC_HOME_P**

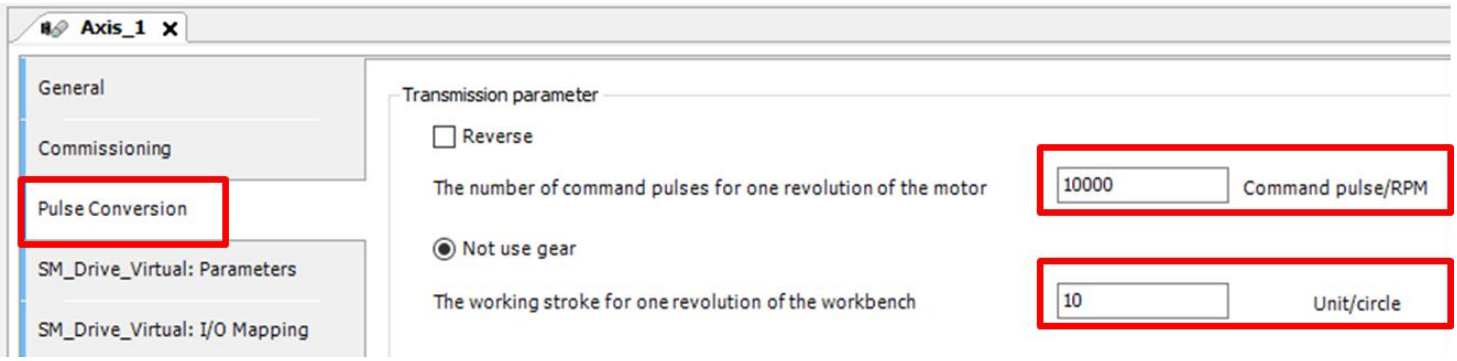
Остальные пункты вкладки **PulseAxisConfig** являются внутренними и заполняются автоматически по мере выбора параметров в пункте **Axis Parameter Setting**.

После настройки режимов работы оси необходимо задать пользовательские единицы расстояния, скорости и ускорения.

Для этого щёлкните дважды левой кнопкой мышки на пункте **Axis_1** в древе проекта:



Откроется вкладка настройки единиц пользователя, в которой необходимо заполнить сначала пункт Pulse Conversion, который устанавливает главное соотношение количества импульсов на 1 оборот мотора и сколько один оборот мотора составит в перемещении конечного механизма в единицах пользователя (**user units**):



На рисунке выше мы установили 10000 импульсов на 1 оборот вала мотора, что будет соответствовать 10 мм перемещения конечного механизма (в нашем примере). Это означает, что если в командах движения будет задана позиция 10, то контроллер выдаст 10000 импульсов и привод сделает ровно 1 оборот, а конечный механизм пройдёт 10 единиц пользователя.
Единицами пользователя могут выступать любые удобные единицы: миллиметры, микроны, дюймы и т.д.

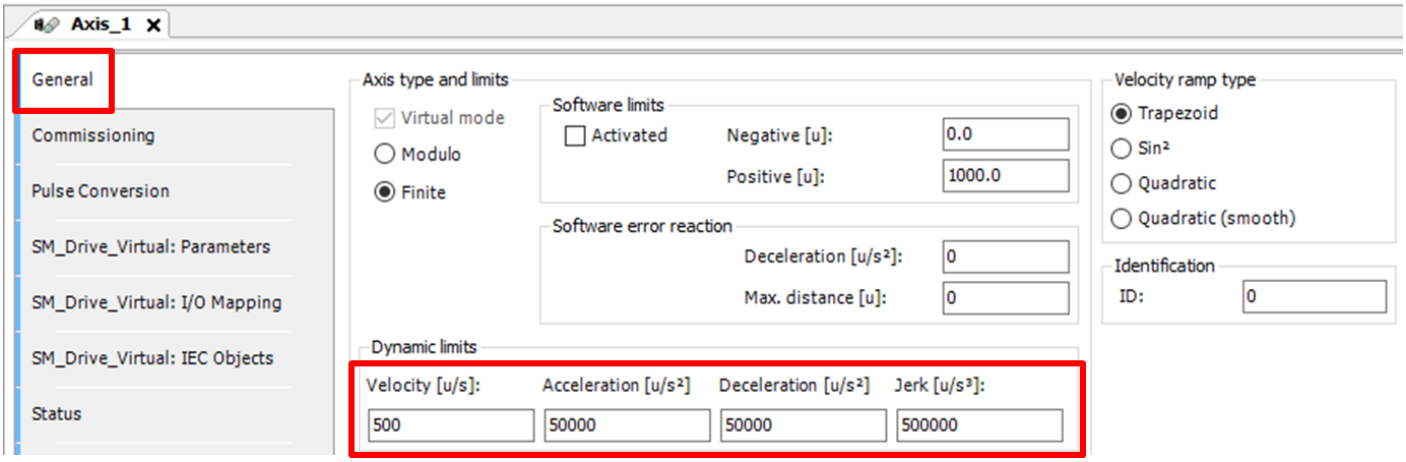
Примечание:
Так как у контроллера максимальная частота выхода 200 кГц, то для привода с номинальной максимальной скоростью вращения в 3000 об/мин максимально возможное число импульсов на оборот в настройках сервопривода будет 4000 импульсов на оборот. Получается это следующим образом:

3000 : 60 = 50 оборотов в секунду, 200000 : 50 = 4000

В нашем примере мы выбрали 10000 импульсов на оборот. Это означает, что максимально возможная скорость, которую можно будет задать приводу составит:

200000 : 10000 = 20 оборотов в секунду, или 20 * 60 = 1200 оборотов в минуту

Далее необходимо открыть пункт **General** и задать предельные значения скорости и ускорения/замедления в единицах пользователя:



В нашем примере, где мы выбрали перемещение на 1 оборот двигателя в 10 ед., предельная скорость в 3000 об/мин будет соответствовать:

10 * 50 = 500 ед. пользователя в секунду (u/s)

Расчёт предельных ускорения/замедления рассчитывается следующим образом:

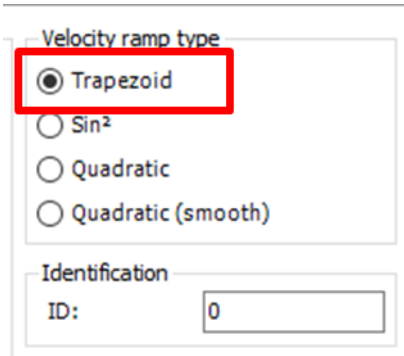
Сначала определяется предельное теоретическое ускорение за 1 мс. Для этого единицы скорости переводятся в мс, т.е. нужно умножить на 1000:

500 * 1000 = 500 000 ед. пользователя в секунду в квадрате (u/s2)

т.е. при этом ускорении привод разгонится до 3000 об/мин за 1 мс (теоретически)

Как правило, максимальное ускорение всегда ограничивают хотя бы 10 мс.
Т.е. надо разделить на 10 и получится значение 50 000, которое и стоит в настройках выше. Параметр Jerk как правило ставят в 10 раз больше, чем ускорение. В нашем примере это будет 500 000.

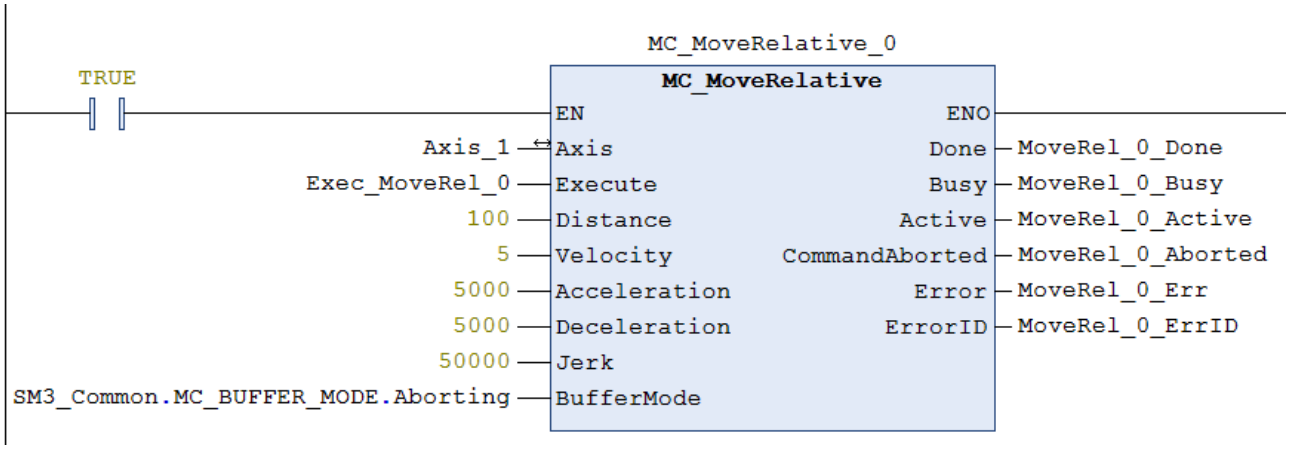
Тип рампы лучше выставлять для всех случаев **Trapezoid** (стандартная трапеция). Параметр Jerk обеспечивает S-образность кривой разгона.



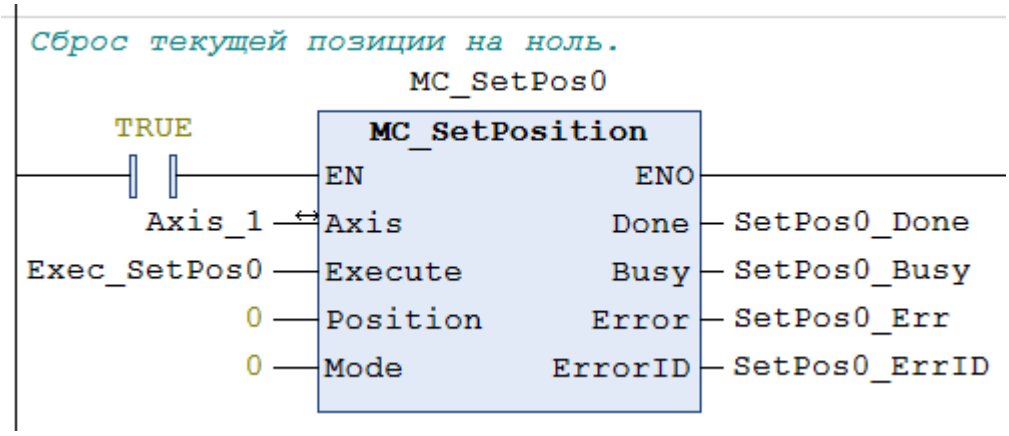
Импульсная ось также должна быть сначала запущена командой MC_Power, а далее используются в обычном порядке команды управления движением Soft Motion и LS.



Команда относительного перемещения:



Сброс текущей позиции оси на ноль осуществляется командой **MC_SetPosition** с нулевым значением позиции:

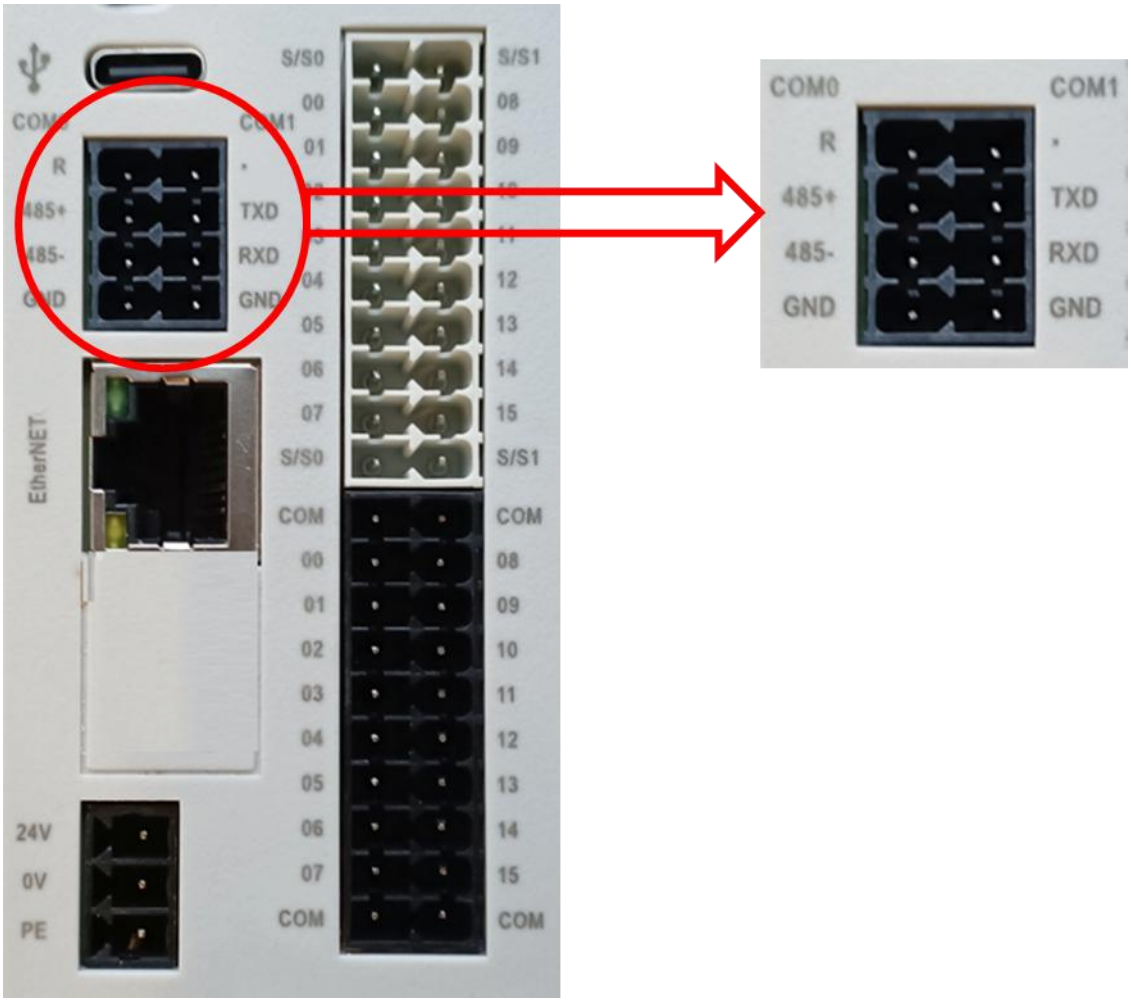


Последовательная связь по протоколу Modbus RTU Master

Контроллеры серии ME200 имеют на борту 1 порт RS232 и 1 порт RS485. Оба порта могут работать как в режиме Мастера, так и Ведомого. В данной главе рассмотрена работа в режиме Мастера по протоколу Modbus RTU.

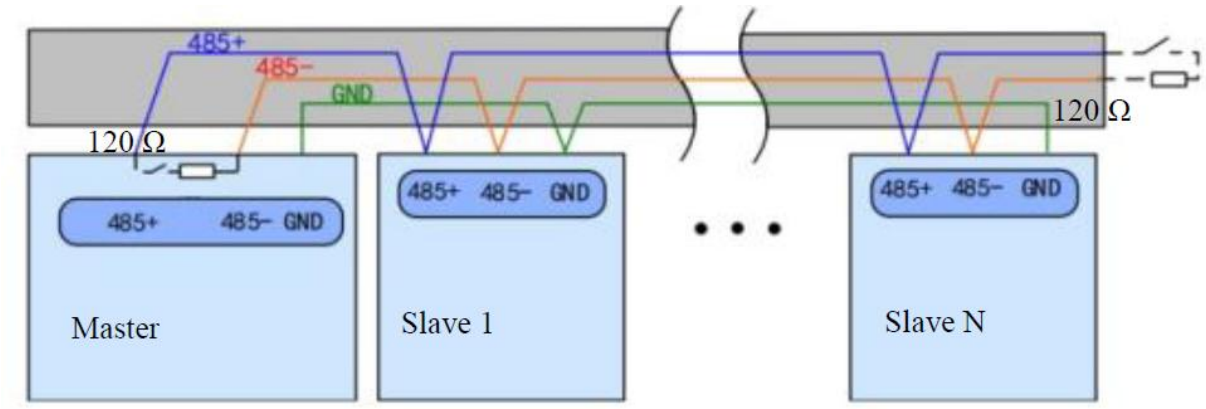
Физически порты имеют на контроллере следующее расположение:

COM0 -RS485, COM1 – RS232



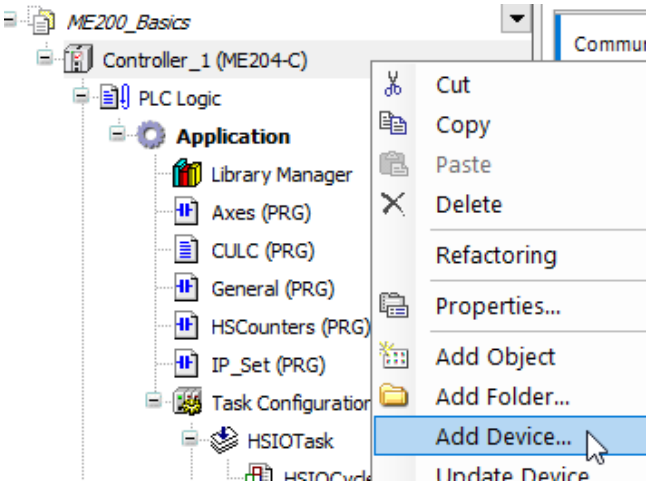
По RS232 возможно соединение только точка-точка. Максимальная длина кабеля на скорости 9600 максимум 15 метров. На скорости 115200 – 3 метра.

По RS485 возможно подключение до 31 Ведомого. Максимальная длина кабеля на скорости 9600 до 300 метров. Далее необходимо использовать репитер. Однако, такая длина кабеля достигается только при грамотной разводке кабеля связи и использовании согласующих резисторов на концах линии, а также балластного провода для выравнивания нулевого потенциала всех станций.

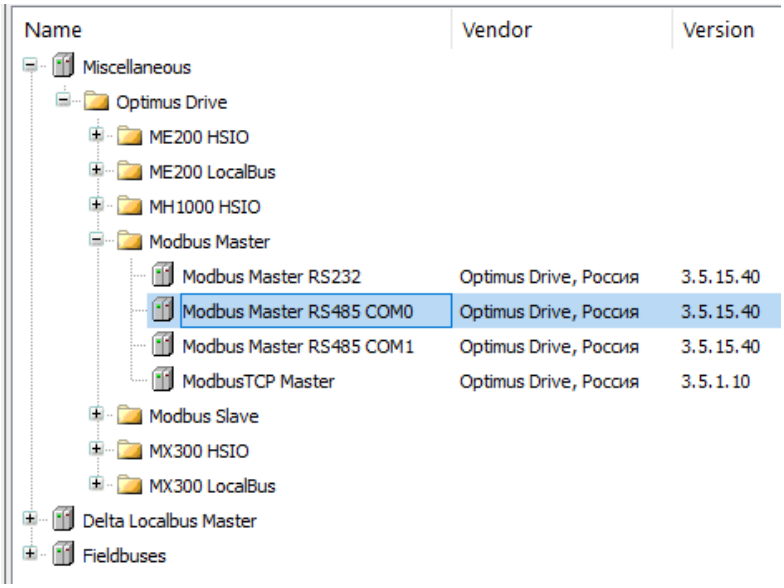


У контроллеров ME200 имеется встроенный резистор 120 Ом, который подключается путём установки перемычки между клеммами 485+ и R. Таким образом, мы соединяем клемму RS485+ через резистор 120 Ом на клемму RS485-.

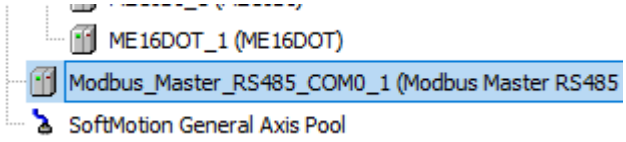
Для организации связи по Modbus RTU в проект необходимо добавить адаптер соответствующего порта. Щёлкните правой кнопкой мышки на вкладке Device и в открывшемся меню выберите пункт **Add Device**:



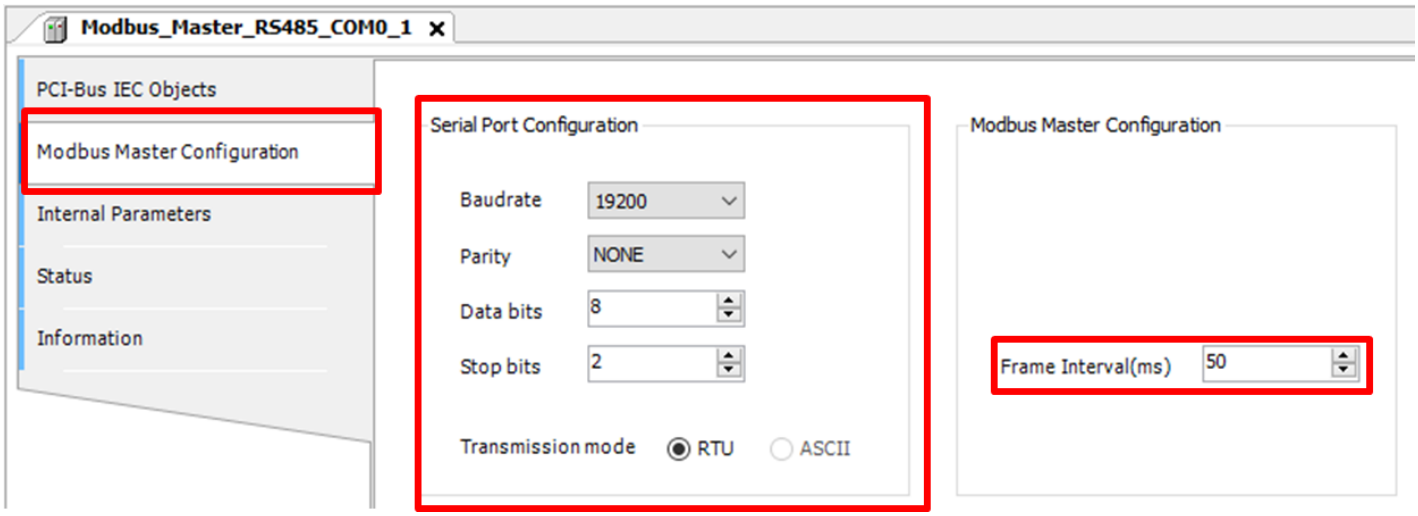
В открывшемся окне выберите папку **Miscellaneous – Optimus Drive – Modbus Master – Modbus Master RS485 COM0**:



В древе проекта появится пункт **Modbus_Master_RS485_COM0_1**:

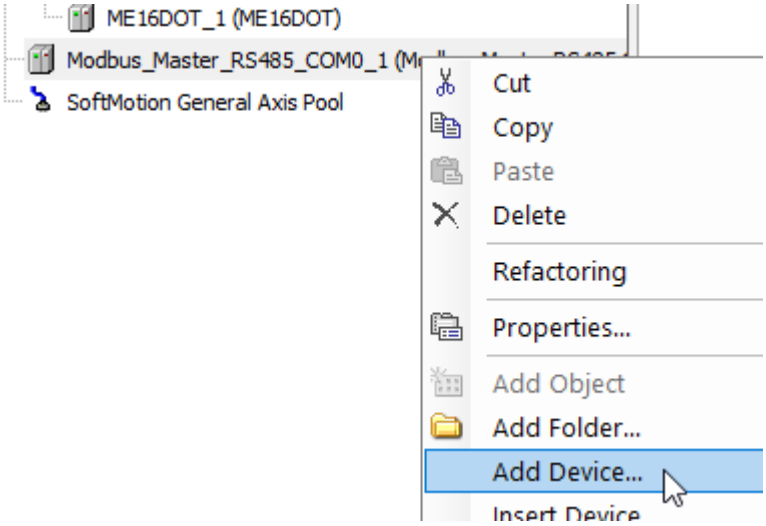


Щёлкните по этому пункту дважды левой кнопкой мышки и в открывшейся вкладке сделайте настройки связи в разделе **Modbus Master Configuration** для порта RS485_COM0:

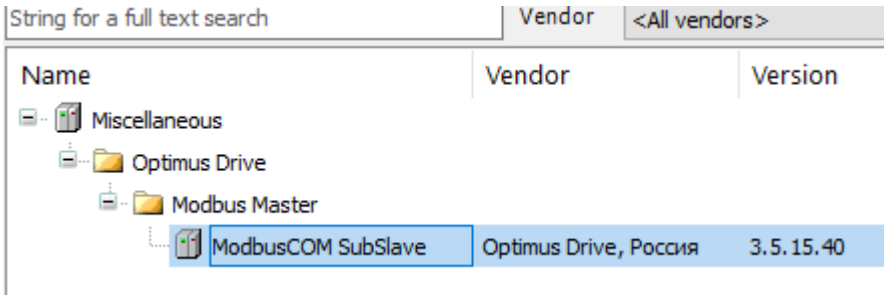


Поддерживается только режим Modbus RTU. В нашем примере выставлена скорость 19200, протокол 8, N, 2. Пункт справа **Frame Interval (ms)** определяет темп отправки пакетов. В нашем примере стоит 50 мс.

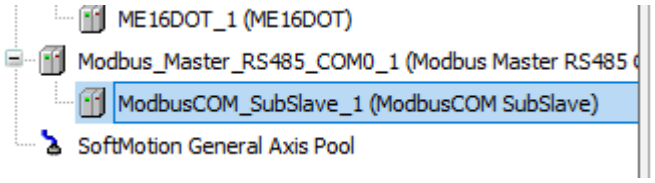
Далее щёлкните на пункте Modbus_Master_RS485_COM0_1 правой кнопкой мышки и в открывшемся окне выберите пункт **Add Device** и добавьте Ведомое устройство **Modbus RTU**:



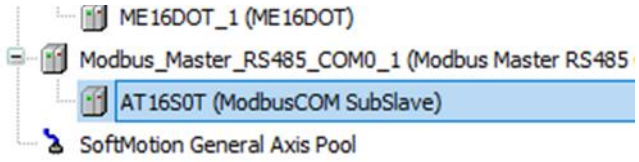
В открывшемся окне выберите пункт **ModbusCOM_SubSlave**:



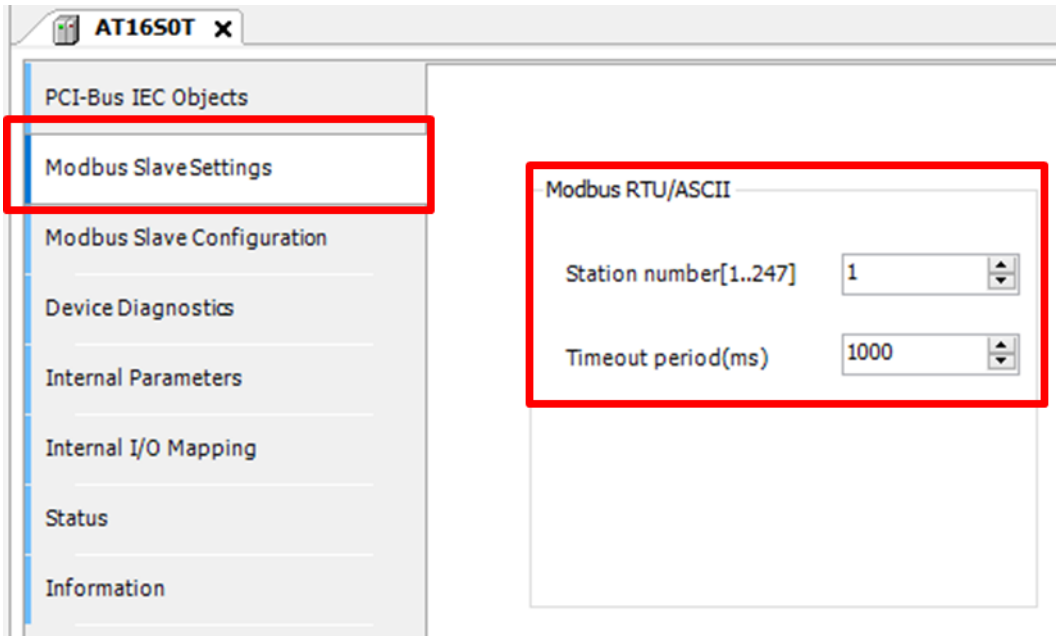
В древе проекта появится пункт **ModbusCOM_SubSlave_1**:



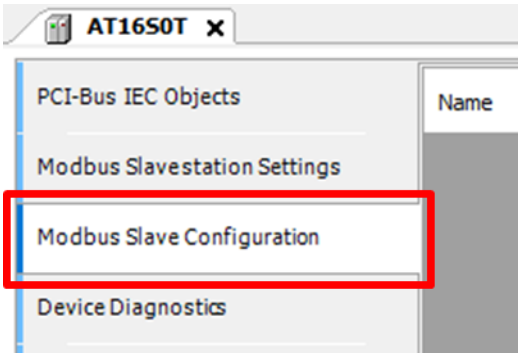
Название Ведомого можно поменять. Например заменить на контроллер AT16S0T:



Щёлкните дважды на Ведомом, откроется вкладка для создания запросов.
В пункте **Modbus Slave Station Settings** необходимо задать сетевой адрес Ведомого и таймаут связи:



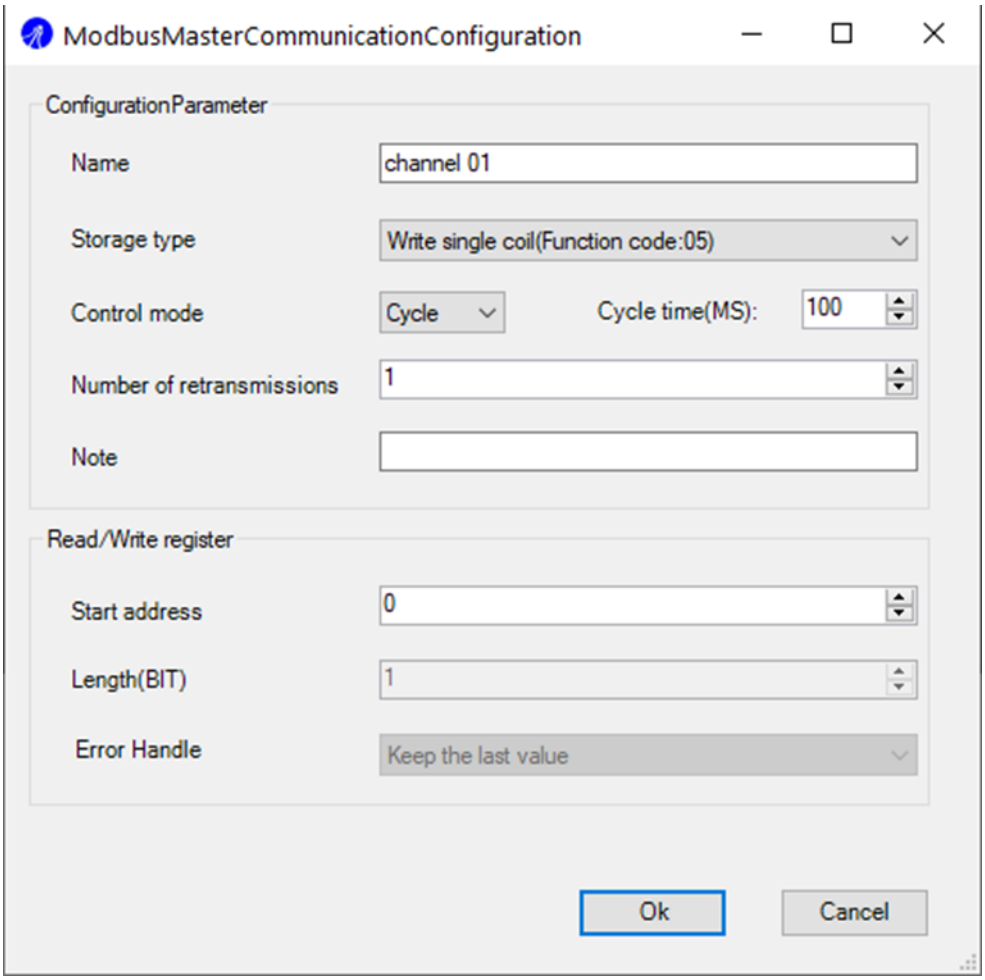
Далее в пункте **Modbus Slave Configuration** необходимо задать регистры Ведомого и их количество для чтения/записи Мастером. Всего может быть создано 31 соединение с Ведомым.



Для создания поля обмена данным с Ведомым необходимо нажать кнопку **Add**:



Появится поле обмена данными с Ведомым, в котором необходимо определить регистры Ведомого, их количество и регистры Мастера для чтения/записи данных:



Открывшееся окно содержит следующие настройки:

Name – Имя запроса (латинские буквы и цифры)

Storage Type – Код функции Modbus

Control Mode – Метод отправки запросов. ***Cyclic*** – циклически в автоматическом режиме, **Trigger**– запрос отправляется по переднему фронту назначенного регистра. Регистр можно посмотреть во вкладке ***Internal I/O Mapping***. При выборе режима Trigger появится переменная **Trigger control bit**.

Cyclic Time – Время цикла опроса в миллисекундах. Для метода ***Cyclic***

Number of retransmissions – Количество повторных запросов

Note – Комментарии к запросу (справочная информация, к самому запросу отношения не имеет)

Read/Write Register – блок, в котором оформляется чтение/запись данных из Ведомого

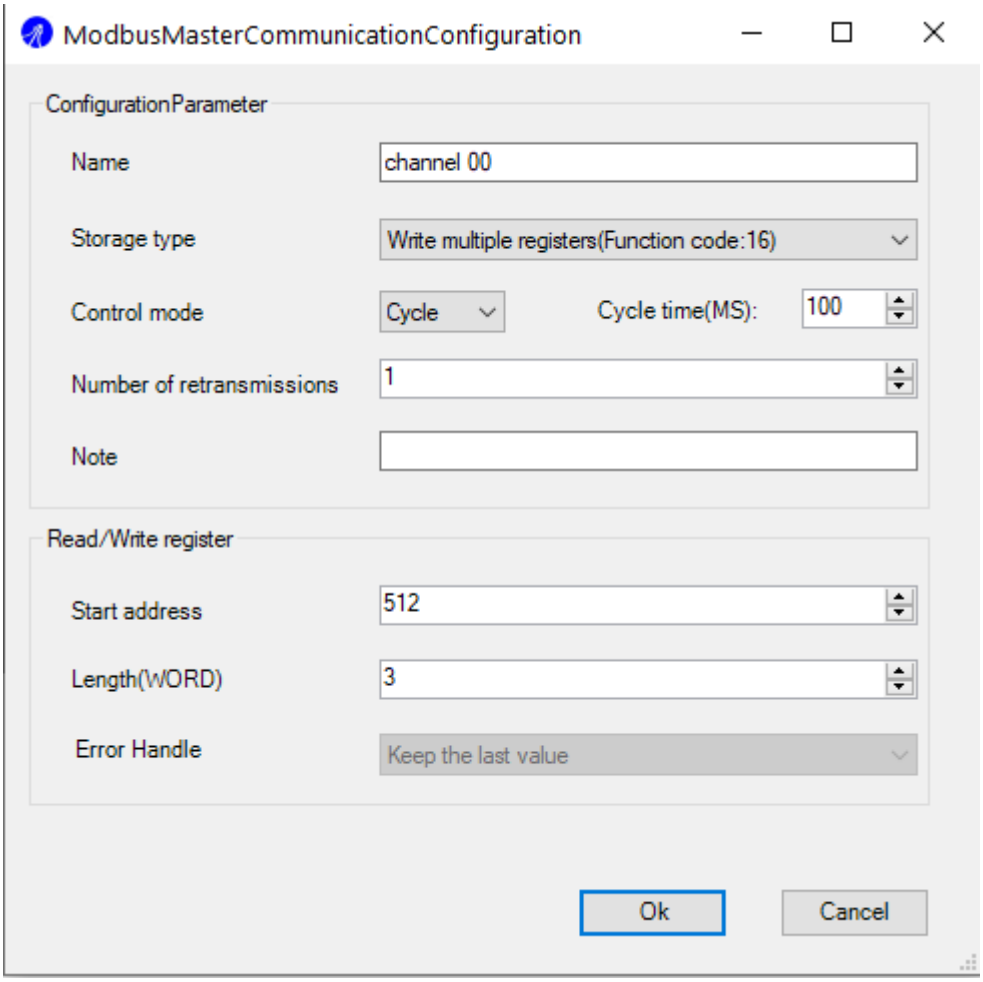
Start Address – Адрес регистра Ведомого, задаётся в HEX записью вида 0x0000

Length – Длина данных для чтения/записи

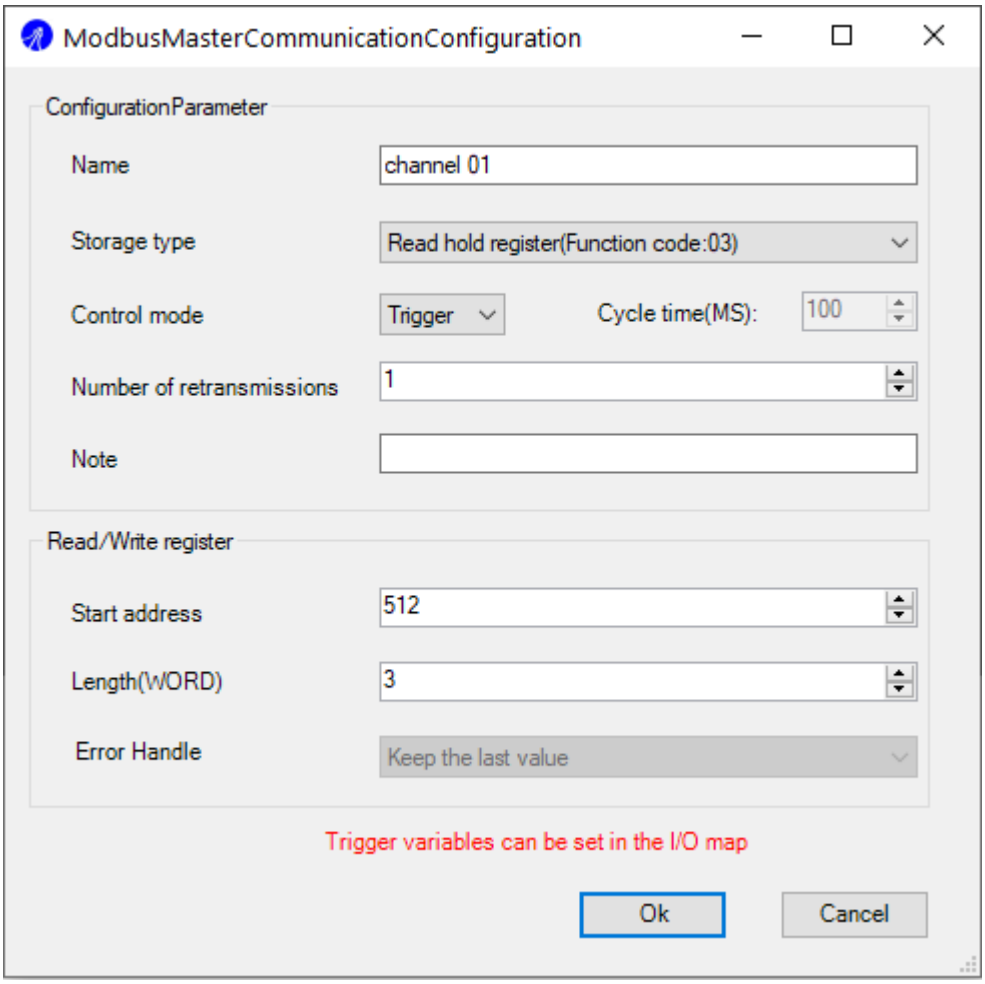
Error Handling – Фиксировано на сохранять состояние регистров при потере связи

Ниже для примера оформлено два запроса Channel_0 и Channel_1 для записи и чтения регистров V0, V1 и V2 контроллера Optimus Drive AT16S0T-RU, который выступает в качестве Ведомого устройства

Запись. Имя – Channel 00, Команда 16 (групповая запись регистров), метод опроса – Cycle циклический с тактом в 100 мс, Начальный регистр для чтения задаётся в десятичном виде без смещения – 512 (0x0200), количество последовательно записываемых регистров 3 штуки.



Чтение. Имя – Channel 01, Команда 3 (групповое чтение регистров), метод опроса – Trigger, из программы при помощи переменной. Начальный регистр для чтения задаётся в десятичном виде без смещения – 512 (0x0200) соответствует регистру V0 в контроллере AT16S0T, количество последовательно считываемых регистров 3 штуки.

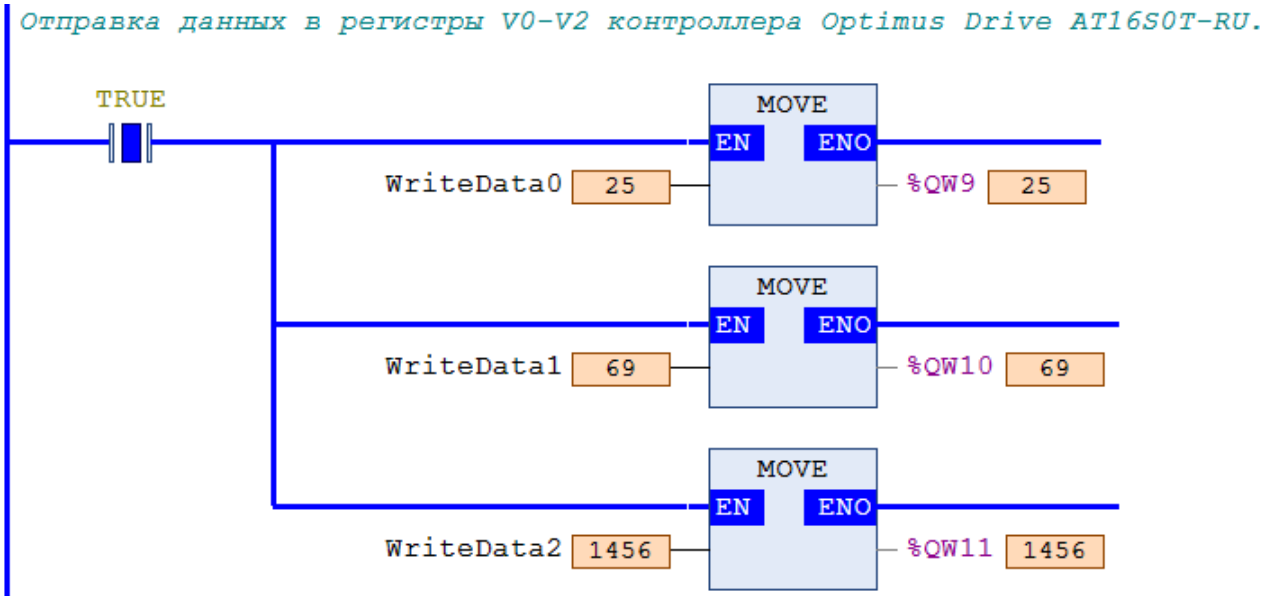


После создания каналов опроса система автоматически выделит под них регистры в соответствии с заявленным количеством регистров в Ведомом устройстве. В нашем примере будет 3 регистра на запись и 3 на чтение. Посмотреть можно в пункте **Internal I/O Mapping**:

Channel	Address	Type	Unit	Description
ErrorCode	%IW36	WORD		ModbusRTU ErrorCode
channel 00	%QW9	ARRAY [0..2] OF WORD		Write multiple registers
channel 00[0]	%QW9	WORD		Write #0200
channel 00[1]	%QW10	WORD		Write #0201
channel 00[2]	%QW11	WORD		Write #0202
channel 00	%IX74.0	BOOL		Processing complete state
channel 01	%IW38	ARRAY [0..2] OF WORD		Read hold register
channel 01[0]	%IW38	WORD		Read #0200
channel 01[1]	%IW39	WORD		Read #0201
channel 01[2]	%IW40	WORD		Read #0202
channel 01	%IX82.0	BOOL		Processing complete state
channel 01	%QX24.0	BOOL		Trigger control bit

Приведённые выше регистры можно напрямую использовать в программе для отправки/приёма данных от ведомого устройства. Сам обмен будет осуществляться автоматически. Также, можно создать свои переменные поверх выделенных системой регистров.

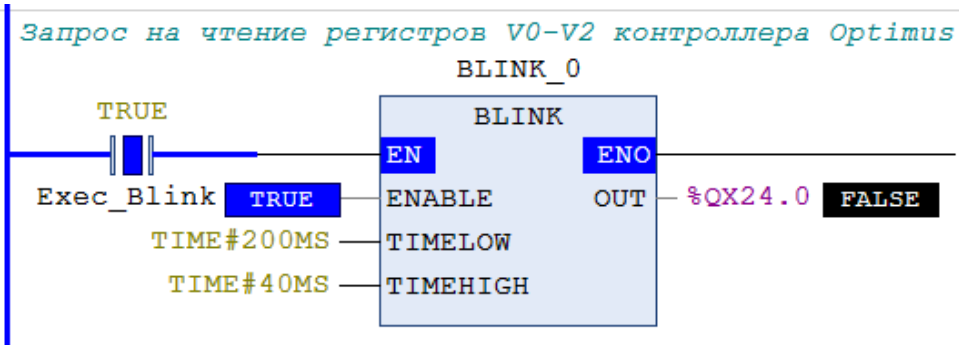
Запись данных в Ведомое устройство (Write). Запросы отправляются циклически без дополнительных условий.



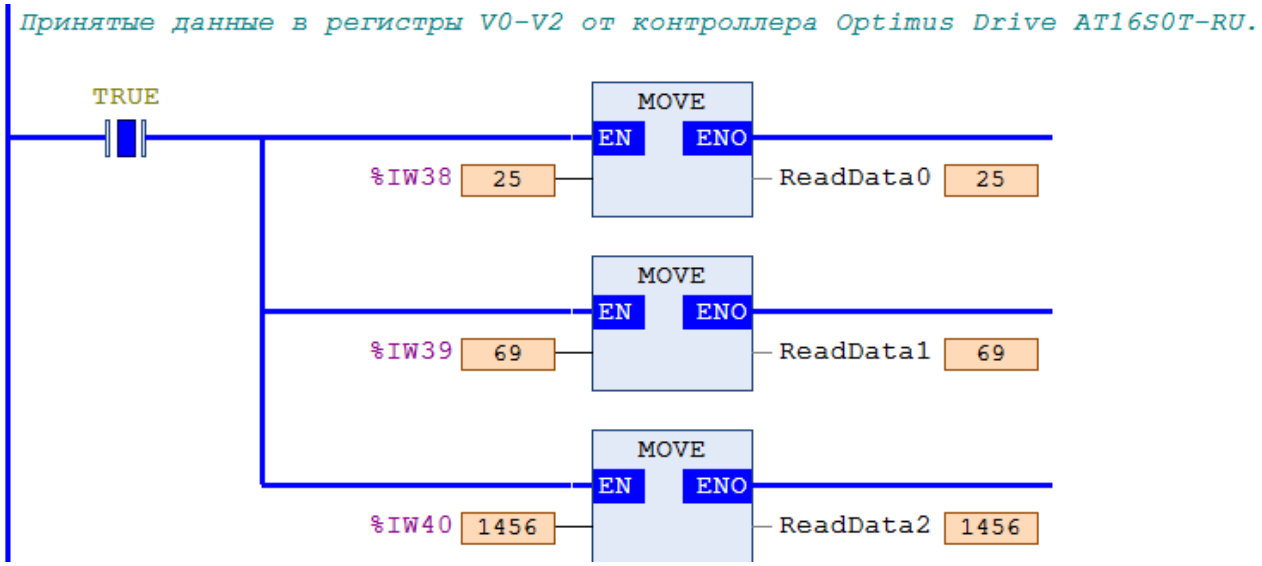
Приём данных от Ведомого устройства (Read). Запросы отправляются по тригеру:

channel 01	%QX24.0	BOOL		Trigger control bit

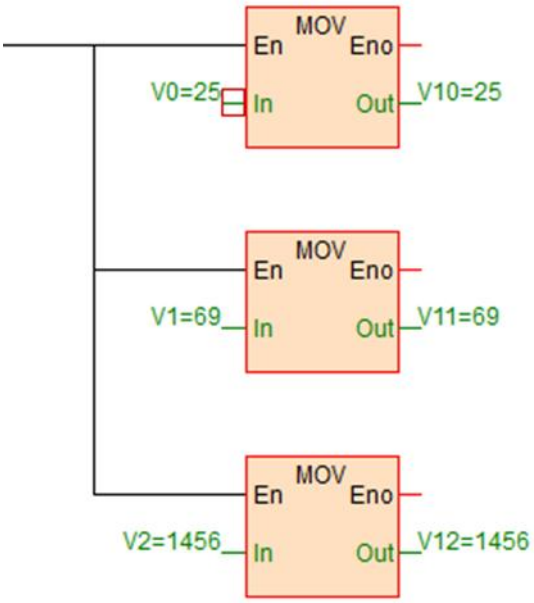
Триггер можно организовать при помощи команды Blink:



Принятые данные от Ведомого:



Мониторинг программы контроллера AT16S0T-RU:

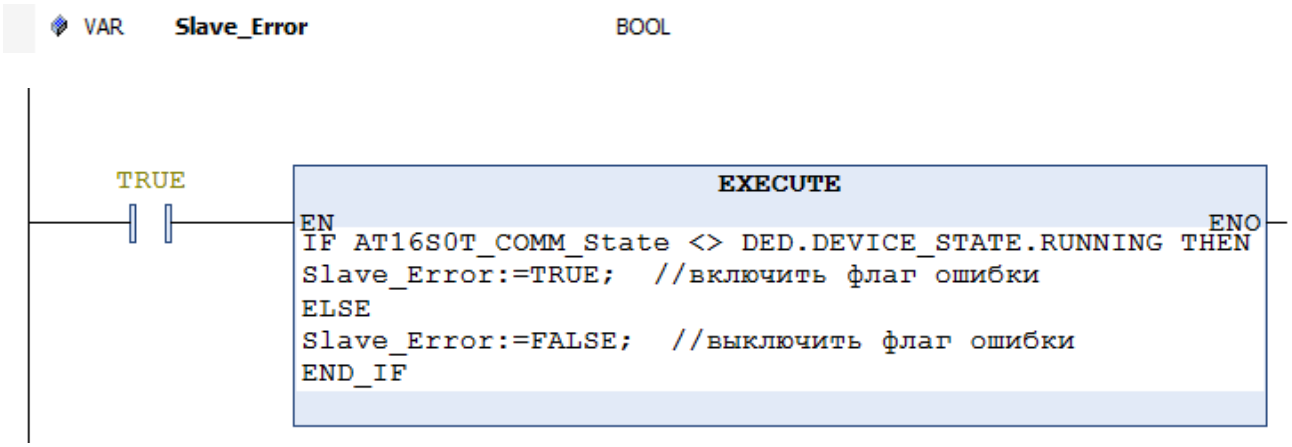


Ошибки связи можно проверять через свойство узла **GetDeviceState** и переменную типа **DED.DEVICE_STATE**

В нашем примере это переменная **AT16S0T_COMM_State**:

	Scope	Name	Address	Data type	Initialization
17	VAR	**AT16S0T_COMM_State**		DED.DEVICE_STATE	

В программе состояние связи с Ведомым удобно проверять с помощью кода в блоке **Execute**, в котором используется переменная **Slave_Error**:

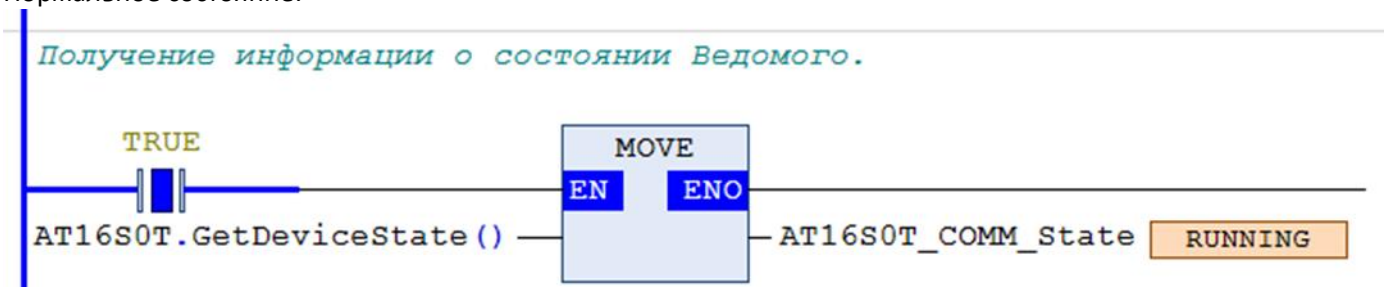


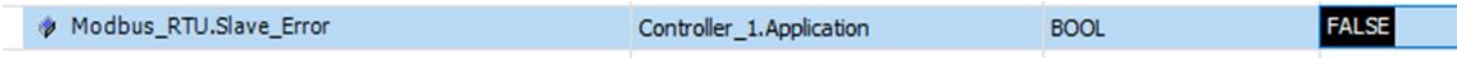
Текст программы из блока Execute:

```
IF AT16S0T_COMM_State <> DED.DEVICE_STATE.RUNNING THEN
Slave_Error:=TRUE;  //включить флаг ошибки
ELSE
Slave_Error:=FALSE;  //выключить флаг ошибки
END_IF
```

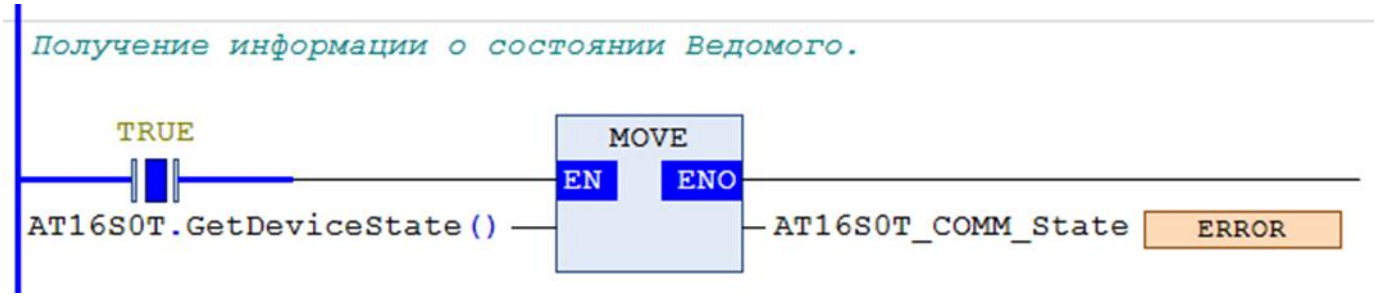
Мониторинг состояния Ведомого в программе:

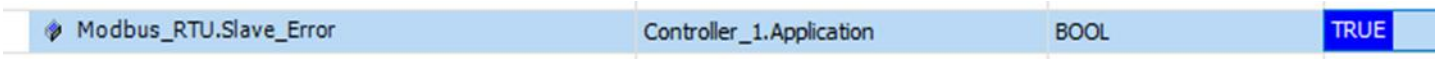
Нормальное состояние:





Ошибка связи:

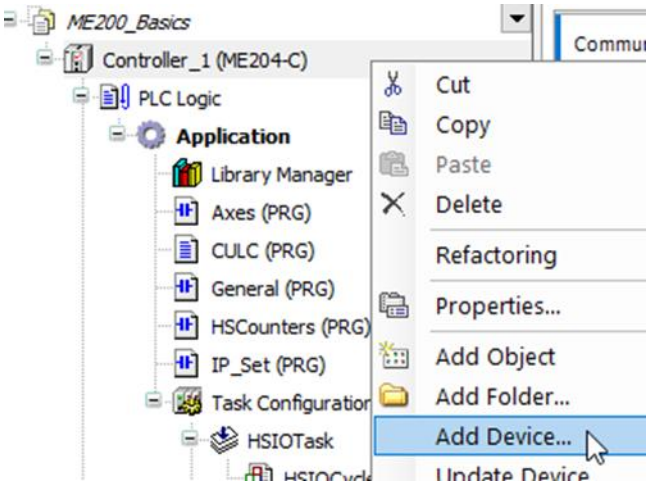




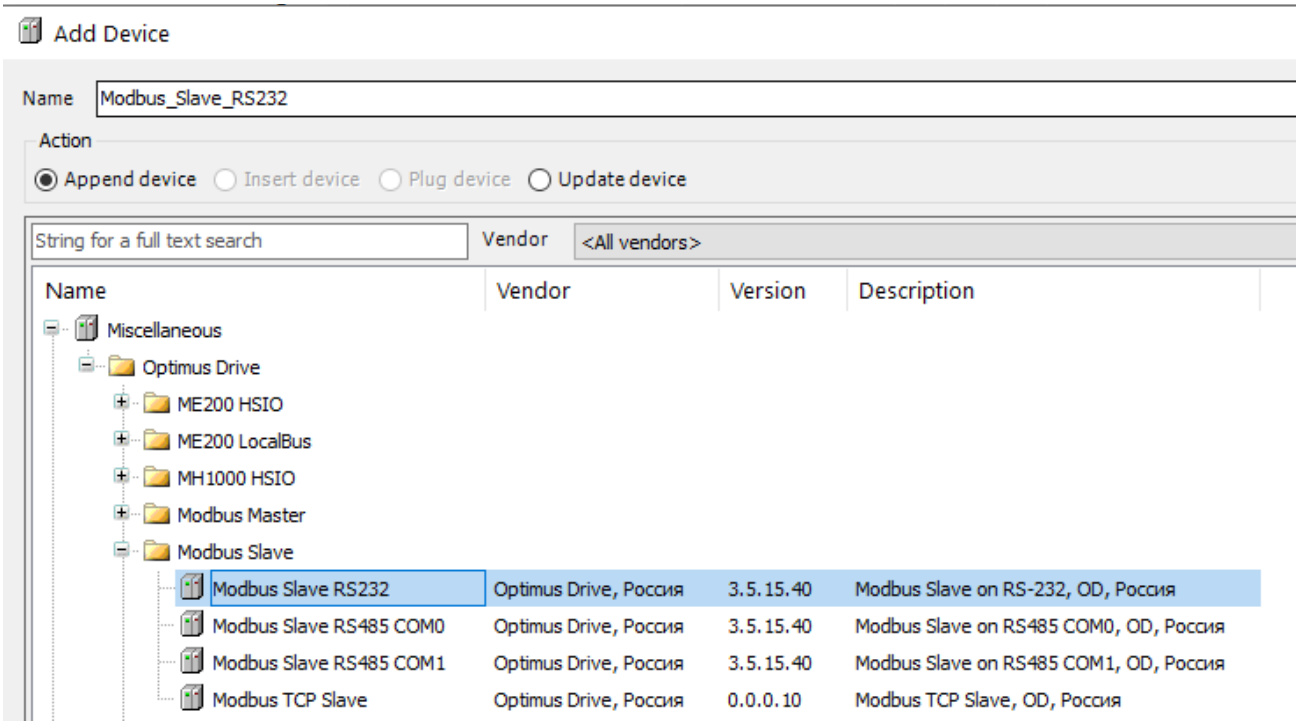
Последовательная связь по протоколу Modbus RTU Slave

Контроллеры серии ME200 имеют на борту 1 порт RS232 и 1 порт RS485. Оба порта могут работать как в режиме Мастера, так и Ведомого. В данной главе рассмотрена работа в режиме Ведомого по протоколу Modbus RTU. Назначение клемм портов приведено в предыдущей главе.

Для перевода порта последовательной связи в режим Modbus RTU Slave необходимо подключить к проекту адаптер связи типа **Modbus_Slave_*****. Для этого встаньте в древе проекта на пункт **Device** и нажмите правую кнопку мышки.
В появившемся меню выберите пункт **Add Device**:

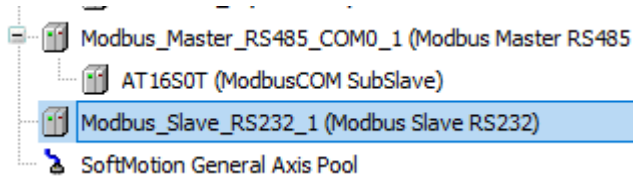


В открывшемся окне выберите пункт **Miscellaneous** и адаптер нужного порта, например RS232:



Нажмите **ОК**.

В древе проекта появится пункт **Modbus_Slave_RS232**:

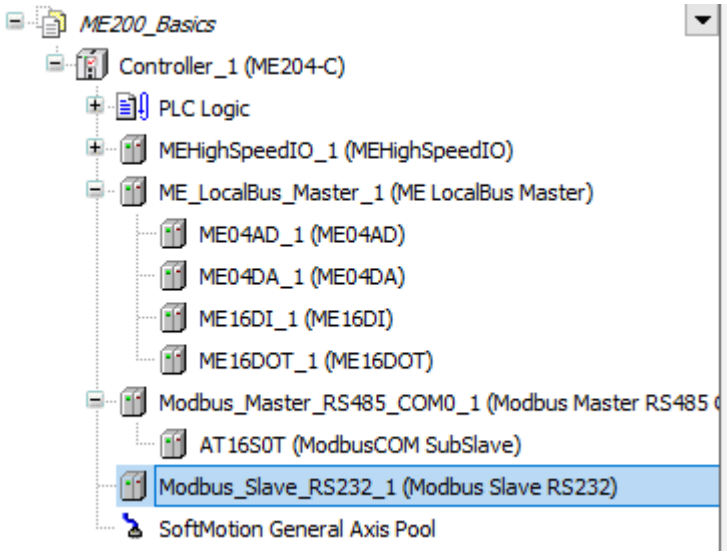


После добавления адаптера в проект и его загрузки, контроллер откроет доступ к своей памяти по протоколу Modbus RTU через соответствующий порт. Адресация будет одинаковая для всех портов. Согласно спецификации протокола Modbus, он имеет возможность читать булевы и словные регистры без обозначения типов данных в них. Поэтому таблица адресов Modbus будет выглядеть следующим образом:

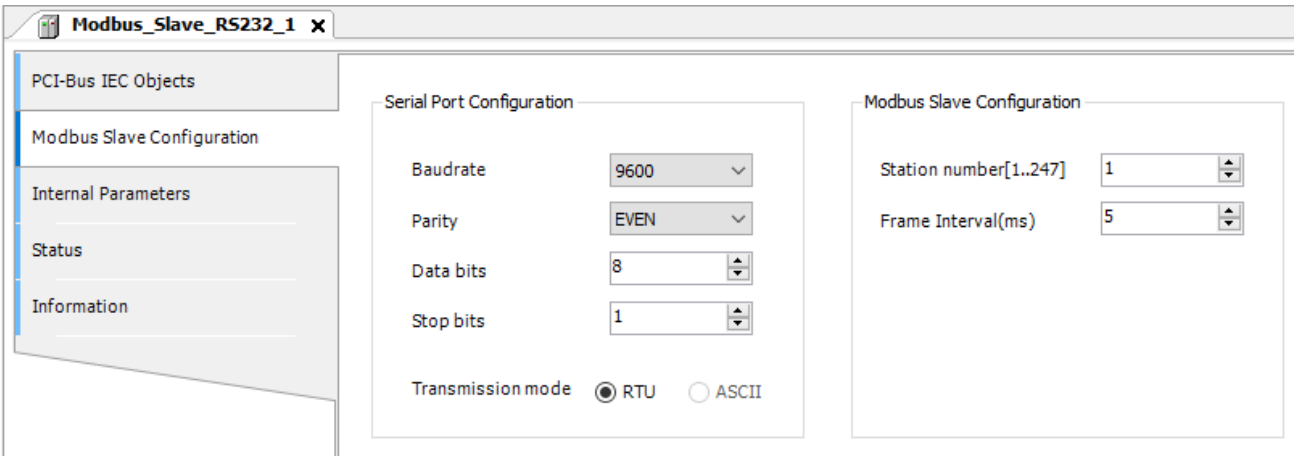
Тип	Диапазон	Функция	Инициализация	Количество
Q (выходы)	(QX0.0 ~ QX8191.7)	0x01,0x05,0x0f	0	65536
I (входы)	(IX0.0 ~ IX8191.7)	0x02	0	65536
M (данные)	%MW0~%MW65535	0x03,0x06,0x10	0	65536

Регистры типа %MW0~%MW65535 являются стандартными словами 16 бит. Адресация к битам в словах не поддерживается. Поэтому в программе биты типа %MX0.0… можно использовать, но их состояние нужно передавать как словный регистр, а потом на стороне Мастера разбирать по битам.
Например, маркеры %MX0.0.. %MX0.7 и %MX1.0.. %MX1.7 будут входить в состав регистра %MW0, маркеры %MX2.0.. %MX2.7 и %MX3.0.. %MX3.7 будут входить в состав регистра %MW1 и т.д.

Для настройки скорости передачи и протокола связи щёлкните два раза на пункте адаптера порта в древе проекта **Modbus_Slave_RS232**:



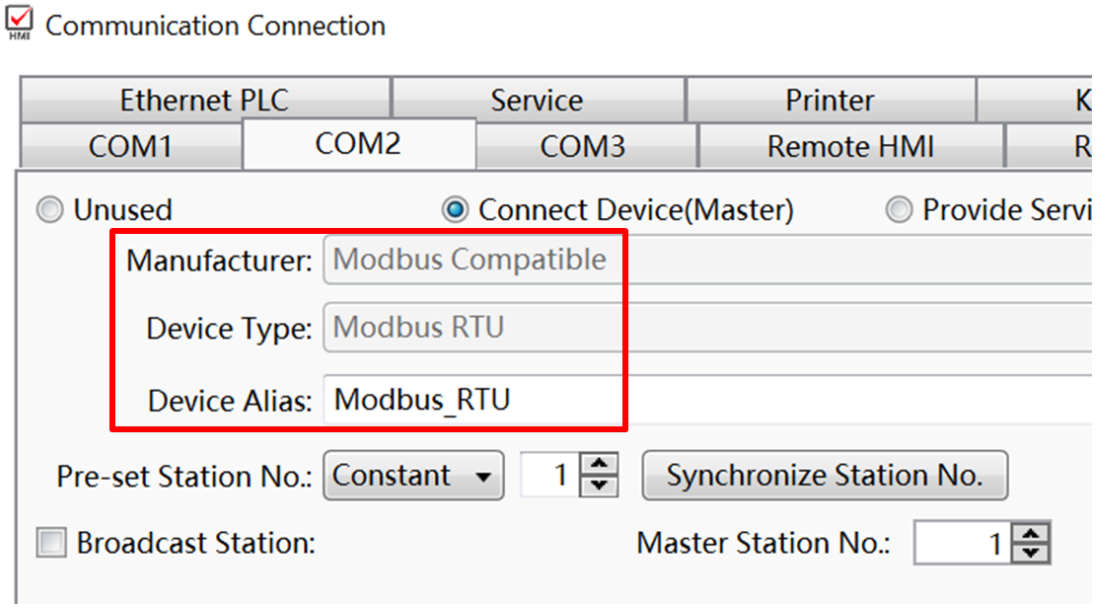
В открывшейся вкладке выберите пункт **Modbus Slave Configuration** и сделайте нужные настройки скорости и формата передачи данных:



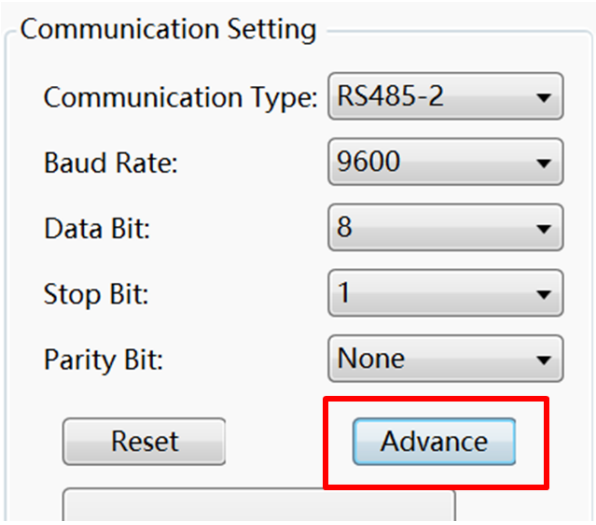
После загрузки проекта в контроллер, он будет в состоянии отвечать на Modbus RTU запросы с заданной скоростью и форматом данных (9600, 8,N,1 в нашем примере).
Режим Modbus ASCII контроллерами серии ME200 не поддерживается.

Контроллеры серии ME200 могут выступать в качестве Ведомого устройства для любого стандартного Мастера Modbus RTU. Например, можно рассмотреть в связь с панелью оператора Optimus Drive VI20-070S-FE-RU.

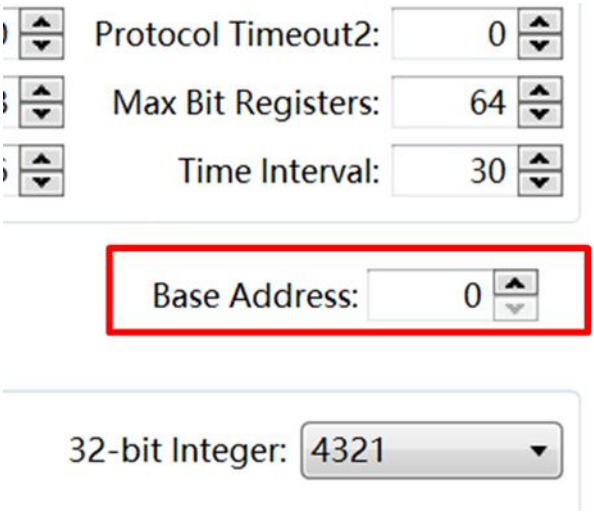
В панели необходимо выбрать стандартный Modbus RTU драйвер:



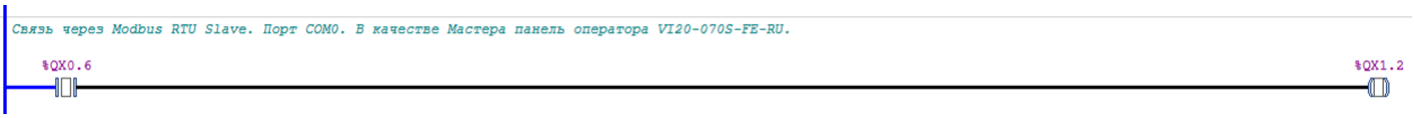
Задать скорость и протокол:



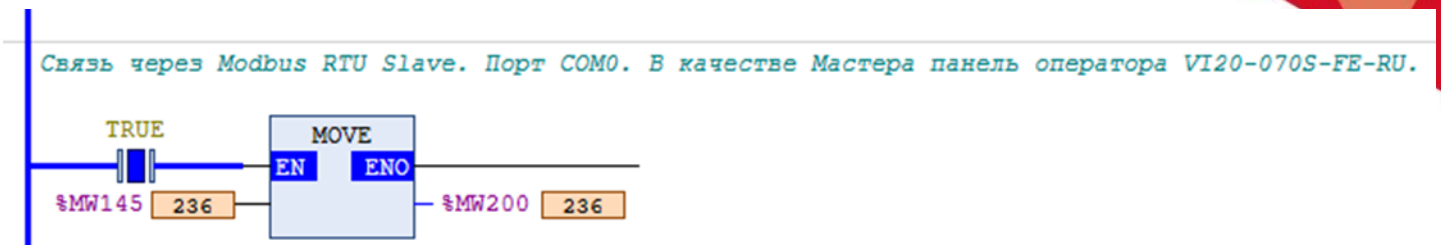
и во вкладке Advance убрать смещение адреса:



В программе контроллера создана простая программа для отображения данных, в которой задействованы следующие регистры:

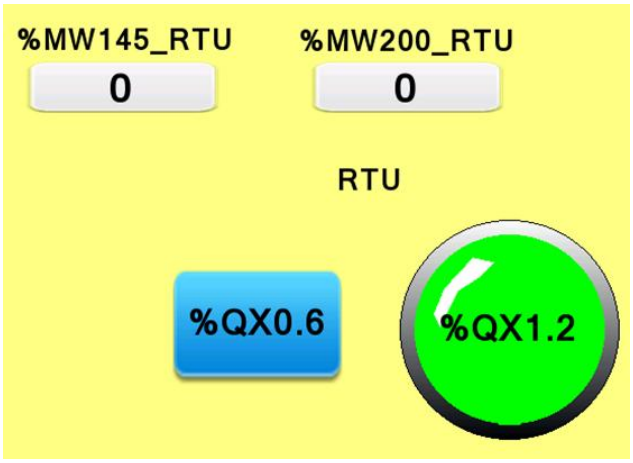


Регистр %QX0.6 - контакт, регистр %QX1.2 - выходная катушка.

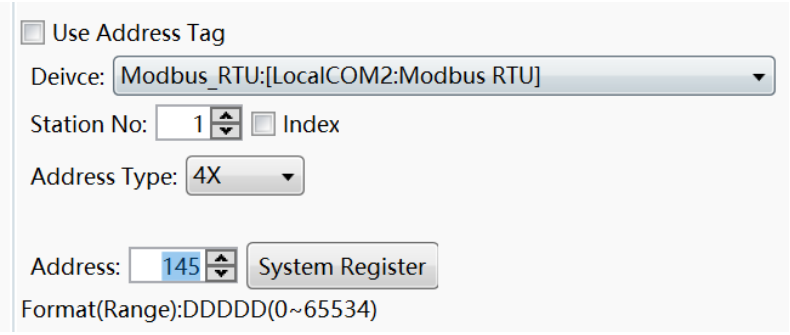


Регистр %MW145 - источник данных, регистр %MW200 - приёмник данных.

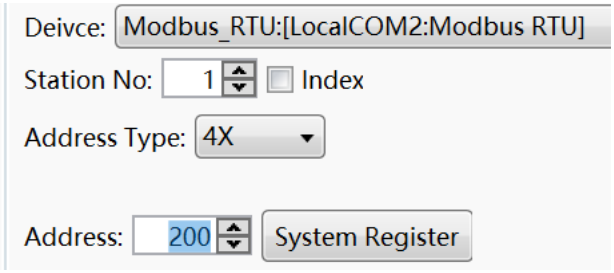
В панели оператора нарисован простой экран с четырьмя объектами соответственно: Input, Display, Button и Indicator.



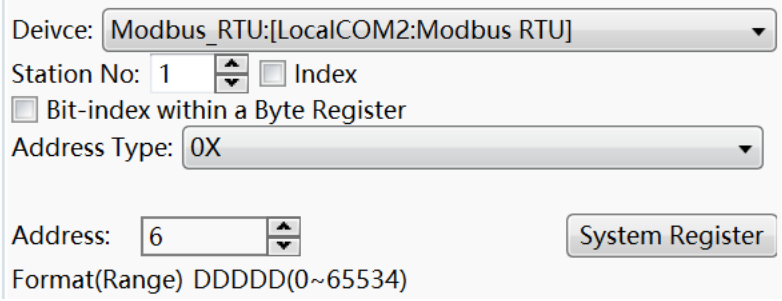
В панелях используется десятичное задание адреса, поэтому обращение к регистру %MW145 будет выглядеть так:



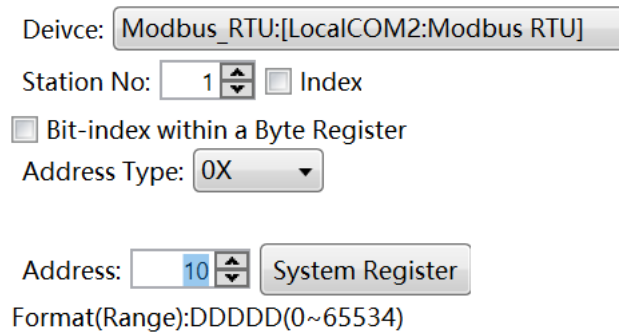
а к регистру %MW200 так



Таким образом, нужно просто указать номер регистра %MW****.
Обращение к булеву регистру %QX0.6 выглядит так:



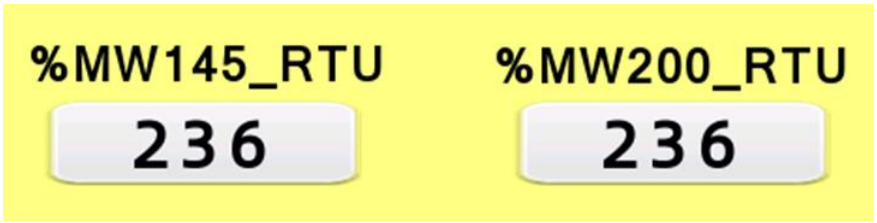
а к регистру %QX1.2 так:

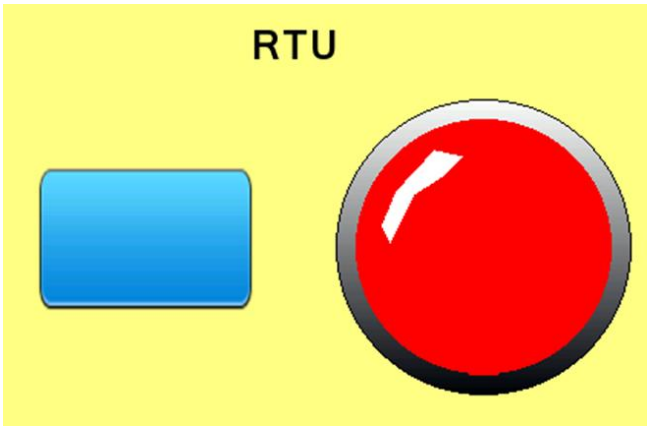


Т.е. идёт сплошная десятичная адресация. Каждый последующий регистр увеличивает адрес на 1.

%QX0.0.. %QX0.7 имеют десятичные адреса 0..7
%QX1.0.. %QX1.7 имеют десятичные адреса 8..15
%QX2.0.. %QX2.7 имеют десятичные адреса 16..23
и т.д.

При начале опроса контроллера панель будет отображать состояние регистров:





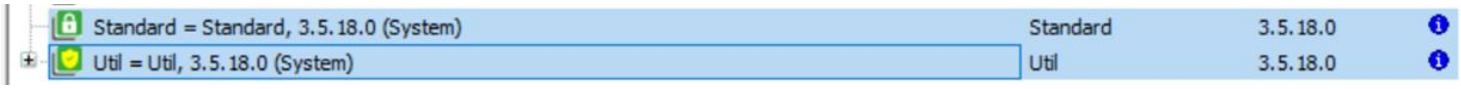
Связь по протоколу Modbus TCP Master

Контроллеры серии ME200 могут работать как в режиме Modbus TCP Client (Master), так и в режиме Modbus TCP Server (Slave), причём оба режима могут использоваться одновременно. В данной главе рассматривается организация связи контроллера с Ведомыми устройствами по протоколу Modbus TCP Client (Master).

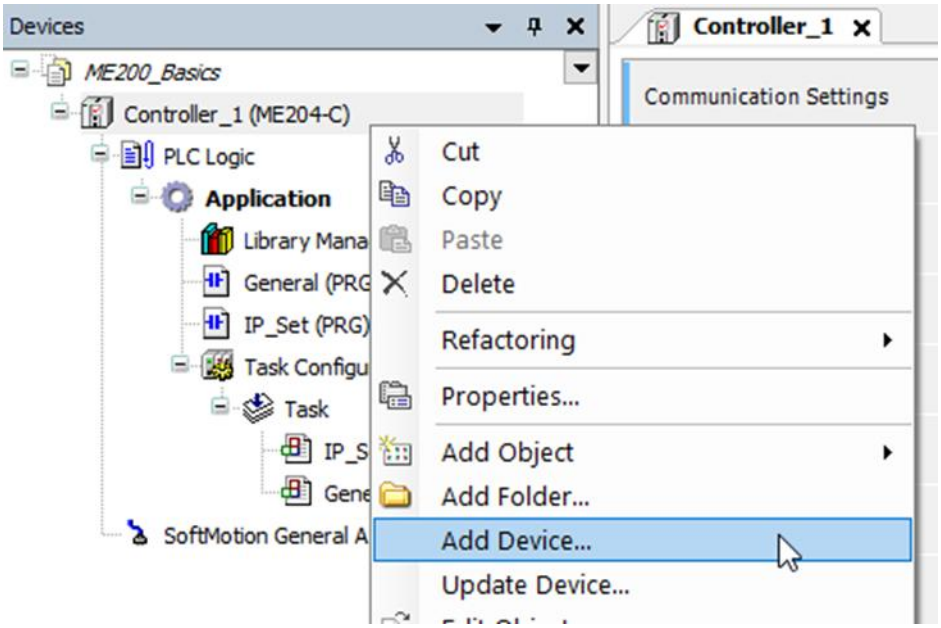
Для использования Modbus TCP Client (Master) к проекту должны быть подключены следующие библиотеки: **IoDrvEthernet, IoDrvModbusTCP и DED**.



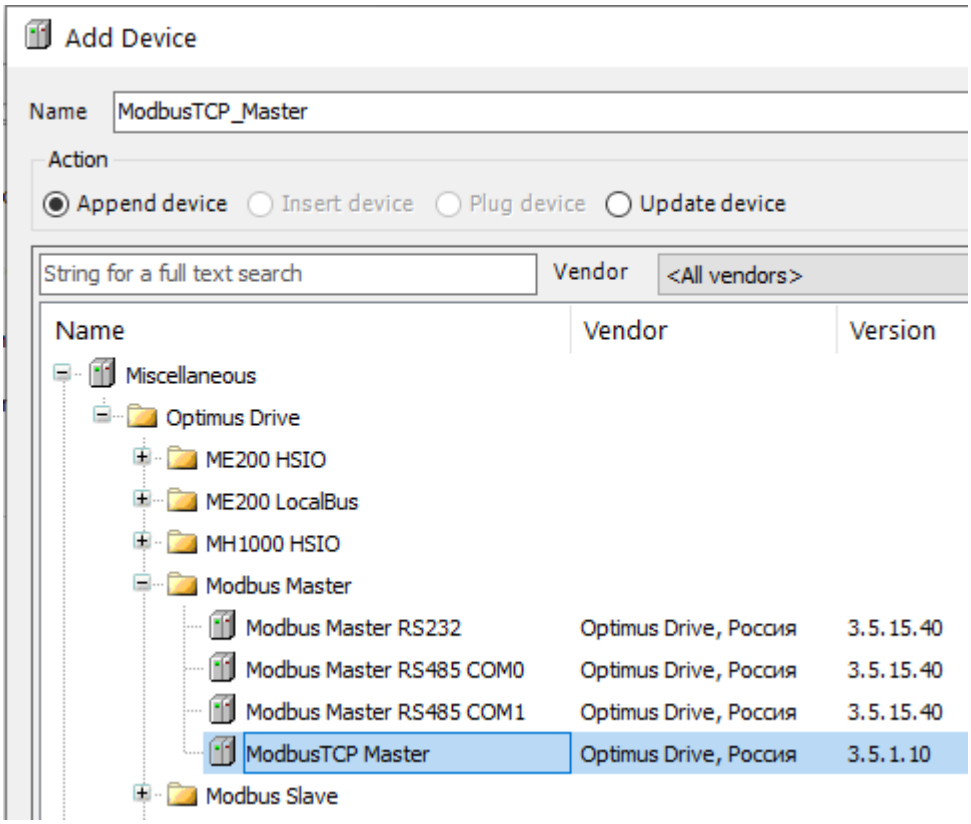
Также, для работы стандартного функционала типа переднего/заднего фронта, таймеров, счётчиков и т.п. к проекту должна быть подключена библиотека **Standard**, а для использования расхожего функционала типа фликеров и т.п. библиотека **Util**.



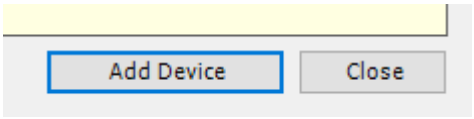
Для начала работы через порт Ethernet необходимо сначала добавить в проект устройство типа **Ethernet**. Для этого встаньте на пункт **Device** и щёлкните правой кнопкой мышки. В появившемся меню выберите пункт **Add Device**:



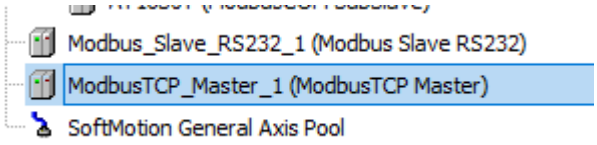
Выберите пункт **Miscellaneous – Optimus Drive – Modbus Master – Modbus TCP Master**:



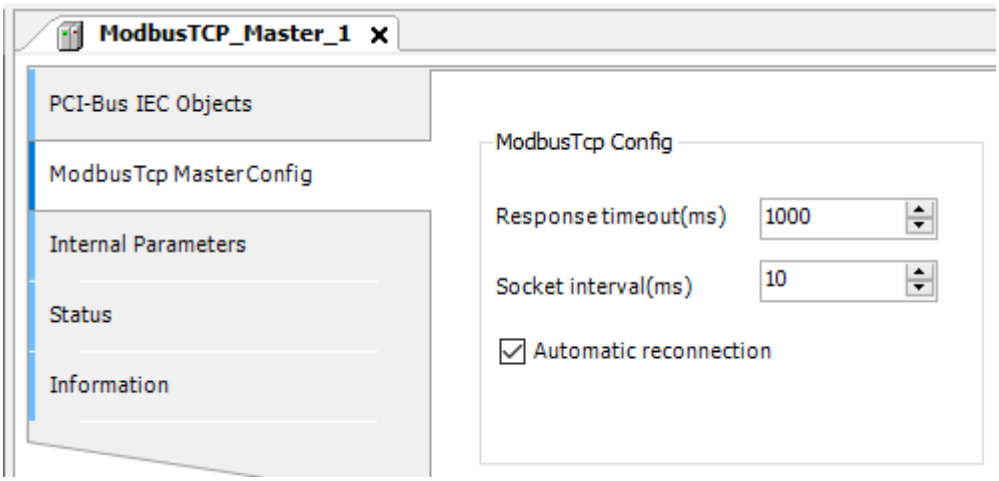
Нажмите кнопку **Add Device**:



В древе проекта появится пункт **Modbus TCP Master**:



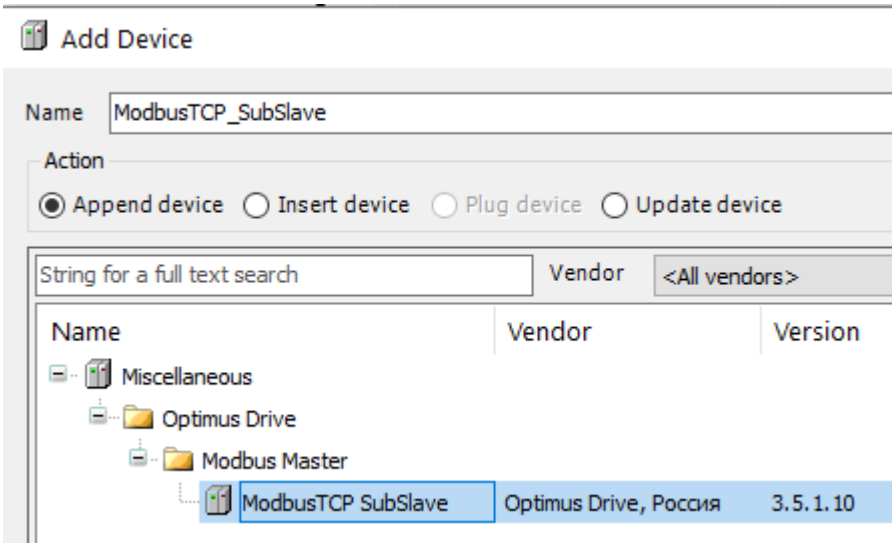
Щёлкните дважды левой кнопкой мышки на пункте **Modbus TCP Master**, откроется вкладка настроек параметров Мастера:



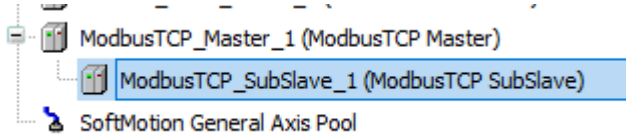
В разделе **Modbus TCP Master Config** необходимо задать:

Таймаут связи мс (Response Time (ms))
Задержка отправки пакетов мс (Socket interval (ms))
Автоматическое восстановление связи (Automatic reconnection) поставить флажок

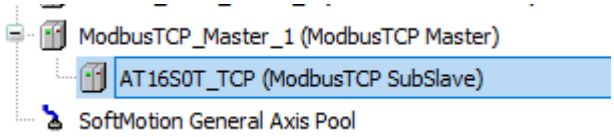
Контроллер готов к работе по протоколу Modbus TCP Client (Master) и осталось добавить линки, т.е. конкретные ведомые устройства, которые контроллер будет опрашивать. Для этого встаньте мышкой на пункт **Modbus TCP Master** и щёлкните правой кнопкой мышки и в отрывшемся меню выберите пункт **Add Device.**
В открывшемся окне выберите **Miscellaneous – Optimus Drive – Modbus Master – Modbus TCP SubSlave:**



У Вас в древе проекта появится пункт с Ведомым устройством,

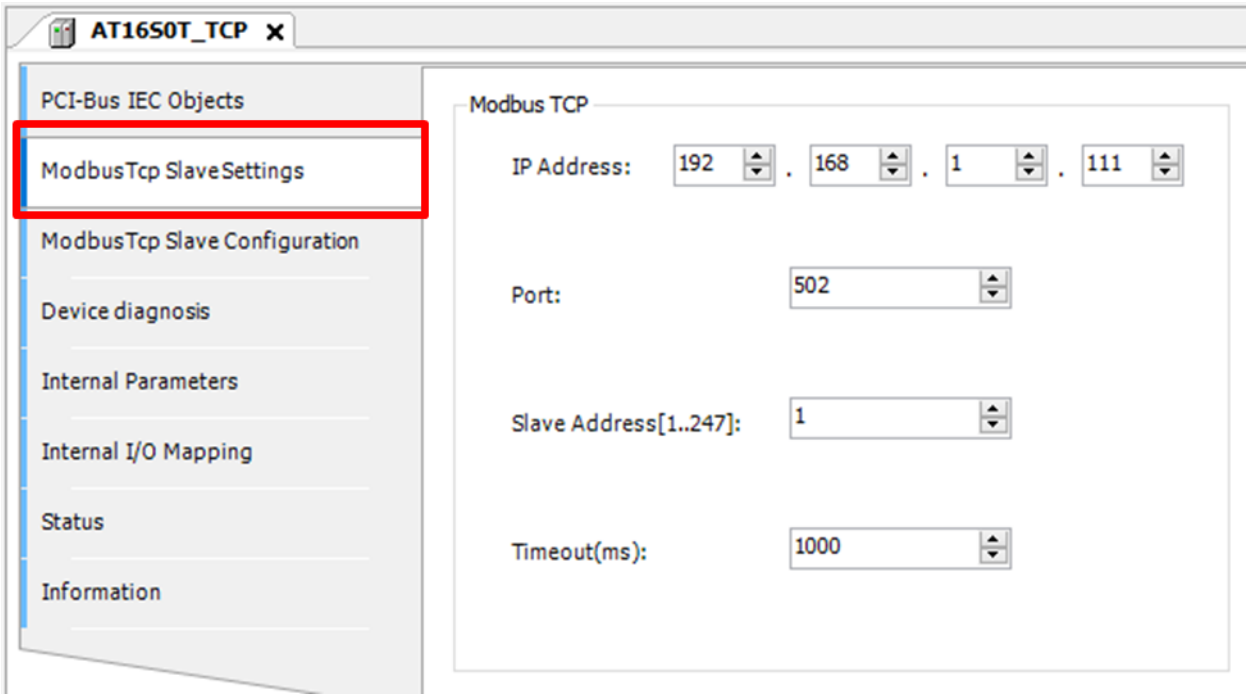


которому можно сразу переименовать на нужное название Ведомого. В нашем примере в качестве Ведомого будет использоваться контроллер Optimus Drive AT16S0T-RU

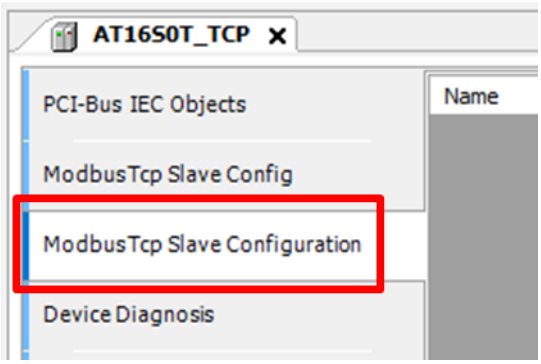


Имя узла - AT16S0T_TCP

Щёлкните дважды левой кнопкой мышки на Ведомом, откроется вкладка настройки обмена с Ведомым. В разделе **Modbus TCP Slave Config** установите IP адрес Ведомого (в нашем примере 192.168.1.111), Modbus адрес Ведомого (Slave Address), порт (Port), Таймаут (Timeout):

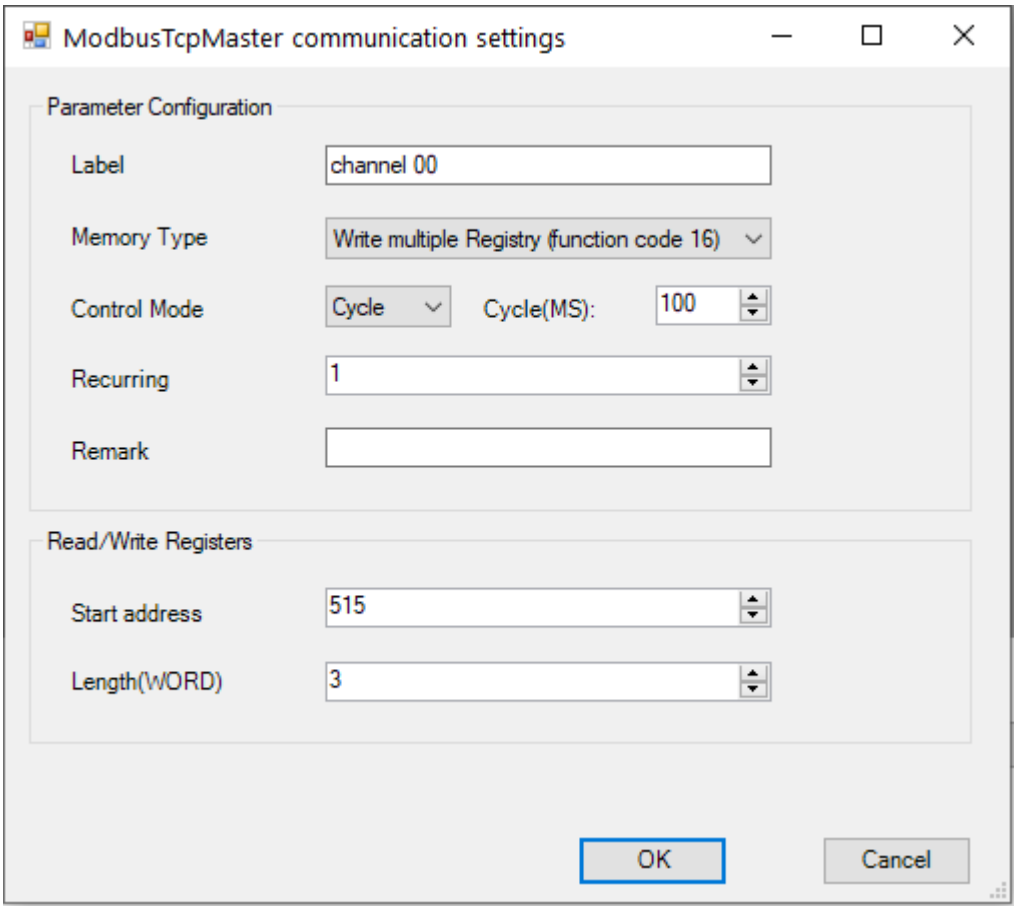


Далее во вкладке **Modbus Tcp Slave Configuration** необходимо создать нужное количество запросов:



Для оформления запроса нажмите кнопку **Add**. Откроется окно создания запроса:

Открывшееся окно содержит следующие настройки:



Label – Имя запроса (латинские буквы и цифры)

Memory Type – Код функции Modbus

Control Mode – Метод отправки запросов. ***Cyclic*** – циклически в автоматическом режиме, **Trigger**– запрос отправляется по переднему фронту назначенного регистра. Регистр можно посмотреть во вкладке ***Internal I/O Mapping***. При выборе режима Trigger появится переменная **Trigger control bit**.

Cycle(MS) – Время цикла опроса в миллисекундах. Для метода ***Cyclic***

Recurring – Количество повторных запросов

Remark – Комментарии к запросу (справочная информация, к самому запросу отношения не имеет)

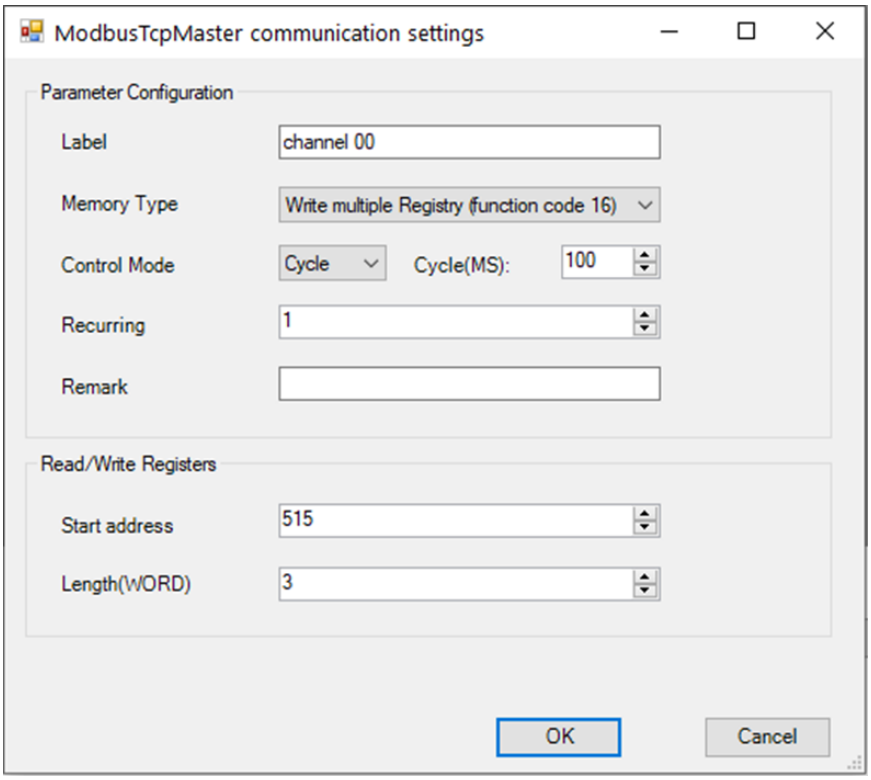
Read/Write Register – блок, в котором оформляется чтение/запись данных из Ведомого

Start Address – Адрес регистра Ведомого, задаётся в HEX записью вида 0x0000

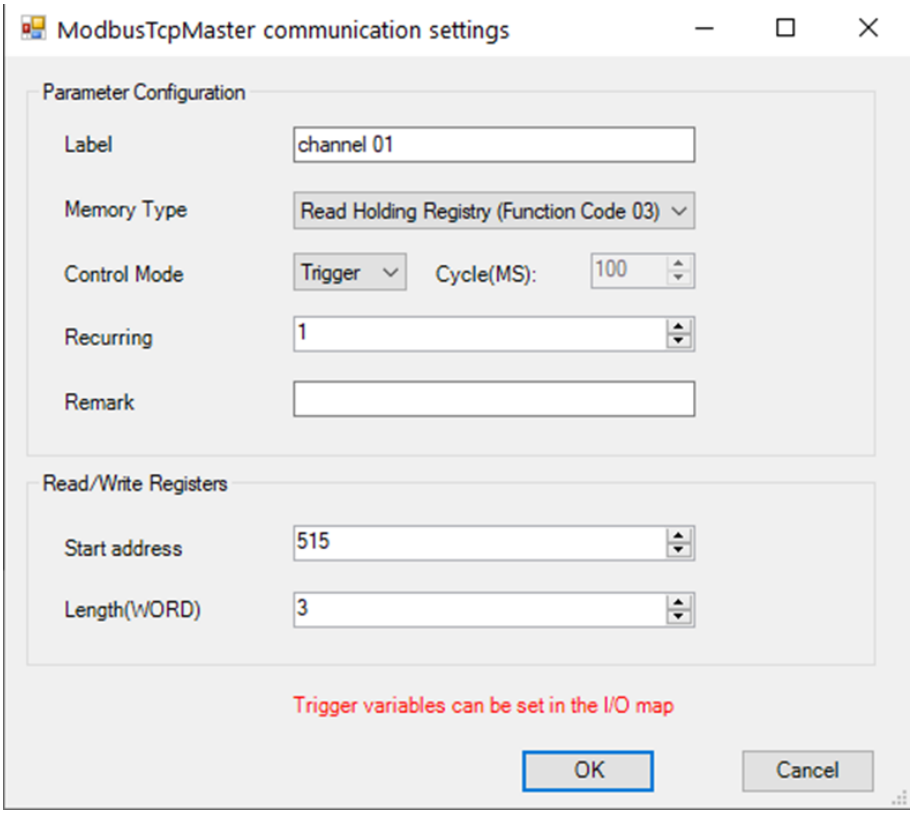
Length – Длина данных для чтения/записи

Ниже для примера оформлено два запроса Channel 00 и Channel 01 для записи и чтения регистров V3-V5 контроллера Optimus Drive AT16S0T-RU

Запись (адрес регистра записывается в десятичном виде без смещения, 515 это регистр V3 (0x0203)). Имя – Channel 00, Команда 16 (групповая запись регистров), метод опроса – циклический с тактом в 100 мс, Начальный регистр для записи 515 (0x0203), количество последовательно записываемых регистров 3 штуки.



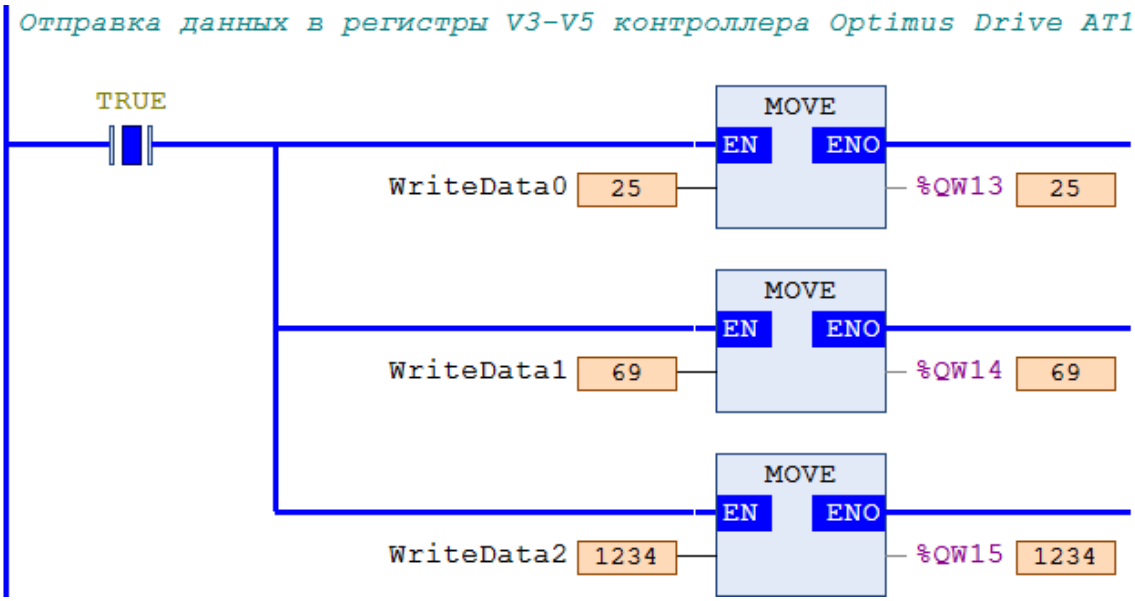
Чтение. Имя – Channel 01, Команда 3 (групповое чтение регистров), метод опроса – Trigger, из программы битом. Начальный регистр для чтения 515 (0x0203), количество последовательно считываемых регистров 3 штуки.



После создания каналов опроса система автоматически выделит под них регистры в соответствии с заявленным количеством регистров в Ведомом устройстве. В нашем примере будет 3 регистра на запись и 3 на чтение. Посмотреть можно в пункте **Internal I/O Mapping**:

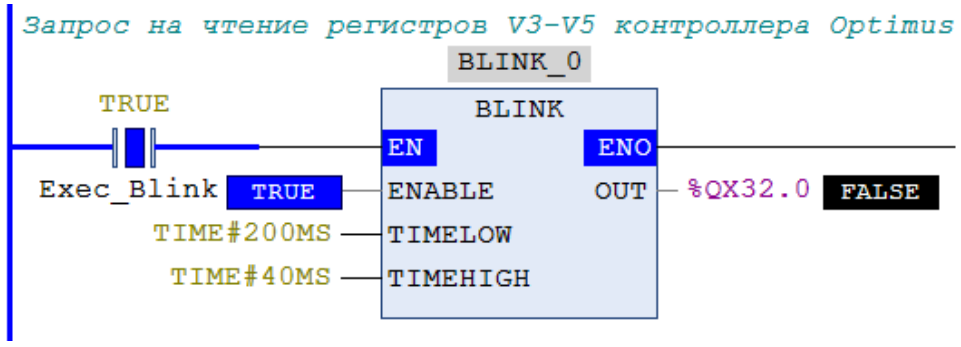
Variable	Mapping	Channel	Address	Type	Unit	Description
		ErrorCode	%IW42	WORD		ModbusTCP ErrorCode
		channel 00	%QW13	ARRAY [0..2] OF WORD		Write multiple Registry (function code 16)
		channel 00[0]	%QW13	WORD		Write #0515)
		channel 00[1]	%QW14	WORD		Write #0516)
		channel 00[2]	%QW15	WORD		Write #0517)
		channel 00	%IX86.0	BOOL		Processing complete state
		channel 01	%IW44	ARRAY [0..2] OF WORD		Read Holding Registry (Function Code 03)
		channel 01[0]	%IW44	WORD		Read #0515
		channel 01[1]	%IW45	WORD		Read #0516
		channel 01[2]	%IW46	WORD		Read #0517
		channel 01	%IX94.0	BOOL		Processing complete state
		channel 01	%QX32.0	BOOL		Trigger control bit

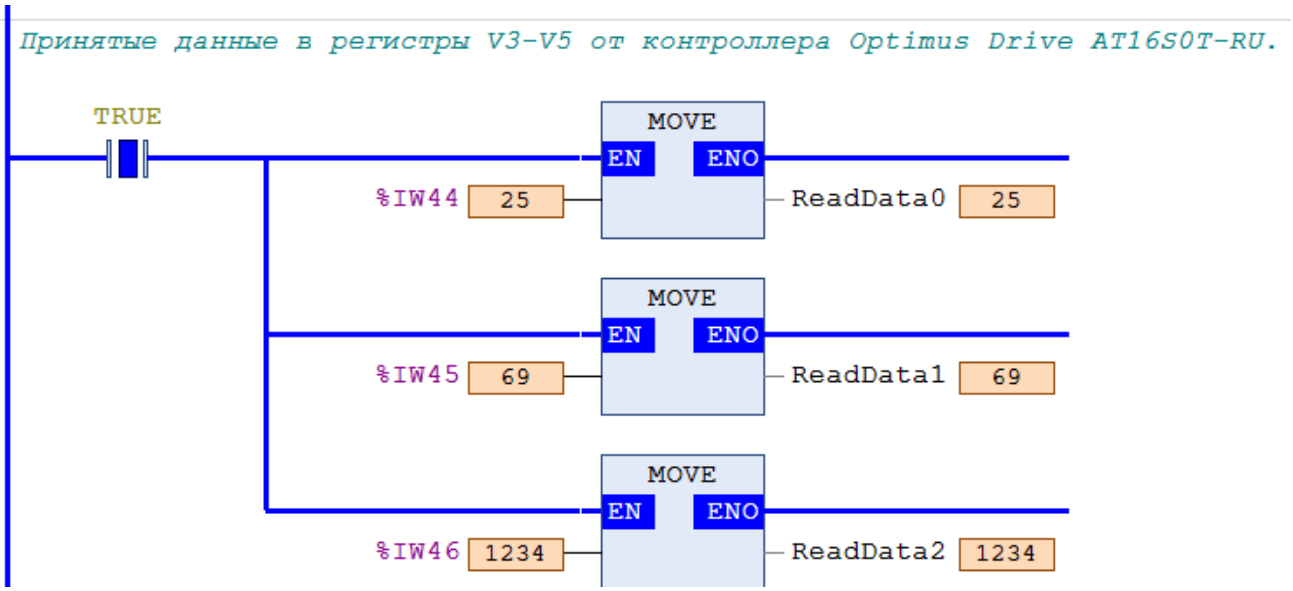
В нашем примере для записи выделены %QW13, %QW14 и %QW15. Для чтения выделены регистры %IW44, %IW45 и %IW46. Данные регистры можно напрямую использовать в программе для обмена данными с Ведомым:



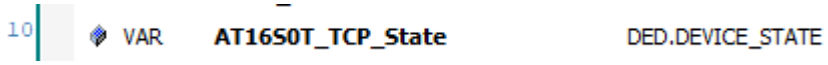
Для запуска триггера удобно использовать команду Blink:

		channel 01	%QX32.0	BOOL		Trigger control bit



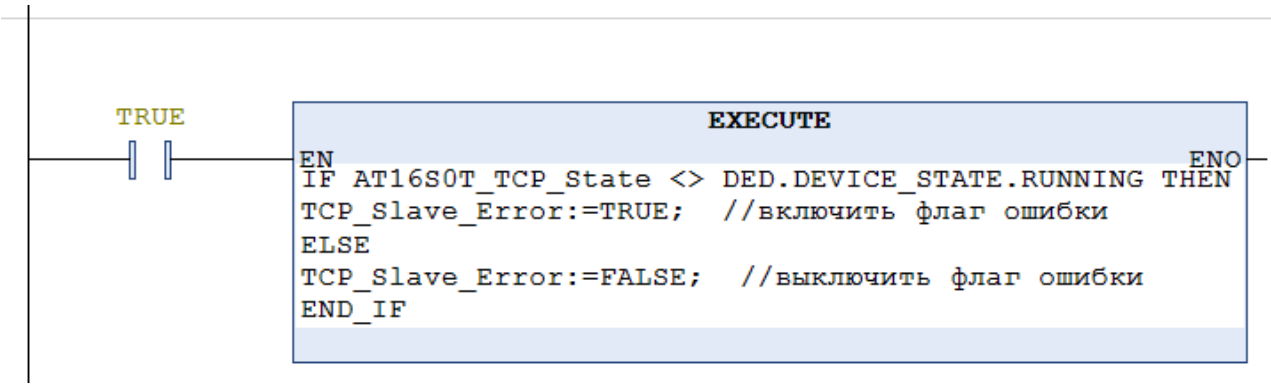


Ошибки связи можно проверять через свойство узла **GetDeviceState** и переменную типа **DED.DEVICE_STATE**



В программе состояние связи с Ведомым удобно проверять с помощью кода в блоке Execute:

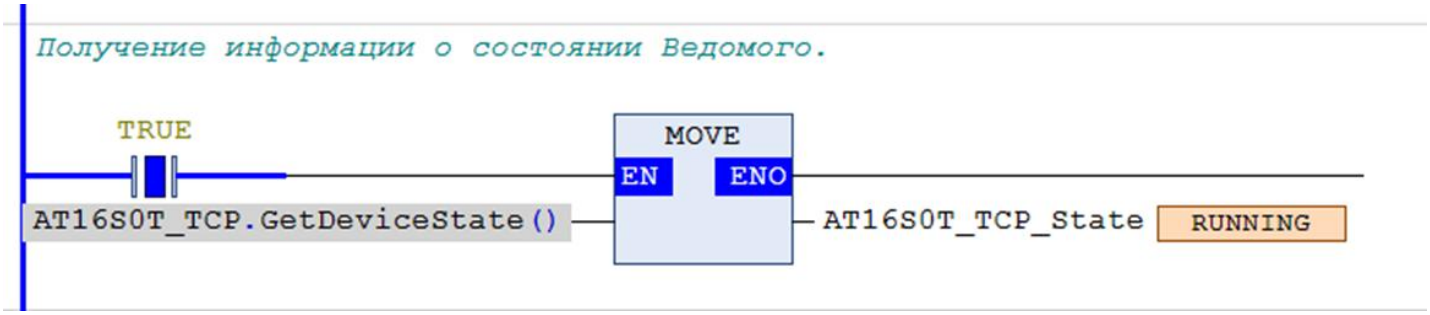


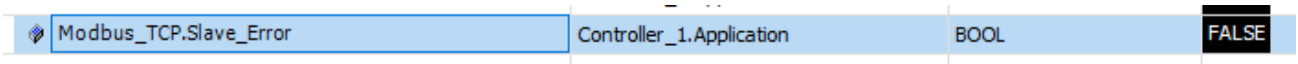


Текст программы из блока Execute:

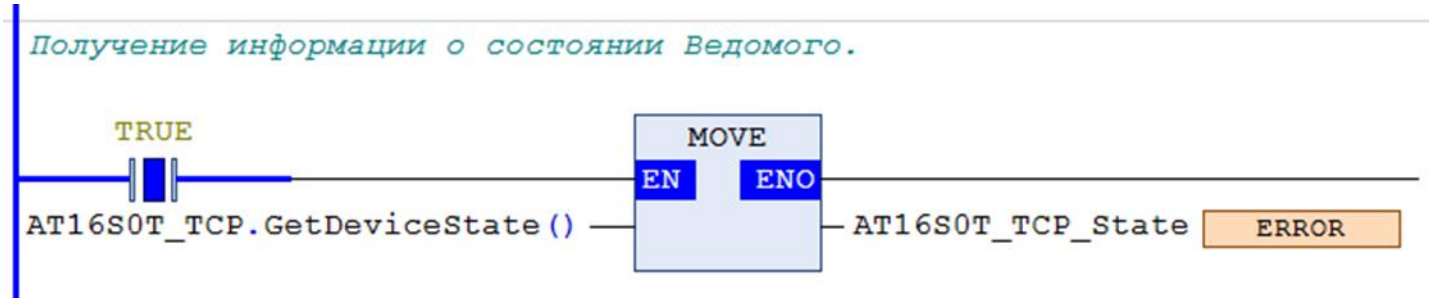
```
IF AT16S0T_TCP_State <> DED.DEVICE_STATE.RUNNING THEN
TCP_Slave_Error:=TRUE;  //включить флаг ошибки
ELSE
TCP_Slave_Error:=FALSE;  //выключить флаг ошибки
END_IF
```

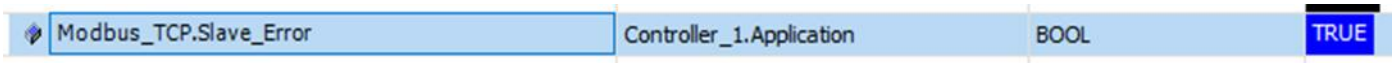
Нормальное состояние ведомого:





Потеря связи с ведомым:

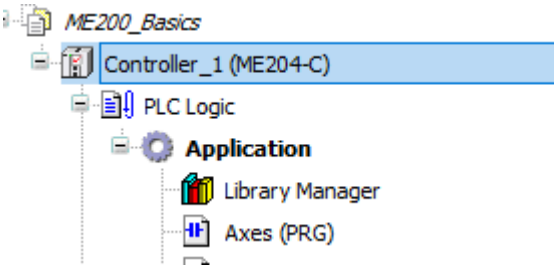




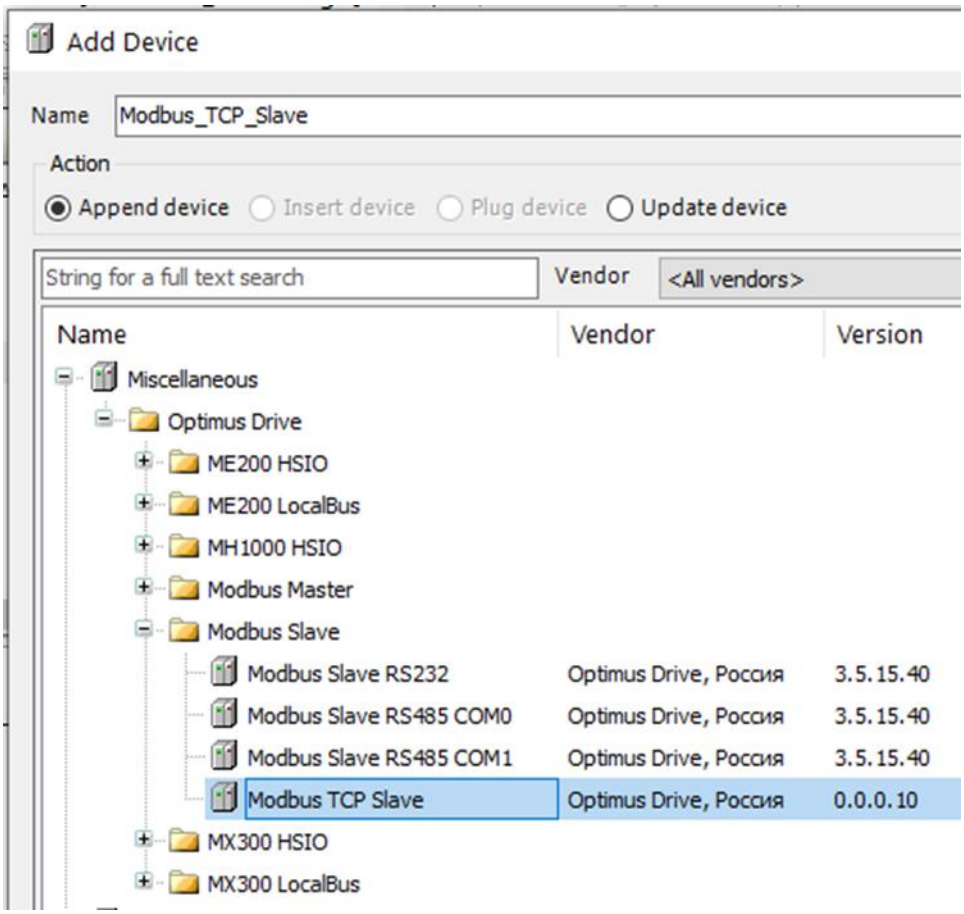
Связь по протоколу Modbus TCP Slave

Контроллеры серии ME200 могут работать как в режиме Modbus TCP Client (Master), так и в режиме Modbus TCP Server (Slave), причём оба режима могут использоваться одновременно. В данной главе рассматривается организация связи с контроллером по протоколу Modbus TCP Server (Slave).

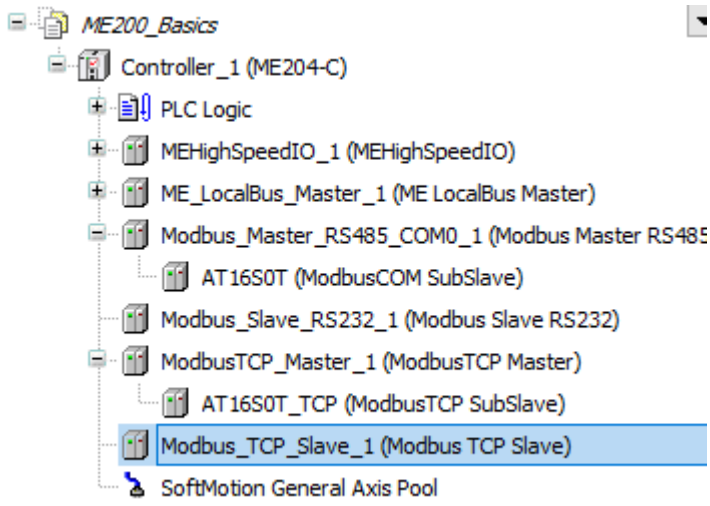
Для организации связи с контроллером в режиме Modbus TCP Server в древо проекта необходимо добавить адаптер
Щёлкните правой кнопкой мышки на пункте **Device (ME204-C)** и в отрывшемся меню выберите пункт **Add Device.**



В открывшемся окне выберите пункт **Miscellaneous – Optimus Drive – Modbus Slave – Modbus TCP Slave**:



В древе проекта появится пункт **Modbus TCP Slave**:
(идёт в древе проекта непосредственно от головного узла Device)



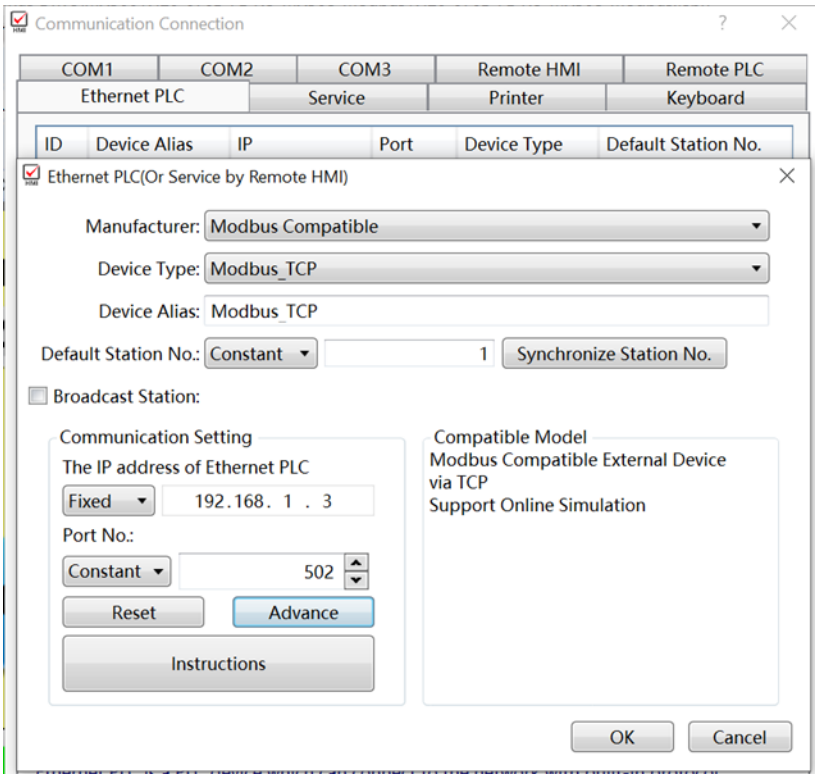
После добавления адаптера в проект и его загрузки, контроллер откроет доступ к своей памяти по протоколу Modbus TCP через порт Ethernet. Согласно спецификации протокола Modbus, он имеет возможность читать булевы и словные регистры без обозначения типов данных в них. Поэтому таблица адресов Modbus будет выглядеть следующим образом (начальный адрес 0x0000):

Тип	Диапазон	Функция	Инициализация	Количество
Q (выходы)	(QX0.0 ~ QX8191.7)	0x01,0x05,0x0f	0	65536
I (входы)	(IX0.0 ~ IX8191.7)	0x02	0	65536
M (данные)	%MW0~%MW65535	0x03,0x06,0x10	0	65536

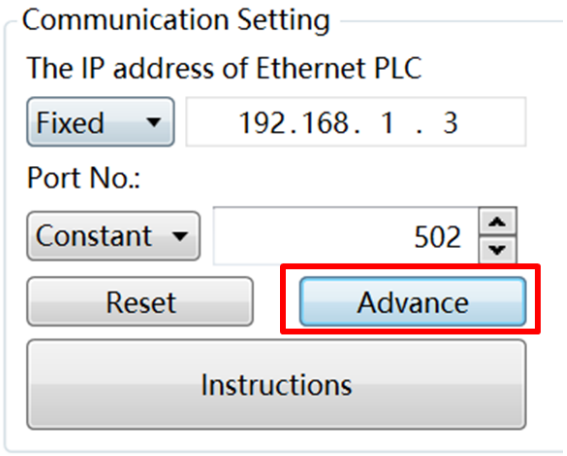
Регистры типа %MW0~%MW65535 являются стандартными словами 16 бит. Адресация к битам в словах не поддерживается. Поэтому в программе биты типа %MX0.0… можно использовать, но их состояние нужно передавать как словный регистр, а потом на стороне Мастера разбирать по битам.
Например, маркеры %MX0.0.. %MX0.7 и %MX1.0.. %MX1.7 будут входить в состав регистра %MW0, маркеры %MX2.0.. %MX2.7 и %MX3.0.. %MX3.7 будут входить в состав регистра %MW1 и т.д.

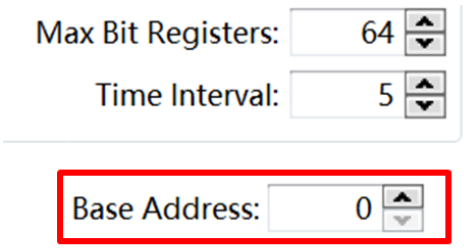
Контроллеры серии ME200 могут выступать в качестве Ведомого устройства для любого стандартного Мастера Modbus TCP. Например, можно рассмотреть в связь с панелью оператора Optimus Drive VI20-070S-FE-RU.

В панели необходимо выбрать стандартный Modbus TCP драйвер:

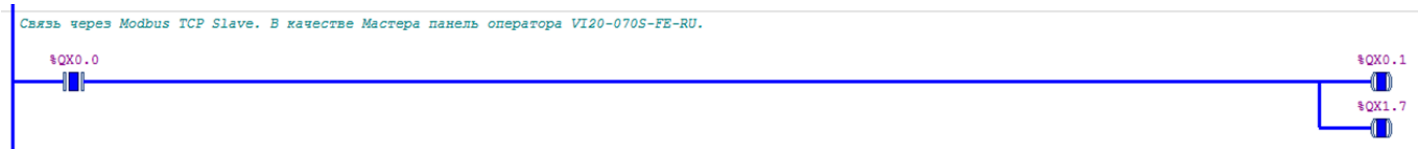


Указать IP адрес контроллера. В нашем примере 192.168.1.3 и во вкладке **Advance** убрать смещение адреса:





В программе контроллера создана простая программа для отображения данных, в которой задействованы следующие регистры:

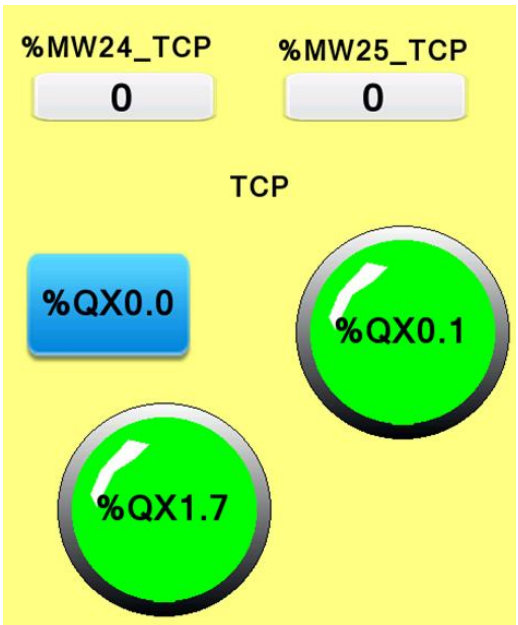


Регистр %QX0.0 - контакт, регистры %QX0.1 и %QX1.7 - выходные катушки.

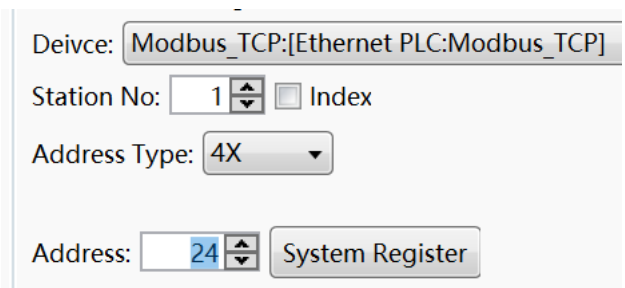


Регистр %MW24 - источник данных, регистр %MW25 - приёмник данных.

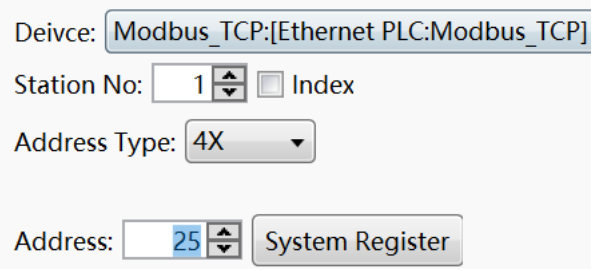
В панели оператора нарисован простой экран с пятью объектами соответственно: Input, Display, Button и 2 штуки Indicator.



В панелях используется десятичное задание адреса, поэтому обращение к регистру %MW24 будет выглядеть так:

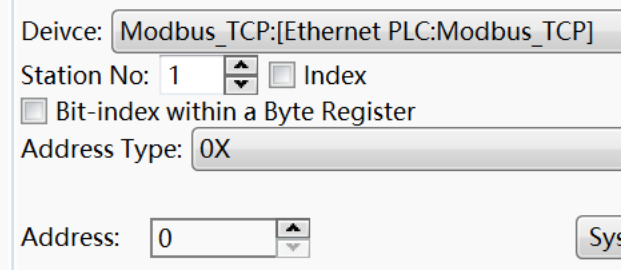


а к регистру %MW25 так

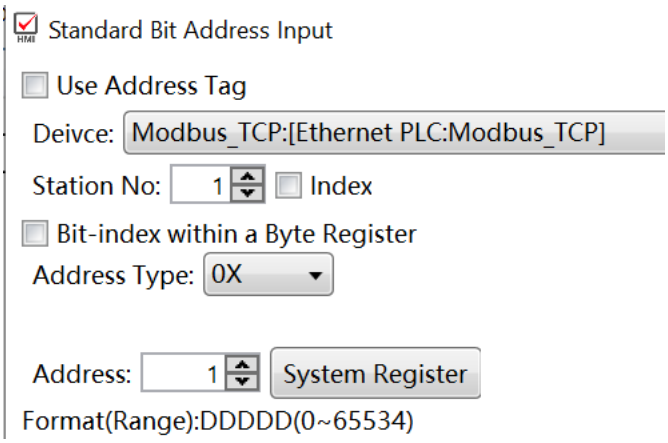


Таким образом, нужно просто указать номер регистра %MW****.

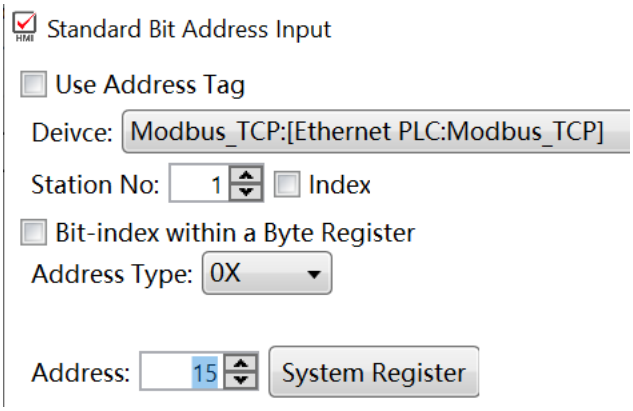
Обращение к булеву регистру %QX0.0 выглядит так:



к регистру %QX0.1 так:



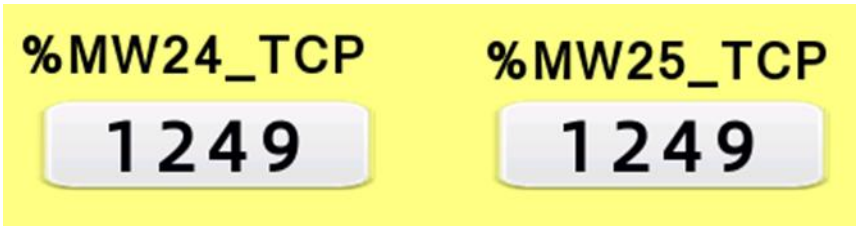
а к регистру %QX1.7 так:

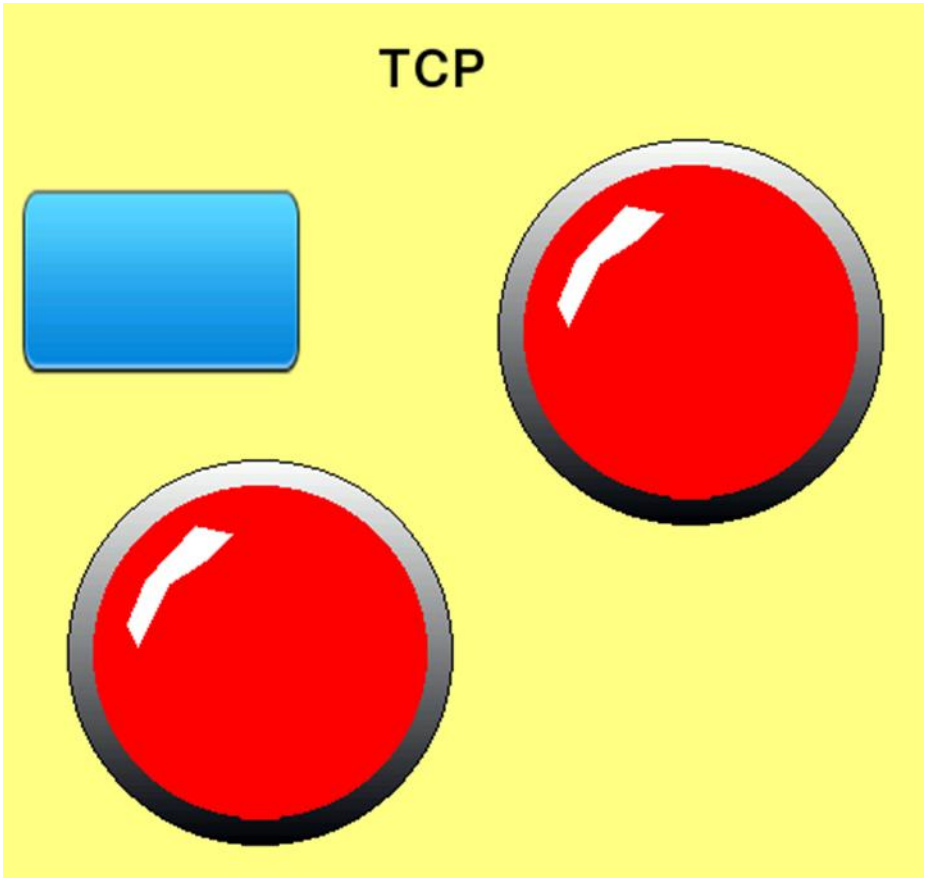


Т.е. идёт сплошная десятичная адресация. Каждый последующий регистр увеличивает адрес на 1.

%QX0.0.. %QX0.7 имеют десятичные адреса 0..7
%QX1.0.. %QX1.7 имеют десятичные адреса 8..15
%QX2.0.. %QX2.7 имеют десятичные адреса 16..23
и т.д.

При начале опроса контроллера панель будет отображать состояние регистров:



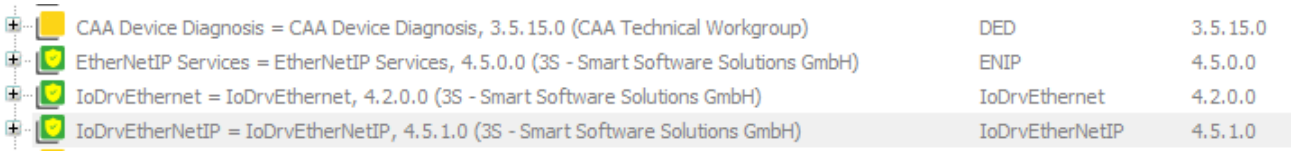


Связь по протоколу Ethernet/IP Scanner (Master)

Протокол **Ethernet/IP** (IP = Industrial Protocol) является одним из наиболее распространённых промышленных протоколов в мире и поддерживается сотнями производителей промышленного оборудования. **Ethernet/IP** (сокращённо **EIP**) использует модель **CIP** (Common Industrial Protocol) поверх стандартного интерфейса **Ethernet** и работает по схеме **Producer/Consumer**. На транспортном уровне используется протокол **UDP**.

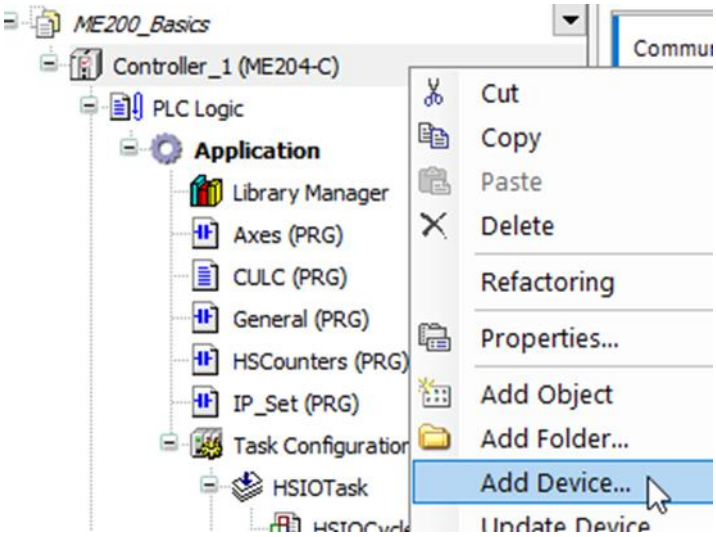
Контроллеры серии ME200 могут работать как в режиме **Ethernet/IP Scanner** (Master), так и в режиме **Ethernet/IP Adapter** (Slave). В данной главе рассматривается организация связи контроллера с Ведомыми устройствами в режиме Ethernet/IP Scanner (Master).

Для использования Ethernet/IP к проекту должны быть подключены следующие библиотеки: IoDrvEthernet, IoDrvEtherNetIP, EtherNetIP Services и DED.

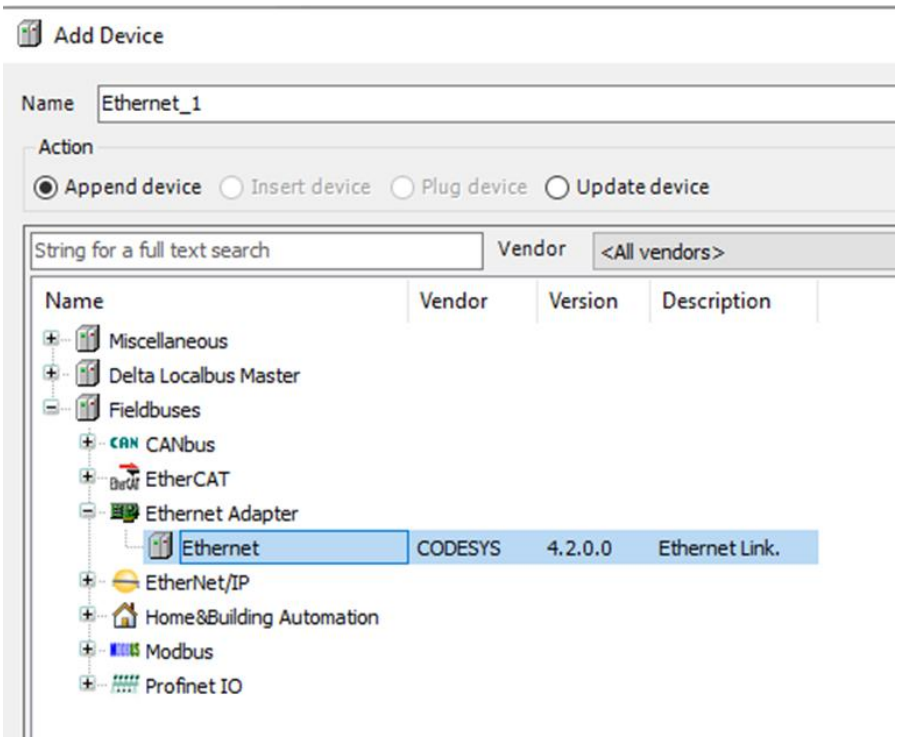


Настройка связи по протоколу Ethernet/IP начинается с добавления к проекту адаптера Ethernet и привязка его к определённому порту.

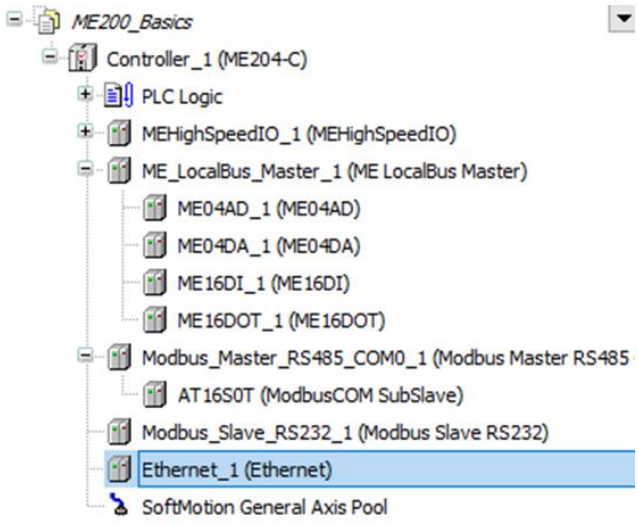
Щёлкните в древе правой кнопкой мышки на пункте **Device** (ME204-C) и в отрывшемся меню выберите пункт **Add Device**



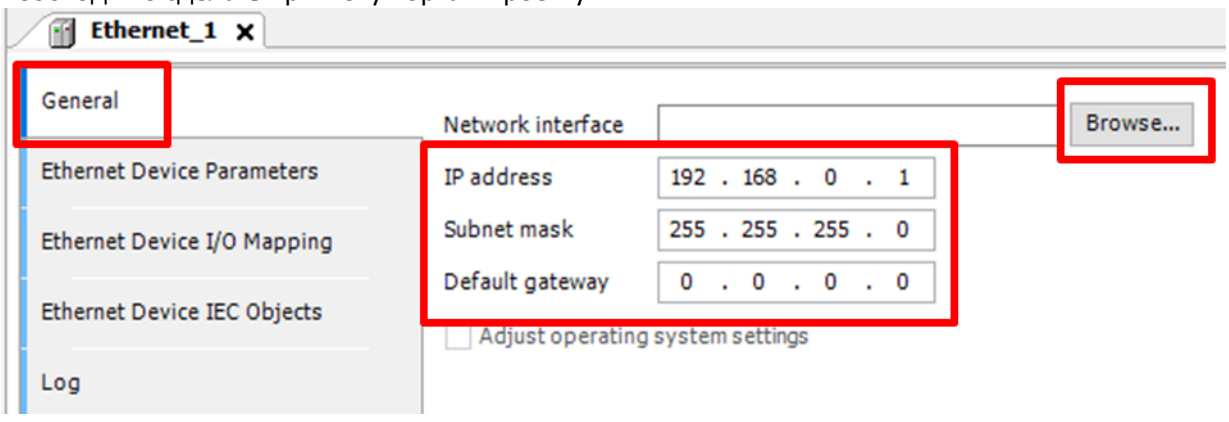
В открывшемся окне выберите **Fieldbuses – Ethernet**:



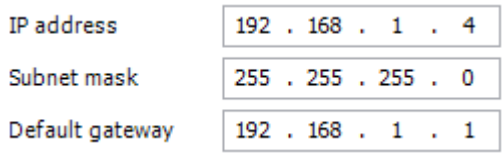
В древе проекта появится пункт **Ethernet**:



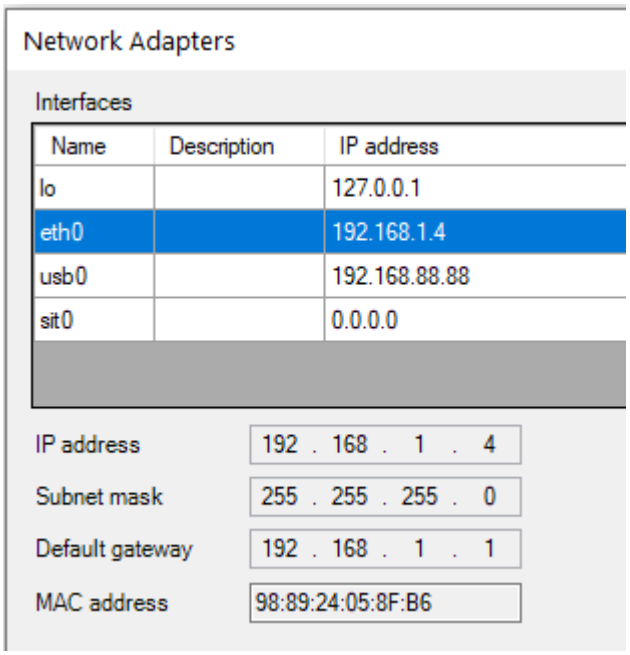
Щёлкните дважды левой кнопкой мышки на пункте **Ethernet**, и в появившейся вкладке в разделе **General** необходимо сделать привязку порта к проекту:



Задайте нужный IP адрес:



и нажмите кнопку **Browse…**. В появившемся окне выберите нужный интерфейс:



Нажмите **ОК**.

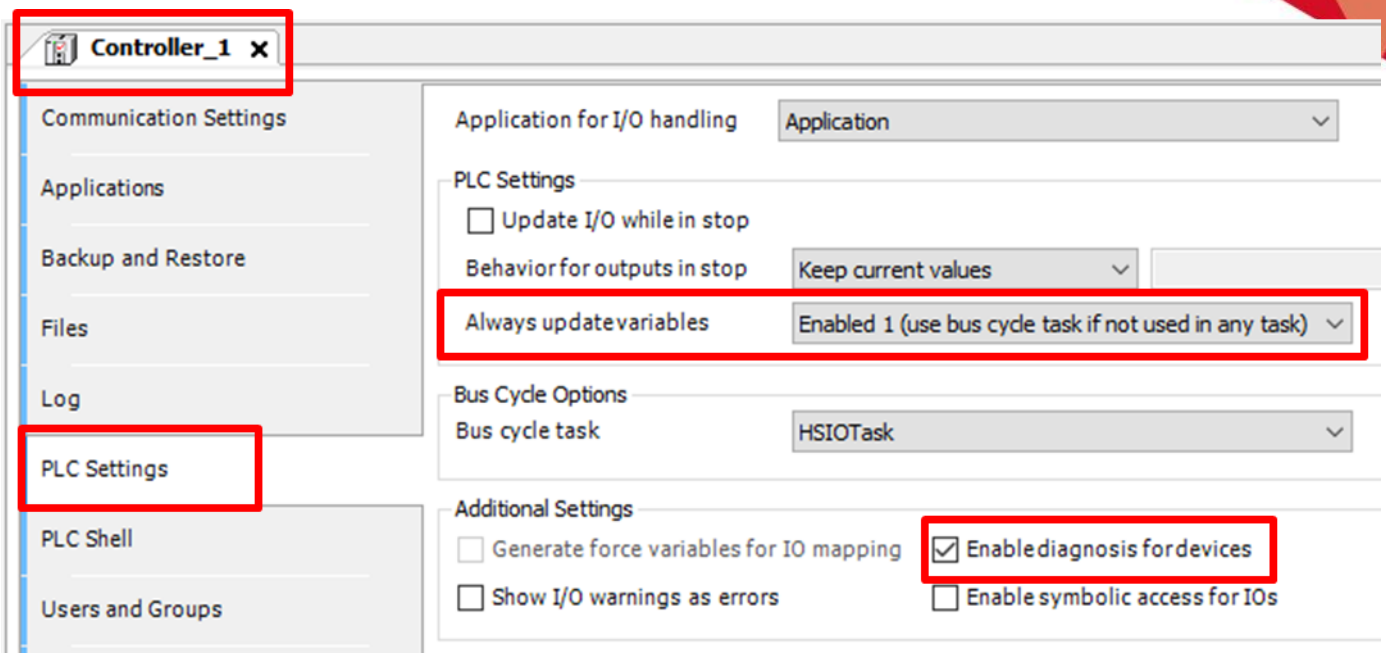
В поле Network Interface появится название интерфейса:



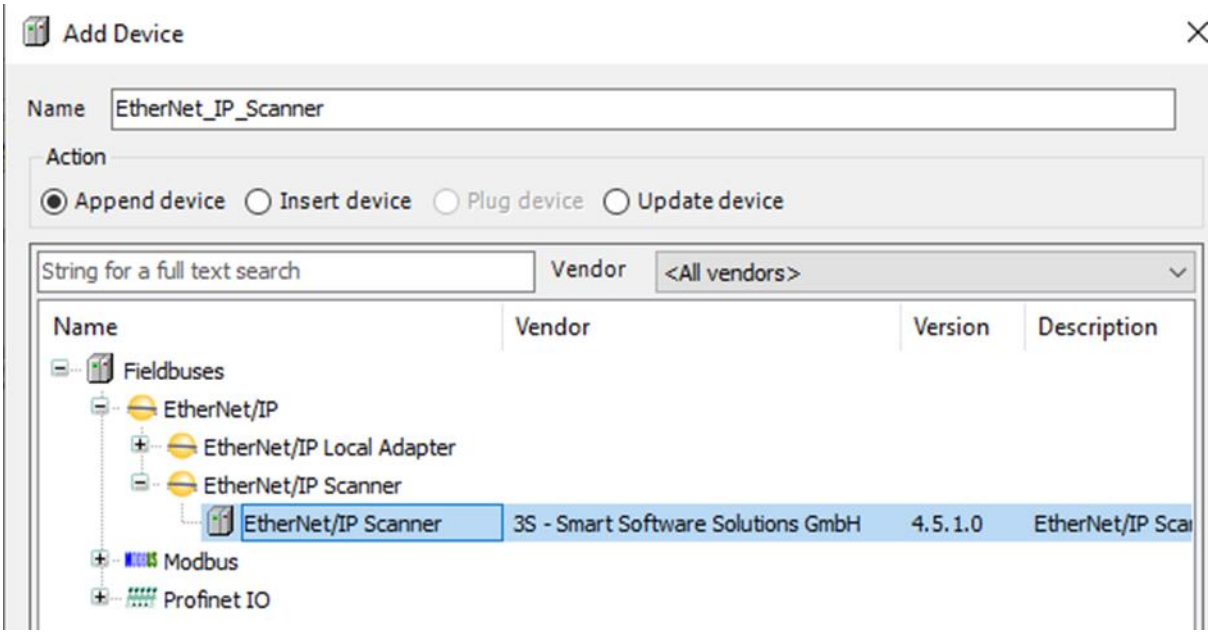
После чего Адаптер будет привязан к Ethernet порту с конкретным IP адресом.

Далее двойным щелчком левой кнопки мыши откройте вкладку **Device** и в пункте **PLC Settings** разрешите обновление переменных. Данный шаг позволяет видеть изменение переменных и регистров без написания в теле программы, т.е. в таблице Watch и в таблицах мэпинга.

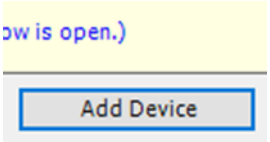
Также, для осуществления диагностики состояния подключенных к проекту устройств необходимо поставить флажок напротив пункта «Enable diagnosis for devices».



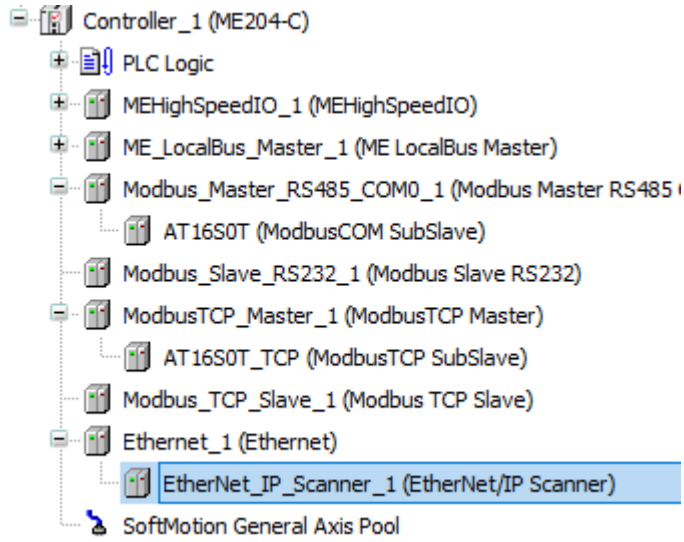
Щёлкните в древе правой кнопкой мышки на пункте **Ethernet** и в отрывшемся меню выберите пункт **Add Device**. Далее в открывшемся окне выберите пункт **Fieldbuses – Ethernet/IP Scanner**:



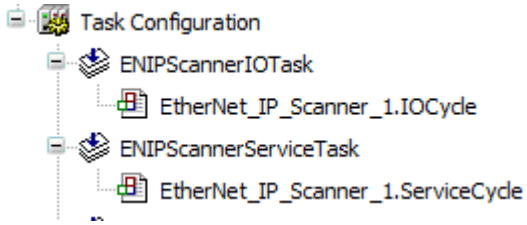
и нажмите **Add Device**:



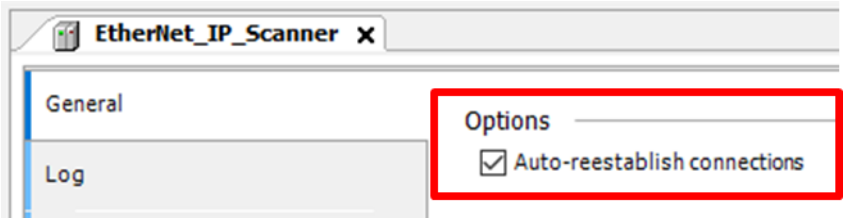
В древе проекта появится пункт **Ethernet/IP Scanner**:



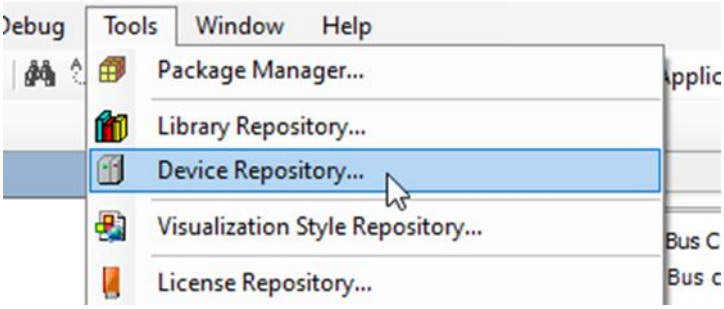
и автоматически будут созданы две задачи в Task Configuration, в которых будут обрабатываться все коммуникационные запросы протокола Ethernet/IP:



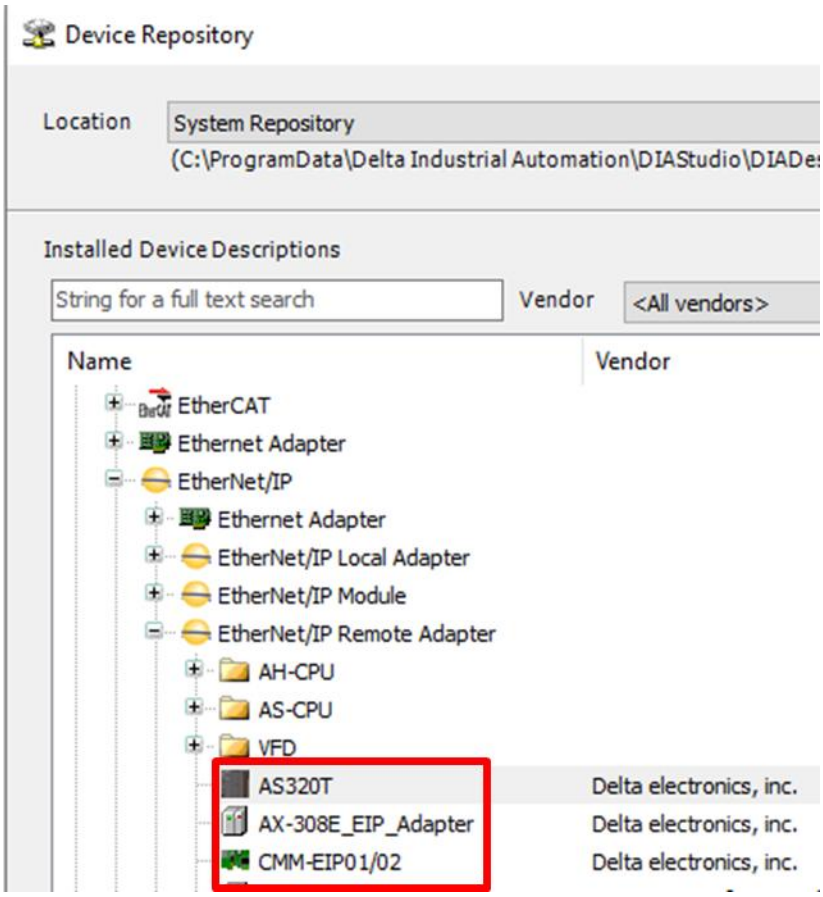
Щёлкните двойным щелчком левой кнопки мышки на пункте **Ethernet/IP Scanner** и в открывшейся вкладке выберите пункт General. Установите флажок автоматического восстановления соединения:



Далее можно перейти к добавлению Ведомых устройств (Ethernet/IP Adapter). Перед началом работы необходимо импортировать EDS файл устройства через пункт меню **Tools - Device Repository**:

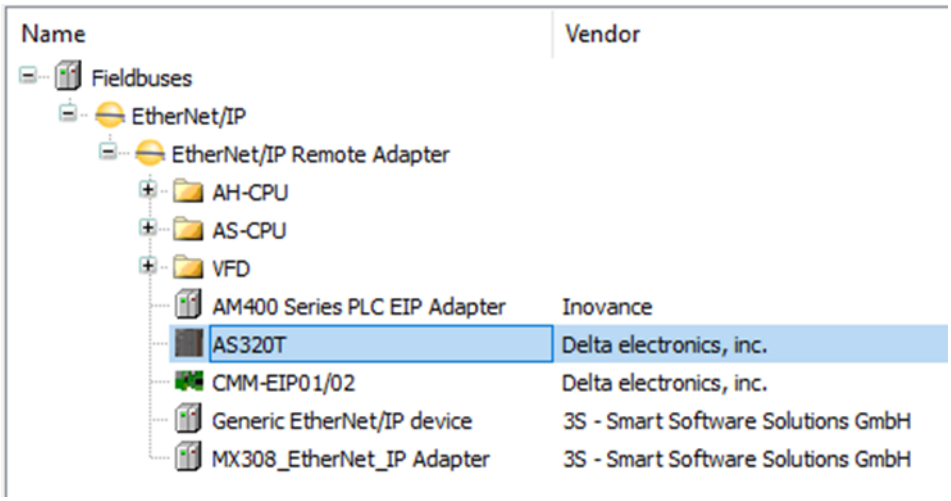


Ниже мы рассмотрим организацию связи с контроллером Delta AS320T и преобразователем частоты Delta MS300 посредством платы CMM-EIP02 по протоколу Ethernet/IP. Предполагается, что на этом этапе будут установлены соответствующие EDS файлы данных устройств:

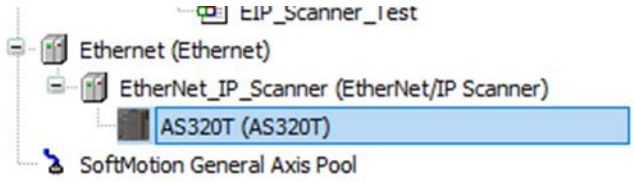

Device Repository
Location
System Repository
(C:\ProgramData\Delta Industrial Automation\DIAStudio\DIADe
Installed Device Descriptions
String for a full text search
Vendor
<All vendors>
Name
Vendor
EtherCAT
Ethernet Adapter
EtherNet/IP
Ethernet Adapter
EtherNet/IP Local Adapter
EtherNet/IP Module
EtherNet/IP Remote Adapter
AH-CPU
AS-CPU
VFD
AS320T
Delta electronics, inc.
AX-308E_EIP_Adapter
Delta electronics, inc.
CMM-EIP01/02
Delta electronics, inc.

Пример настройки связи с контроллером Delta AS320T по протоколу Ethernet/IP

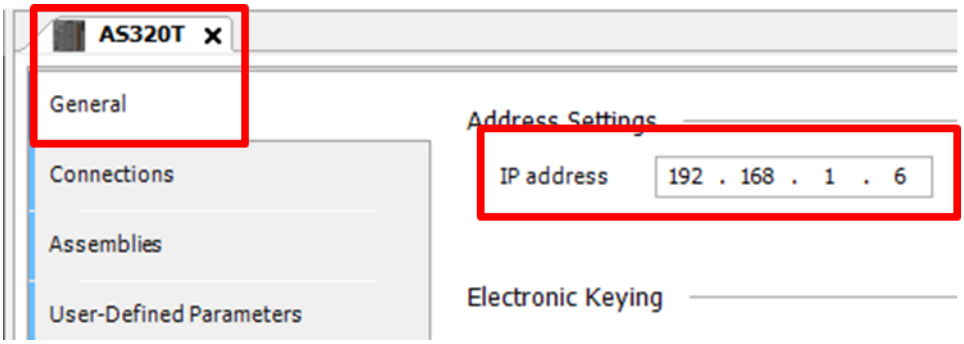
Для начала работы необходимо добавить Ведомое устройство в проект. Для этого щёлкните правой кнопкой мышки в древе проекта на пункте **Ethernet/IP Scanner** и в открывшемся меню выберите пункт **Add Device**. В открывшемся окне выберите из **Device Repository** устройство **AS320T**:



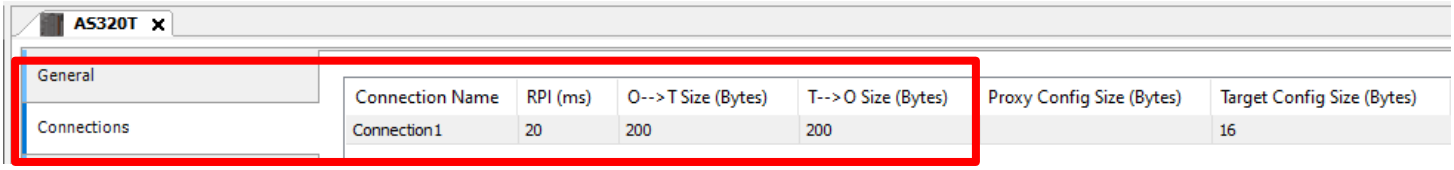
В древе проекта появится устройство:



Щёлкните дважды левой кнопкой мышки на пункте AS320T, и в открывшейся вкладке выберите пункт **General** и введите IP адрес ведомого устройства.



Далее в пункте **Connections** можно увидеть разметку данных от Ведомого (AS320T) и к Ведомому от Мастера (MX308), которая будет сделана системой в соответствие с EDS файлом на контроллер AS320T:



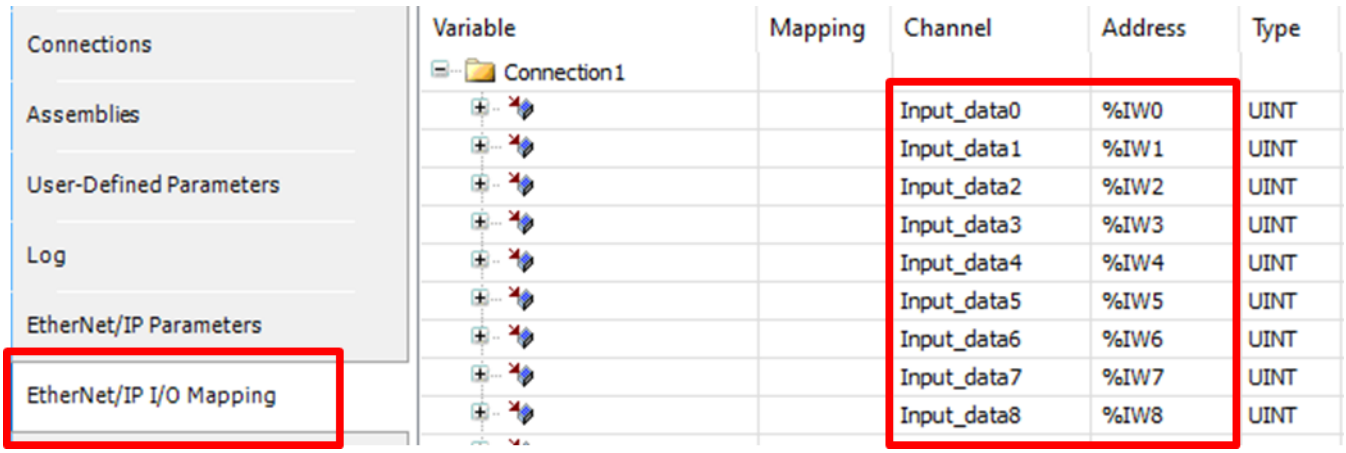
200 байтов от Ведомого (O → T) и 200 к Ведомому (T → O)

Адреса регистров в контроллере AS320T будет размечены также автоматически в соответствии с информацией в EDS файле:

Parameters	Value	Unit	Data Type	Minimum	Maximum	Default
Connection1						
Target Config data						
Conn1_Input(T->O) DeviceType	D		UINT	0	3	0
Conn1_Input(T->O) Reserved	200		UINT	0	500	200
Conn1_Input(T->O) DeviceIndex	1000		UDINT	0	29999	1000
Conn1_Output(O->T) DeviceType	D		UINT	0	3	0
Conn1_Output(O->T) Reserved	200		UINT	0	500	200
Conn1_Output(O->T) DeviceIndex	0		UDINT	0	29999	0

В нашем примере Ведомый контроллер AS320T будет принимать данные в регистры D0 – D99, а отправлять Мастеру из регистров D1000 – D1099. Количество и номера регистров можно поменять.

Далее в пункте **Ethernet/IP I/O Mapping** можно увидеть выделенные системой регистры данных от Ведомого (AS320T) %IW0 - %IW99, и к Ведомому от Мастера (MX308) %QW0 - %QW99, которая будет автоматически сделана системой:



Output_data0	%QW0	UINT
Output_data1	%QW1	UINT
Output_data2	%QW2	UINT
Output_data3	%QW3	UINT
Output_data4	%QW4	UINT
Output_data5	%QW5	UINT
Output_data6	%QW6	UINT
Output_data7	%QW7	UINT
Output_data8	%QW8	UINT
Output_data9	%QW9	UINT
Output_data10	%QW10	UINT
Output_data11	%QW11	UINT

Система по умолчанию назначает регистрам Мастер-контроллера MX308 тип данных UINT, т.е. безнаковый. Но так как в Ведомом контроллере AS320T регистры данных D0 – D29999 являются целочисленными 16 бит со знаком, поэтому в Мастере регистры желательно объявить переменными типа INT:

7	VAR	EIP_RST_TRG		BOOL
8	VAR	_D1000	%IW0	INT

Если загрузить проекты и запустить в онлайн режим оба контроллера, то передача данных будет выглядеть следующим образом:

От Ведомого к Мастеру:

В программе Ведомого контроллера AS320T задаются данные в регистрах D1000 и D1099,

Device Name	Status	Data Type	Value (16bits)
D0			0
D99			0
D1000			789
D1099			1234

которые автоматически передаются Мастеру в регистры %IW0 и %IW99:

Watch 1

Expression	Type	Value
%IW0	WORD	789
%IW99	WORD	1234
%QW0	WORD	0
%QW99	WORD	0
AS320T.eState	ADAPTERSTATE	RUNNING

Состояние связи с Ведомым можно проверить через свойство узла **eState**:

AS320T.eState	MX308_E.Application	ADAPTERSTATE	RUNNING

От Мастера к Ведомому:

В программе Мастера задаются данные в регистрах %QW0 и %QW99

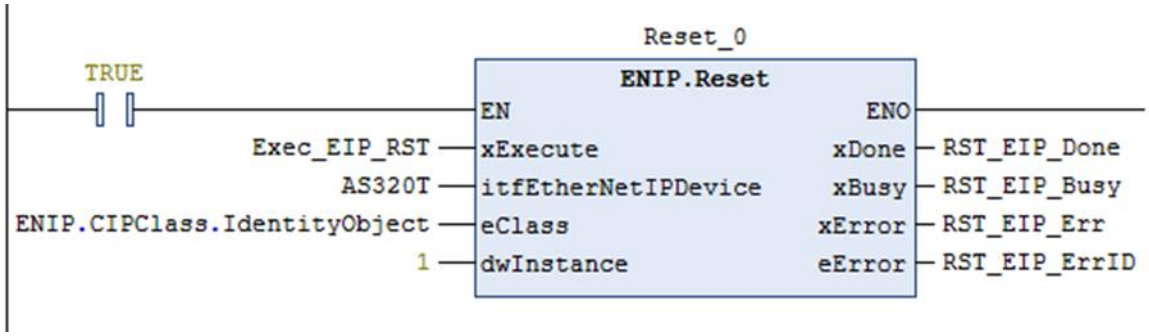
Watch 1

Expression	Type	Value
%IW0	WORD	0
%IW99	WORD	0
%QW0	WORD	45
%QW99	WORD	555
AS320T.eState	ADAPTERSTATE	RUNNING

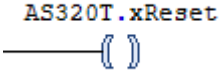
которые автоматически попадают в регистры D0 и D99 Ведомого:

Device Name	Status	Data Type	Value (16bits)
D0			45
D99			555
D1000			0
D1099			0

Для перезагрузки узла можно использовать команду **ENIP.Reset**:



Или свойство узла **xReset**:



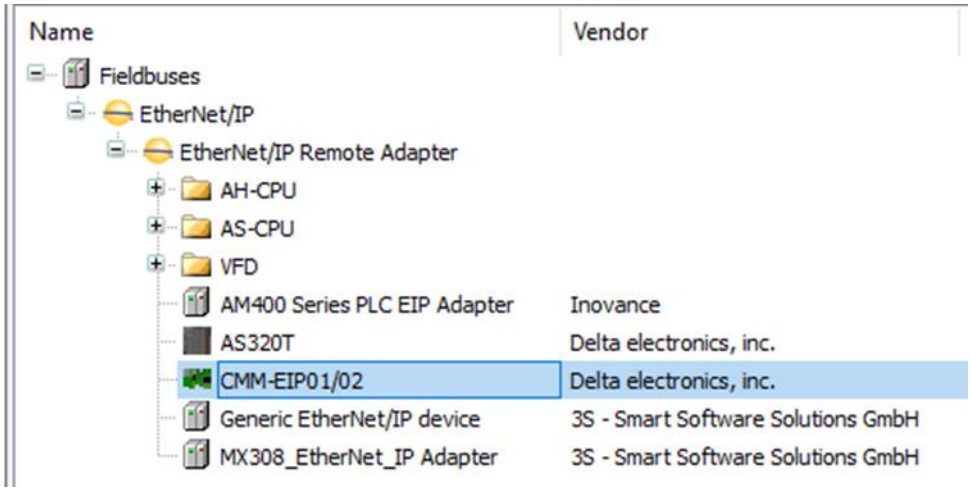
Пример настройки связи с ПЧ Delta MS300 по протоколу Ethernet/IP

Преобразователи частоты типа Delta MS300 осуществляют связь по протоколу Ethernet/IP посредством коммуникационной платы расширения CMM-EIP02. Перед началом работы необходимо убедиться, что версия встроенного ПО (firmware) платы не ниже 2.05.06, а самого преобразователя частоты не ниже 2.02.

Также, необходимо убедиться, что код продукта в EDS файле на данную плату стоит 17157:

```
[Device]
        VendCode = 799;
        VendName = "Delta electronics, inc.";
        ProdType = 12;
        ProdTypeStr = "Communications Adapter";
        ProdCode = 17157;
        MajRev = 1;
        MinRev = 1;
        ProdName = "CMM-EIP01/02";
        Catalog = "CMM-EIP01/02";
        Icon = "CMMS-EIP01.ico";
        IconContents =
```

Для начала работы необходимо добавить Ведомое устройство в проект. Для этого щёлкните правой кнопкой мышки в древе проекта на пункте **Ethernet/IP Scanner** и в открывшемся меню выберите пункт **Add Device**. В открывшемся окне выберите из **Device Repository** устройство **CMM-EIP01/02**:



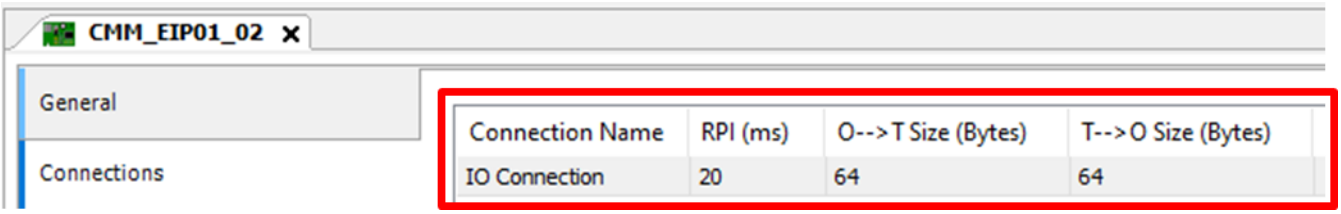
В древе проекта появится устройство:



Щёлкните дважды левой кнопкой мышки на пункте CMM-EIP01/02, и в открывшейся вкладке выберите пункт **General** и введите IP адрес ведомого устройства.



Далее в пункте **Connections** можно увидеть разметку данных от Ведомого (CMM-EIP01/02) и к Ведомому от Мастера (MX308), которая будет сделана системой в соответствие с EDS файлом на плату CMM-EIP01/02:



64 байта от Ведомого (O → T) и 64 к Ведомому (T → O)

Соответствие регистрам привода приводится в таблице ниже:

Данные от привода в сторону контроллера:

Configuration Data

Raw data values Show Parameter Groups

Parameters	Value	Unit	Data Type
IN_Addr_1	[2100] Warn / Error		UINT
IN_Addr_2	[2101] Drive status 1		UINT
IN_Addr_3	[2102] Frequency command 1		UINT
IN_Addr_4	[2103] Output frequency		UINT
IN_Addr_5	[2104] Output current		UINT
IN_Addr_6	[2105] DC bus voltage		UINT
IN_Addr_7	[2106] Output voltage		UINT
IN_Addr_8	[2107] Multi-speed step		UINT
IN_Addr_9	[2108] Max output torque		UINT
IN_Addr_10	[2109] Trigger count		UINT
IN_Addr_11	[210A] PF angle		UINT
IN_Addr_12	[210B] Output torque		UINT
IN_Addr_13	[210C] Real speed(rpm)		UINT
IN_Addr_14	[210D] PG feedback count		UINT
IN_Addr_15	[210E] PG2 pulse count		UINT
IN_Addr_16	[210F] Output Power		UINT
IN_Addr_17	[6100] Status word 2		UINT
IN_Addr_18	[6101] Real mode 2		UINT
IN_Addr_19	[6102] Output Frequency 2		UINT

Данные от контроллера к приводу:

Configuration Data

Raw data values ☑ Show Parameter Groups

Parameters	Value	Unit	Data Type
OUT_Addr_1	[2000] Operation Command 1		UINT
OUT_Addr_2	[2001] Frequency command 1		UINT
OUT_Addr_3	[2002] External Command 1		UINT
OUT_Addr_4	Reserved		UINT
OUT_Addr_5	[6000] Operation Command2		UINT
OUT_Addr_6	[6001] Control mode 2		UINT
OUT_Addr_7	[6002] Frequency command 2		UINT
OUT_Addr_8	[6003] Troque limit 2		UINT
OUT_Addr_9	[6004] Position Command 2, L W		UINT
OUT_Addr_10	[6005] Position Command 2, H W		UINT
OUT_Addr_11	[6006] Torque command 2		UINT
OUT_Addr_12	[6007] Frequency limit 2		UINT

Далее в пункте **Ethernet/IP I/O Mapping** можно увидеть выделенные системой регистры данных от Ведомого (MS300) %IW100- %IW131, и к Ведомому от Мастера (ME204-C) %QW100 - %QW131, которая будет автоматически сделана системой:

IO Connection					
		Error Code	%IW100	UINT	
STATUS_WORD		Reserved2101	%IW101	UINT	
FREQ_SET		Frequence Command	%IW102	UINT	Hz
OUT_FREQ		Output Frequence	%IW103	UINT	Hz
		Output Current	%IW104	UINT	A
DC_BUS		DC Bus Voltage	%IW105	UINT	
		Output Voltage	%IW106	UINT	V
		Speed	%IW107	UINT	

		Control	%QW100	UINT	
		Frequence Command	%QW101	UINT	Hz
		Others	%QW102	UINT	
		OUT_Value	%QW103	UINT	
CTRL_WORD		OUT_Value	%QW104	UINT	
		OUT_Value	%QW105	UINT	
FREQ_CMD		OUT_Value	%QW106	UINT	
		OUT_Value	%QW107	UINT	
		OUT_Value	%QW108	UINT	

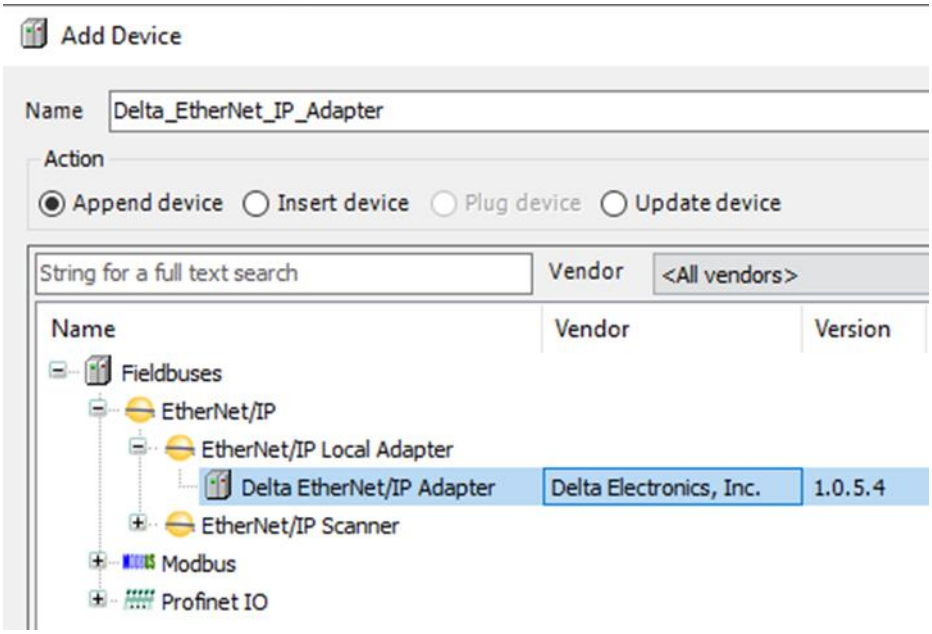
Далее регистрам можно присвоить переменные и использовать в программе контроллера для управления приводом:

CMM_EIP01_02.eState	ADAPTERSTATE	RUNNING
CTRL_WORD	UINT	129
FREQ_CMD	UINT	2500
FREQ_SET	UINT	2500
OUT_FREQ	UINT	1154
DC_BUS	UINT	3060
STATUS_WORD	UINT	13571

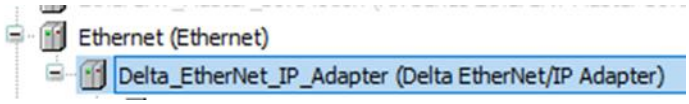
Пример настройки связи с другим контроллером CODESYS по протоколу Ethernet/IP

В среде программирования CODESYS есть специальная процедура для настройки связи между двумя контроллерами CODESYS по протоколу Ethernet/IP. В нашем примере в качестве Сканера (Мастера) будет выступать контроллер ME204-C, а в качестве Адаптера (Ведомого) контроллер Delta AX-308E.

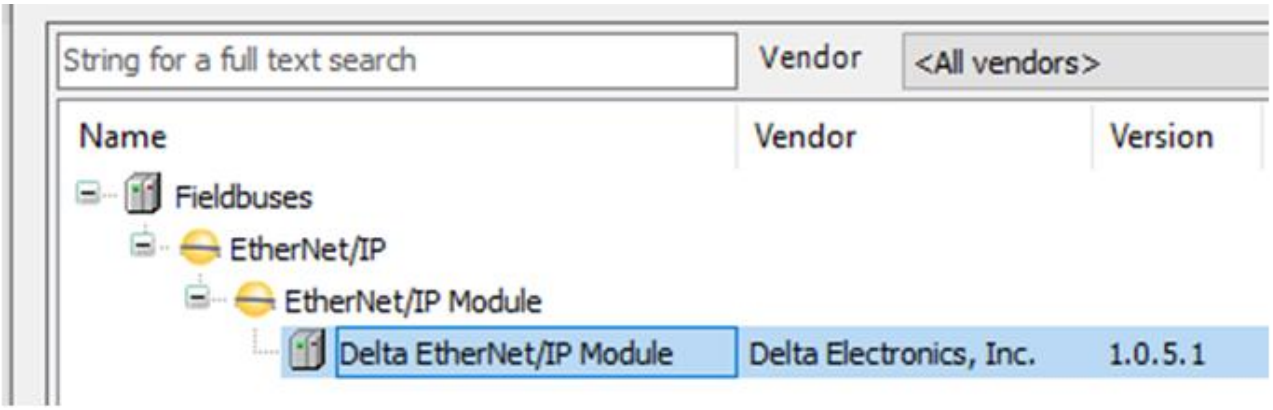
Создайте проект для контроллера Delta AX-308E. При необходимости воспользуйтесь Руководством на данный контроллер.
Добавьте в проект адаптер Ethernet. Выделите мышкой этот пункт в древе проекта и нажмите правую кнопку мышки. В появившемся меню выберите пункт **Add Device**. В открывшемся окне выберите **Delta Ethernet/IP Adapter**:



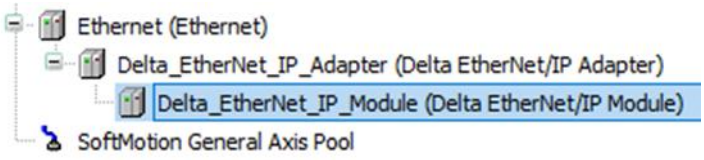
В древе проекта появится пункт:



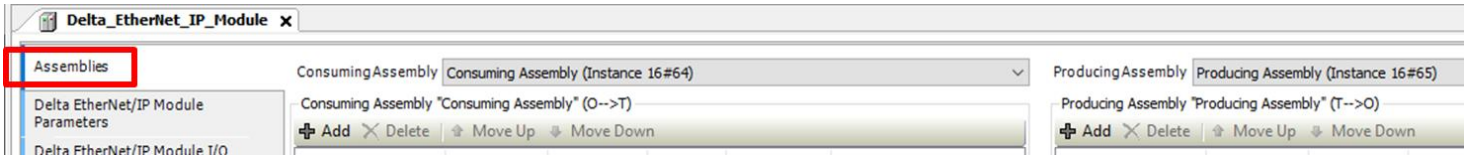
Встаньте мышкой на этот пункт и щёлкните правой кнопкой. В появившемся меню выберите пункт **Add Device**. В открывшемся окне выберите **Delta Ethernet/IP Module**:



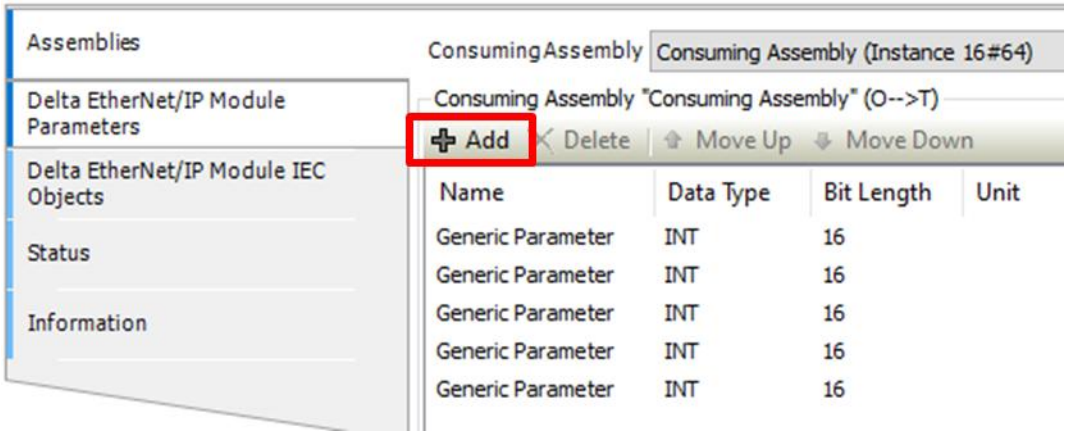
В древе проекта появится пункт:



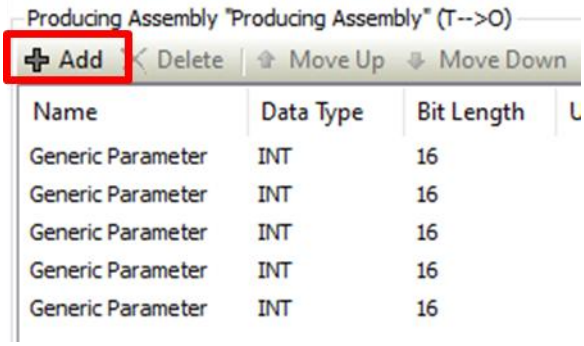
Щёлкните на этом пункте дважды левой кнопкой мышки и в отрывшейся вкладке выберите раздел **Assemblies**:



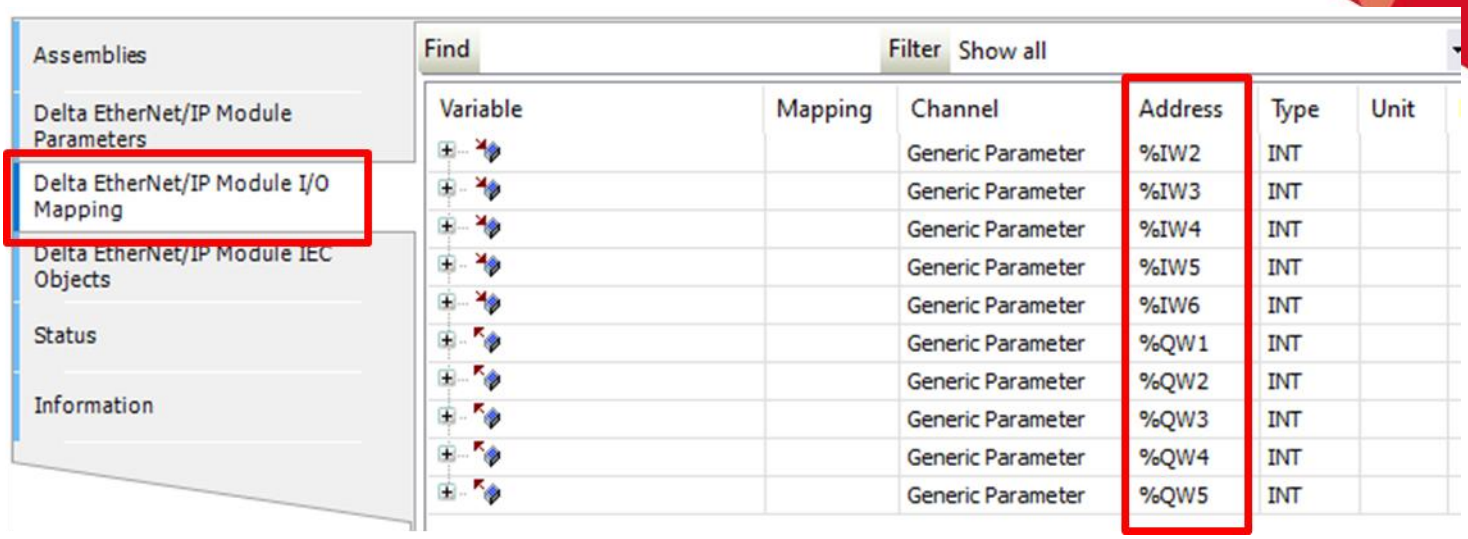
В данном разделе создаются пакеты для отправки и приёма от Мастера. Для добавления регистров в пакет на приём от Мастера нажимайте кнопку **Add** в поле **Consuming Assembly**:



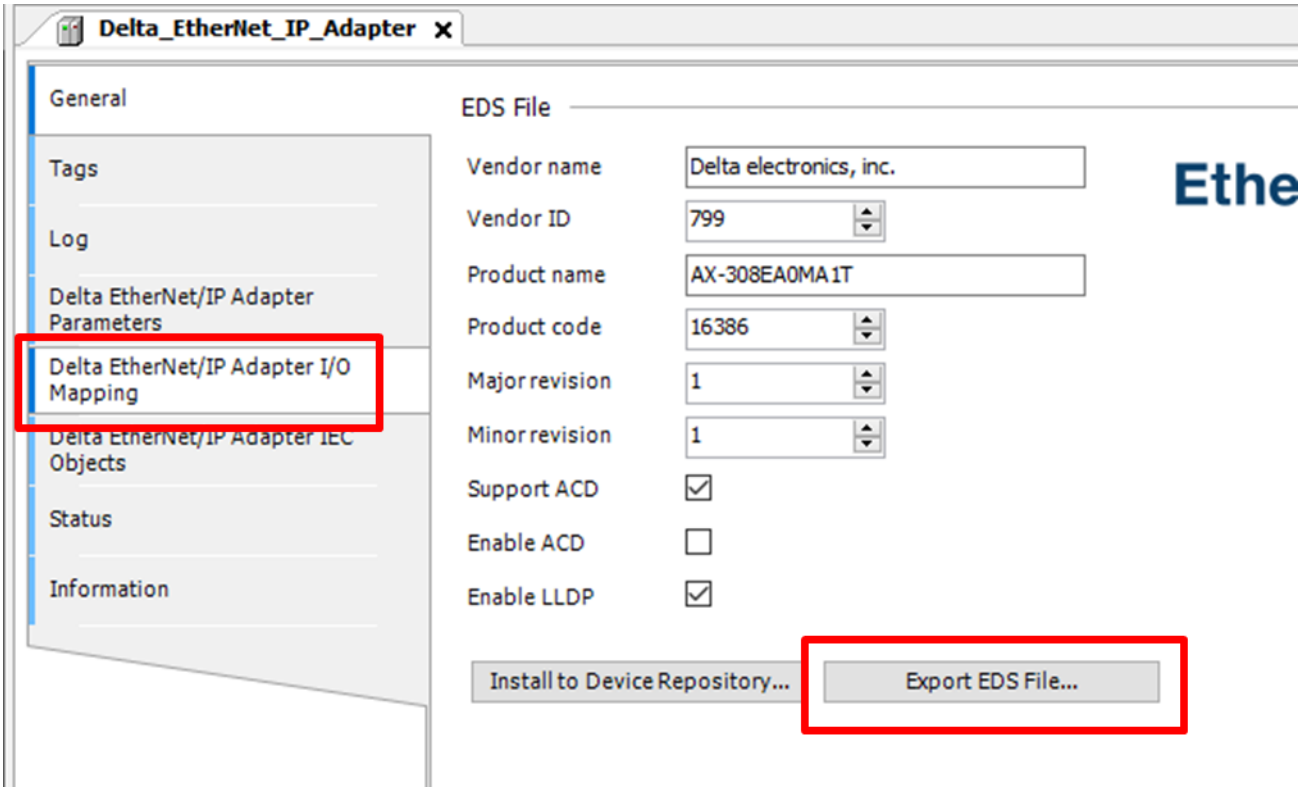
Для добавления регистров в пакет на отправку Мастеру нажимайте кнопку **Add** в поле **Producing Assembly**:



Общий список регистров можно увидеть в разделе **Delta Ethernet/IP Module I/O Mapping**. Здесь же можно присвоить регистрам теги. Регистры %IW2-%IW6 будут содержать данные от Мастера (Consumed Tags), а из регистров %QW1-%QW5 данные будут передаваться Мастеру (Produced Tags).



Для экспорта созданной таблицы данных для обмена с Мастером необходимо двойным щелчком открыть вкладку **Delta Ethernet/IP Adapter** и выбрать раздел **Delta Ethernet/IP Adapter I/O Mapping**:



Нажмите кнопку **Export EDS File** и получите XML файл с тегами. Лучше изменить **ProdName** со стандартного на своё название. Это поможет избежать путаницы со стандартным адаптером при выборе устройства из репозитария.

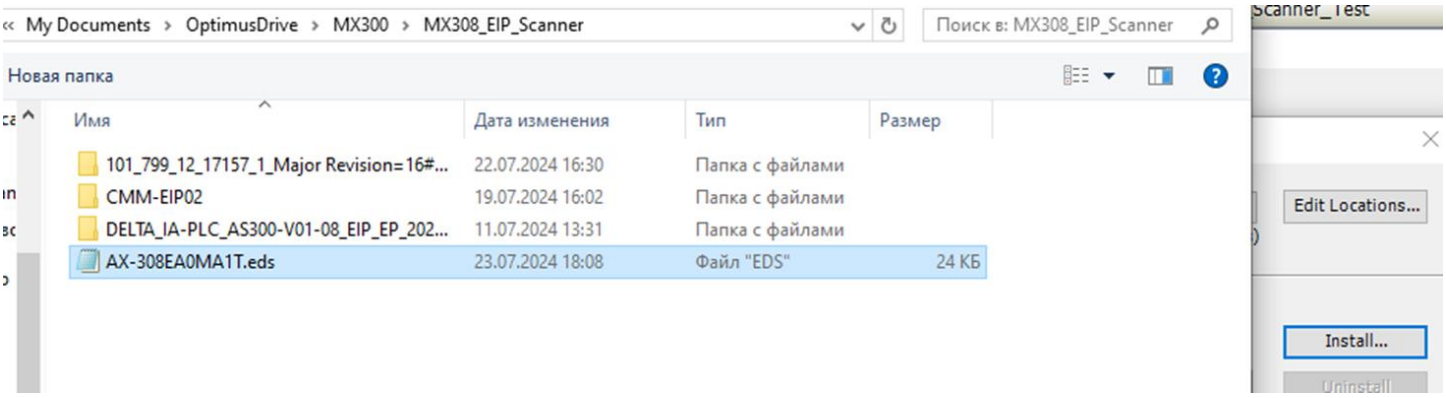
Для изменения поля откройте файл в Блокноте (Notepad) и измените название поле **ProdName**

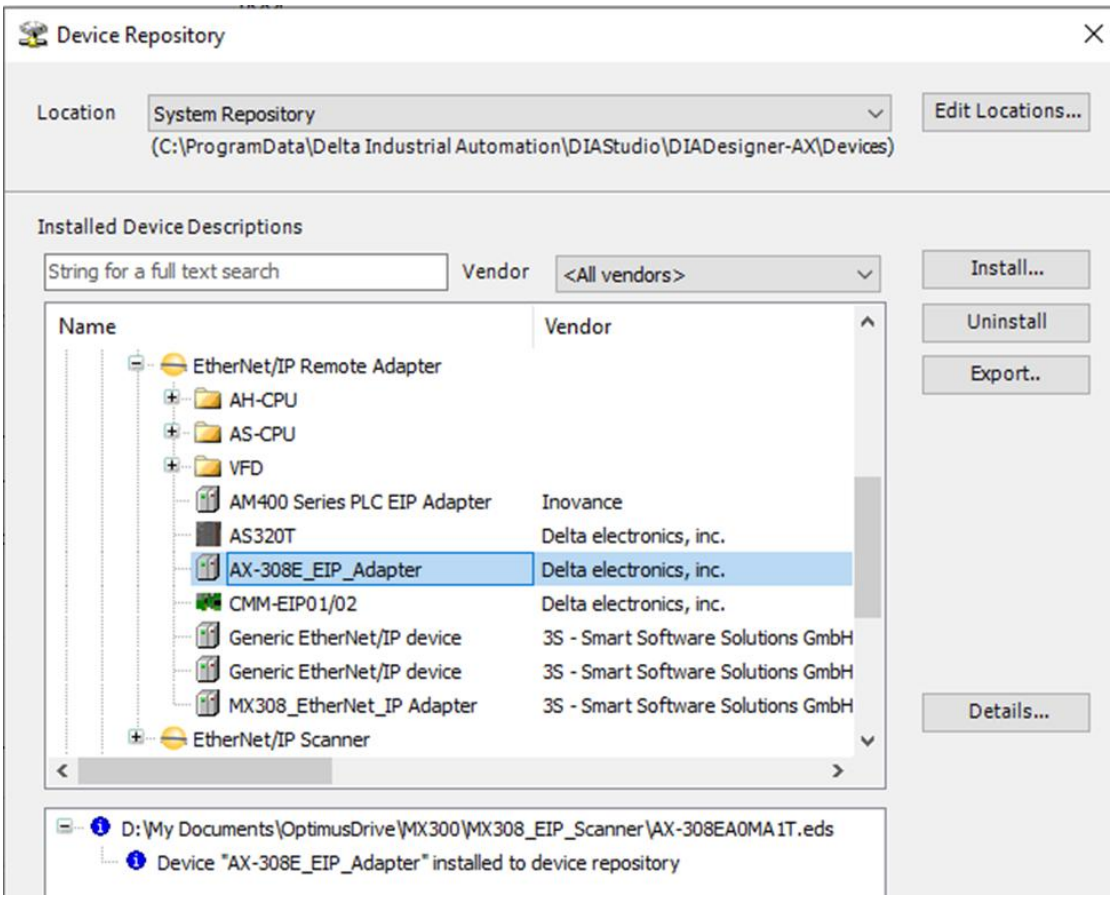
```
[Device]
        VendCode = 799;
        VendName = "Delta electronics, inc.";
        ProdType = 12;
        ProdTypeStr = "Communications Adapter";
        ProdCode = 16386;
        MajRev = 1;
        MinRev = 1;
        ProdName = "AX-308EA0MA1T";
```

например на такое:

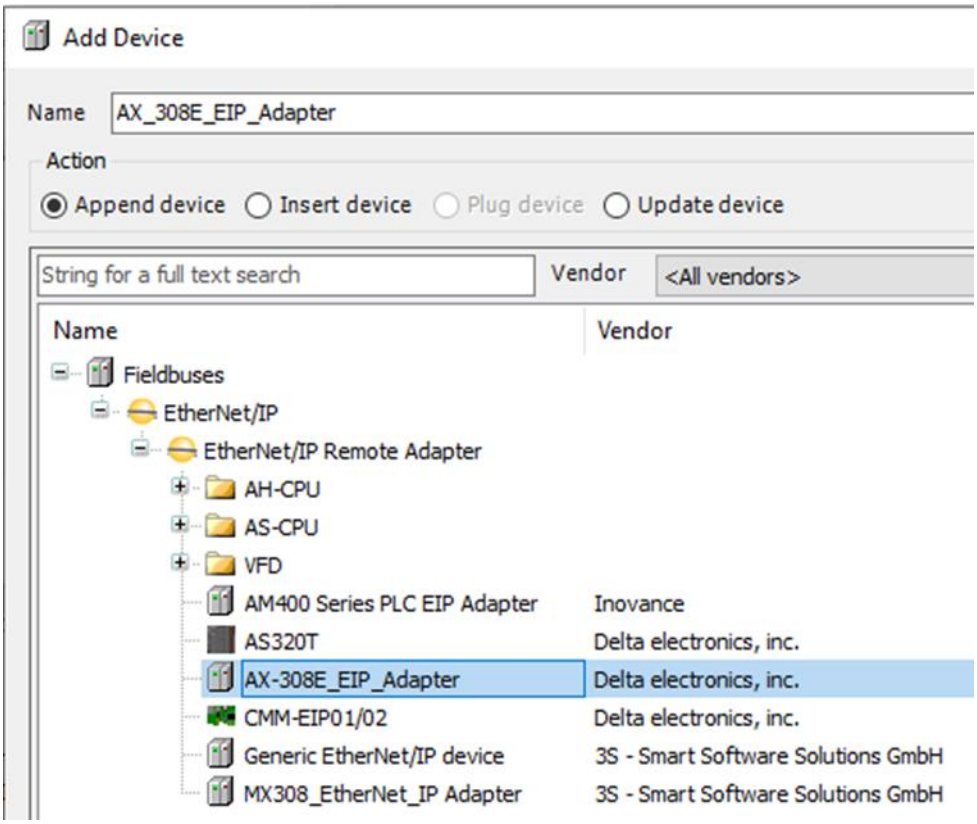
```
[Device]
        VendCode = 799;
        VendName = "Delta electronics, inc.";
        ProdType = 12;
        ProdTypeStr = "Communications Adapter";
        ProdCode = 16386;
        MajRev = 1;
        MinRev = 1;
        ProdName = "AX-308E_EIP_Adapter";
```

Далее файл необходимо импортировать в репозитарий устройств Мастера (ME200). Т.е. там появится устройство типа Adapter Ethernet/IP. Данная процедура, как и создание проекта для контроллеров ME200, описаны в соответствующих Главах настоящего Руководства.

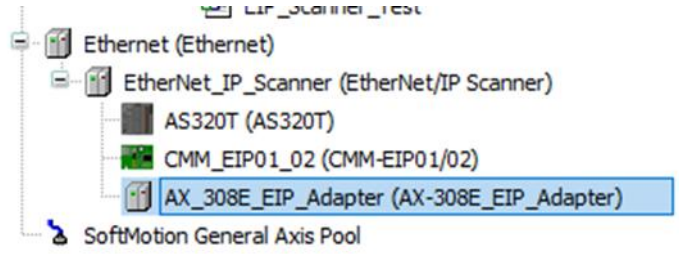




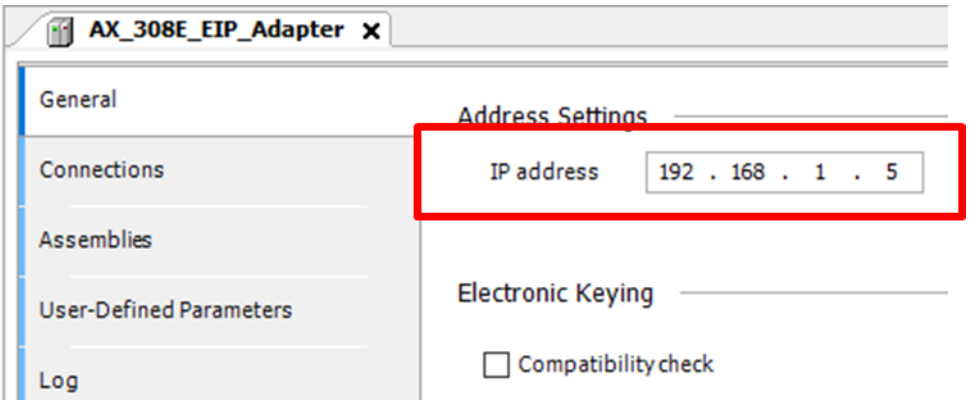
Добавьте адаптер в проект Мастера:



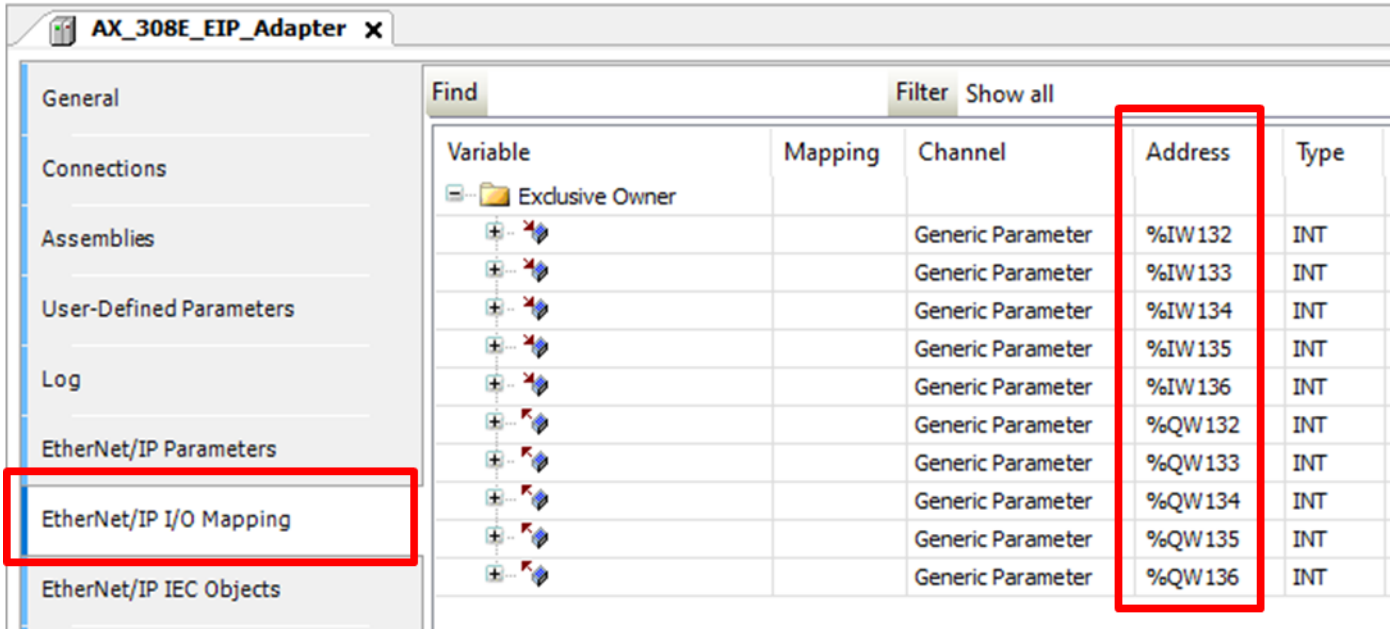
В древе проекта появится устройство:



Выберите это устройство и щёлкните дважды левой кнопкой мышки. Откроется вкладка с настройками. Выберите пункт **General** и выставьте **IP адрес EIP Adapter** (AX308-E):



Система автоматически раздаст адреса в соответствии с импортированным XML файлом, которые можно посмотреть в пункте **EtherNet/IP I/O Mapping**:



Регистры %IW132-%IW136 будут содержать данные от Ведомого (Produced Tags), а из регистров %QW132-%QW136 данные будут передаваться Ведомому (Consumed Tags).
(Produced Tags и Consumed Tags с точки зрения Ведомого).

Если в Мастере (контроллер ME204-C) в регистре %QW132 задать число 45:

%IW132	WORD	0
%QW132	WORD	45
AX_308E_EIP_Adapter.eState	ADAPTERSTATE	RUNNING

То в регистре %IW2 Ведомого (контроллер Delta AX-308E) появится число 45:

Watch 1

Expression	Application	Type	Value
%IW2	Device.Application	WORD	45
%QW1	Device.Application	WORD	0
Delta_EtherNet_IP_Module.eState	Device.Application	MODULESTATE	RUNNING

Состояние связи можно контролировать при помощи элемента структуры Adapter_Name.**eState**

AX_308E_EIP_Adapter.eState

Delta_EtherNet_IP_Module.eState

Если в регистре Ведомого (Delta AX-308E) %QW1 задать число 555:

Watch 1

Expression	Application	Type	Value
%IW2	Device.Application	WORD	0
%QW1	Device.Application	WORD	555
Delta_EtherNet_IP_Module.eState	Device.Application	MODULESTATE	RUNNING

в регистре Мастера (ME204-C) %IW132 появится число 555:

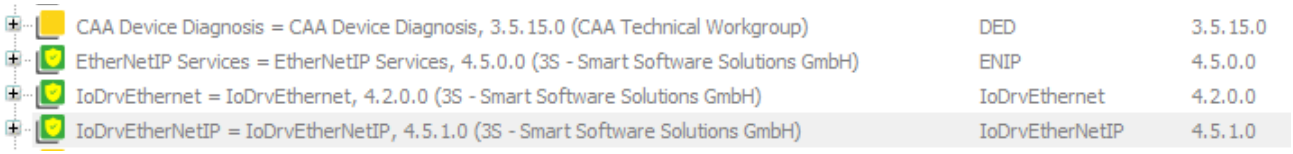
%IW132	WORD	555
%QW132	WORD	0
AX_308E_EIP_Adapter.eState	ADAPTERSTATE	RUNNING

Связь по протоколу Ethernet/IP Adapter (Slave)

Протокол **Ethernet/IP** (IP = Industrial Protocol) является одним из наиболее распространённых промышленных протоколов в мире и поддерживается сотнями производителей промышленного оборудования. **Ethernet/IP** (сокращённо **EIP**) использует модель **CIP** (Common Industrial Protocol) поверх стандартного интерфейса **Ethernet** и работает по схеме **Producer/Consumer**. На транспортном уровне используется протокол **UDP**.

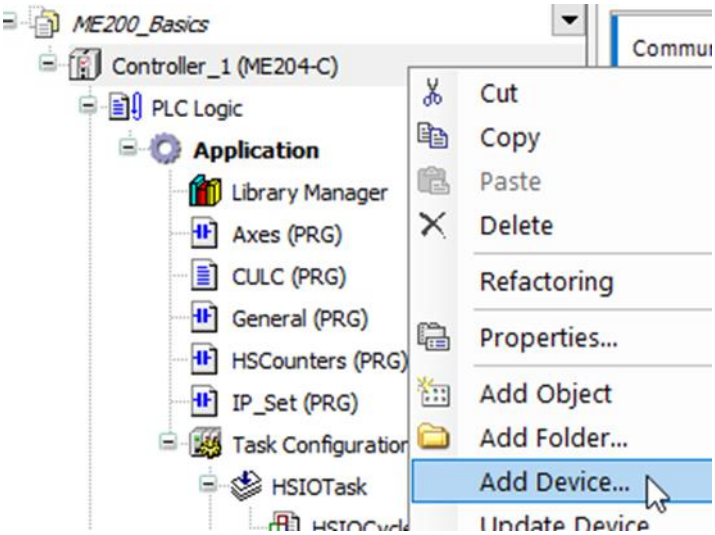
Контроллеры серии ME200 могут работать как в режиме **Ethernet/IP Scanner** (Master), так и в режиме **Ethernet/IP Adapter** (Slave). В данной главе рассматривается организация связи контроллера ME200 с Мастером в режиме Ethernet/IP Adapter (Slave).

Для использования Ethernet/IP к проекту должны быть подключены следующие библиотеки: IoDrvEthernet, IoDrvEtherNetIP, EtherNetIP Services и DED.

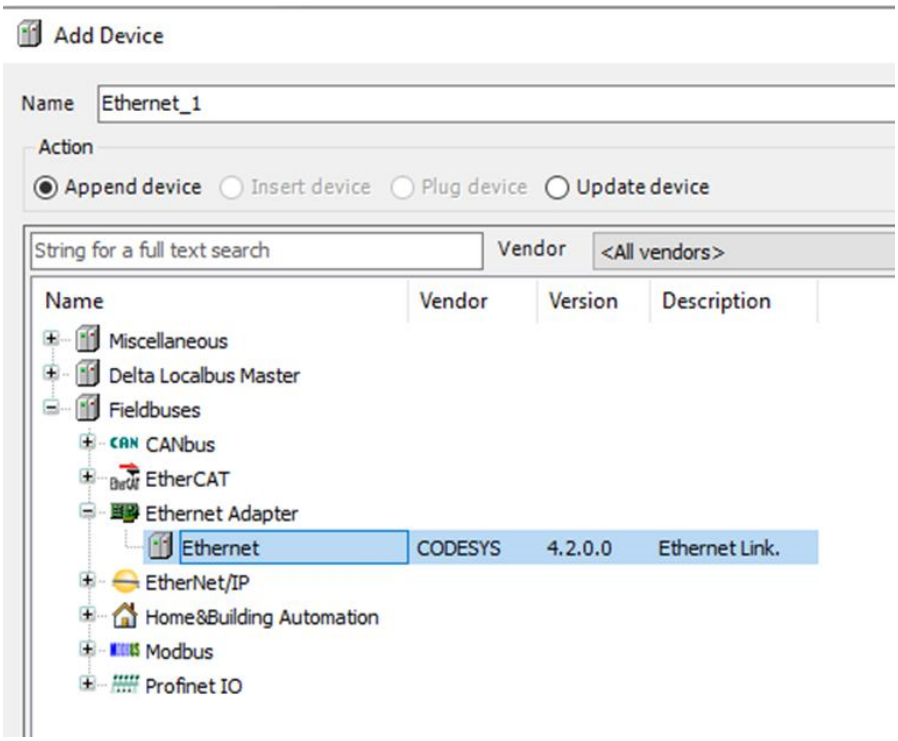


Настройка связи по протоколу Ethernet/IP начинается с добавления к проекту адаптера Ethernet и привязка его к определённому порту.

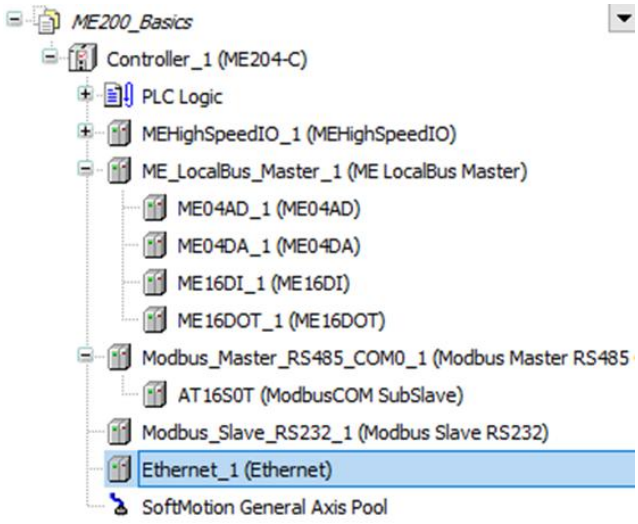
Щёлкните в древе правой кнопкой мышки на пункте **Device** (ME204-C) и в отрывшемся меню выберите пункт **Add Device**



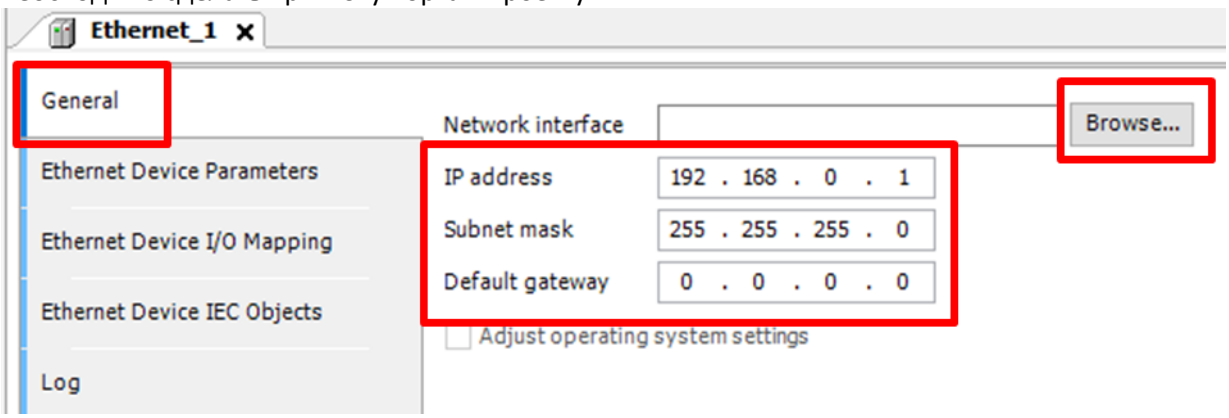
В открывшемся окне выберите **Fieldbuses – Ethernet**:



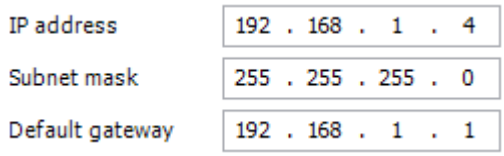
В древе проекта появится пункт **Ethernet**:



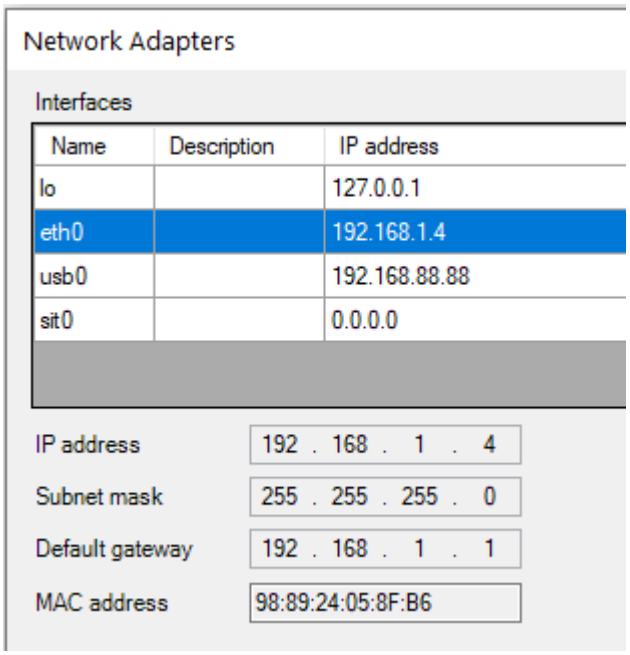
Щёлкните дважды левой кнопкой мышки на пункте **Ethernet**, и в появившейся вкладке в разделе **General** необходимо сделать привязку порта к проекту:



Задайте нужный IP адрес:



и нажмите кнопку **Browse…**. В появившемся окне выберите нужный интерфейс:



Нажмите **ОК**.

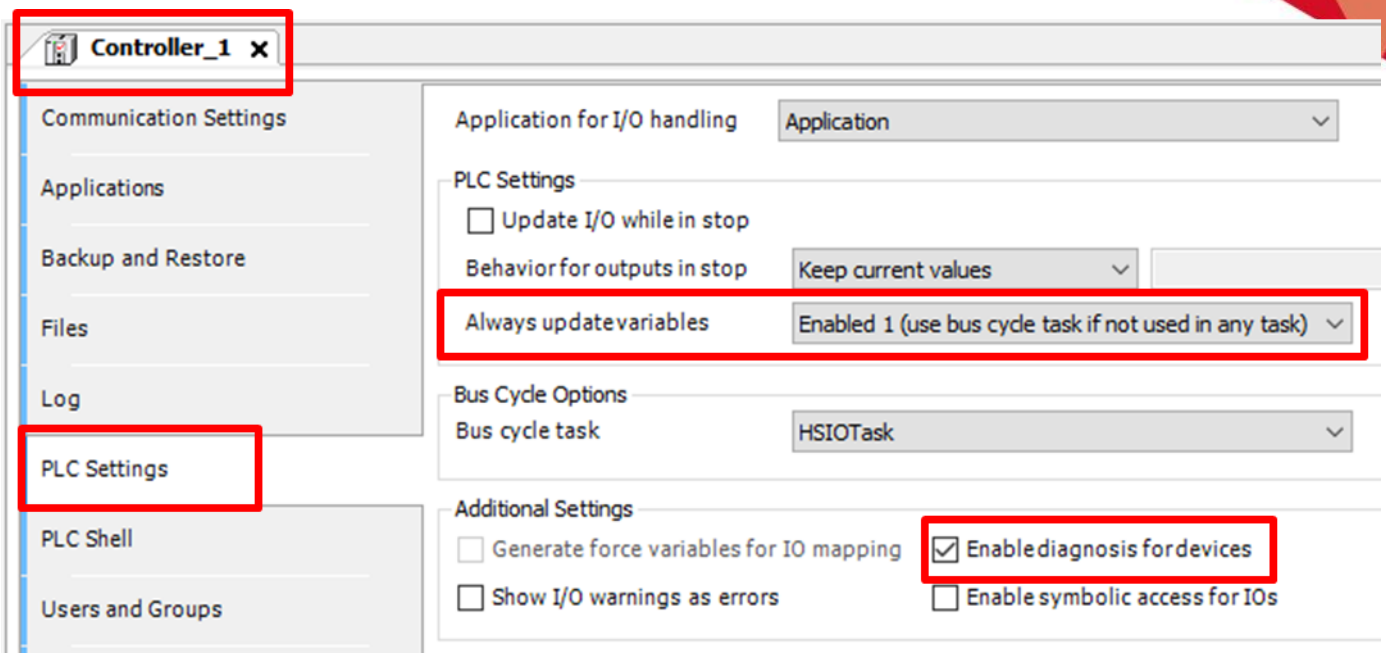
В поле Network Interface появится название интерфейса:



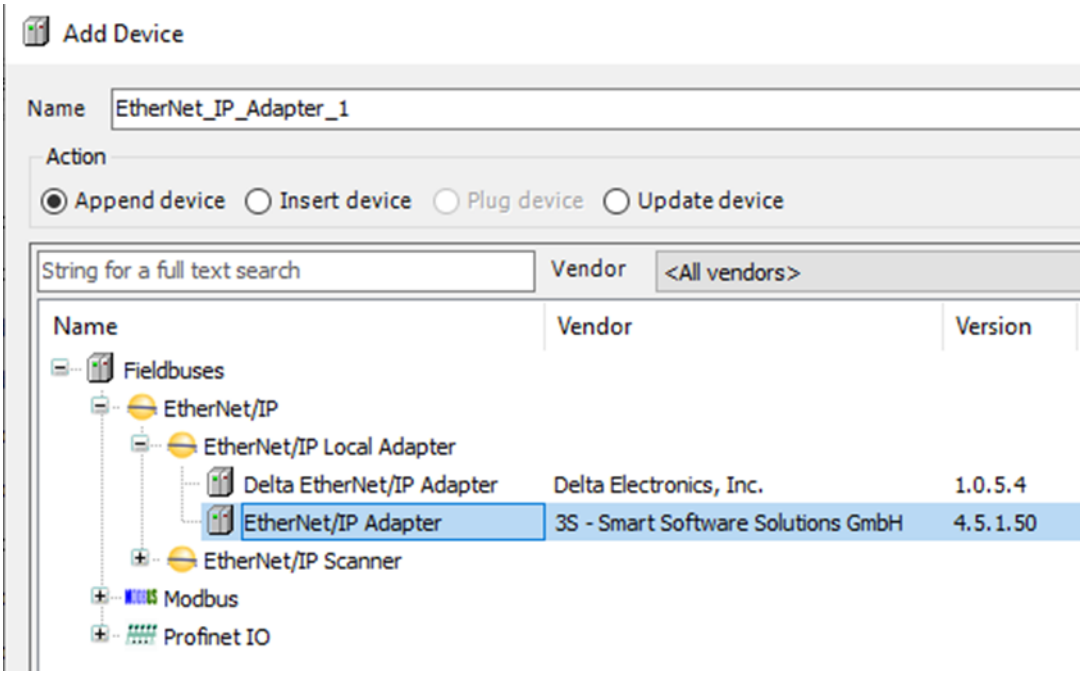
После чего Адаптер будет привязан к Ethernet порту с конкретным IP адресом.

Далее двойным щелчком левой кнопки мыши откройте вкладку **Device** и в пункте **PLC Settings** разрешите обновление переменных. Данный шаг позволяет видеть изменение переменных и регистров без написания в теле программы, т.е. в таблице Watch и в таблицах мэпинга.

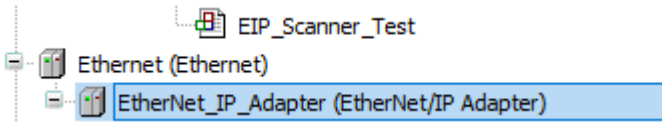
Также, для осуществления диагностики состояния подключенных к проекту устройств необходимо поставить флажок напротив пункта «Enable diagnosis for devices».



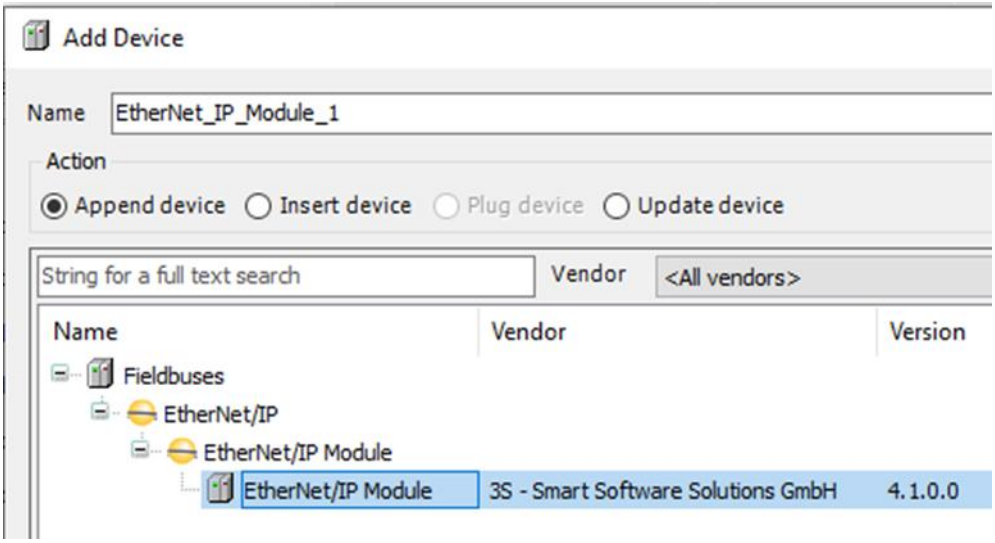
Щёлкните в древе правой кнопкой мышки на пункте **Ethernet** и в отрывшемся меню выберите пункт **Add Device**. Далее в открывшемся окне выберите пункт **Fieldbuses – Ethernet/IP Adapter**:



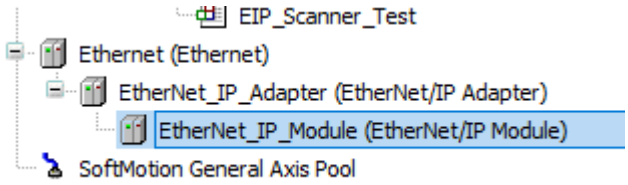
В древе проекта появится устройство:



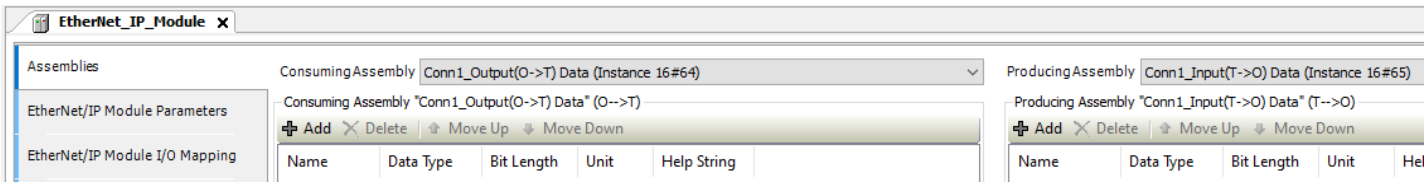
Щёлкните правой кнопкой мышки на данном пункте и добавьте устройство типа **Ethernet/IP Module**:



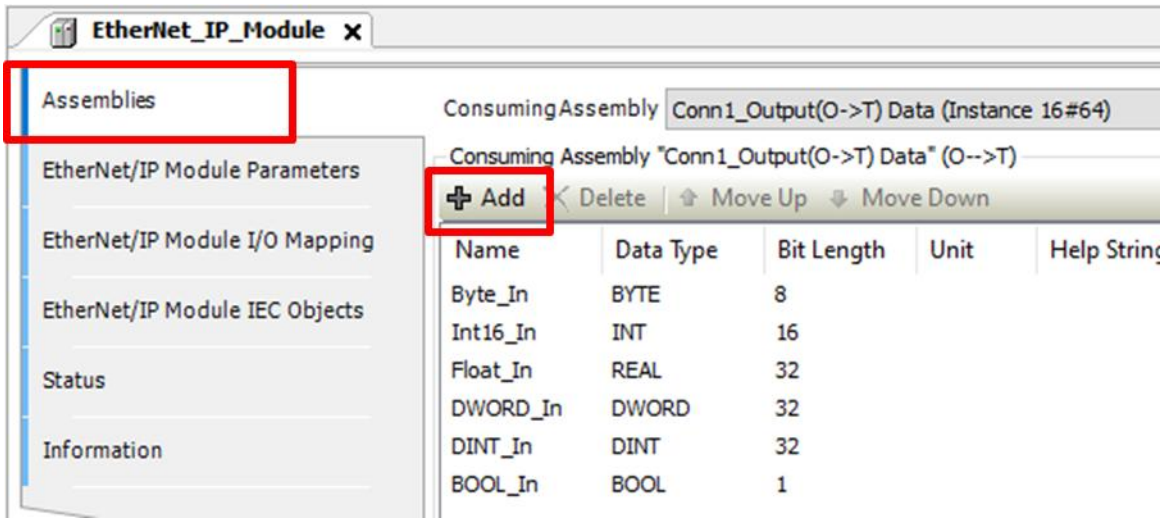
В древе проекта появится устройство:



Щёлкните дважды левой кнопкой мышки на данном пункте и в открывшейся вкладке выберите пункт **Assemblies**:

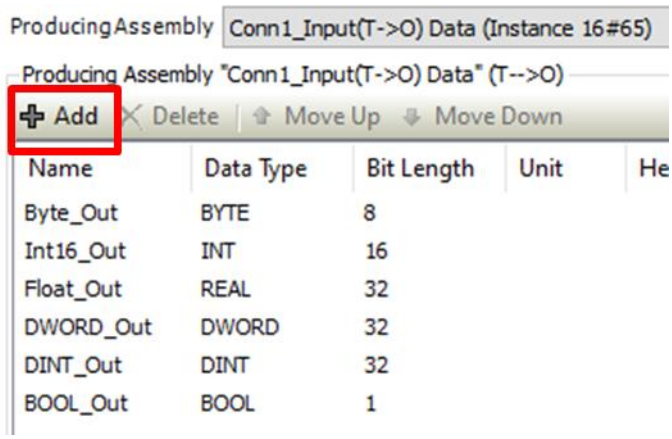


В данном разделе создаются пакеты для отправки и приёма от Мастера. Для добавления регистров в пакет на приём от Мастера нажимайте кнопку **Add** в поле **Consuming Assembly**:

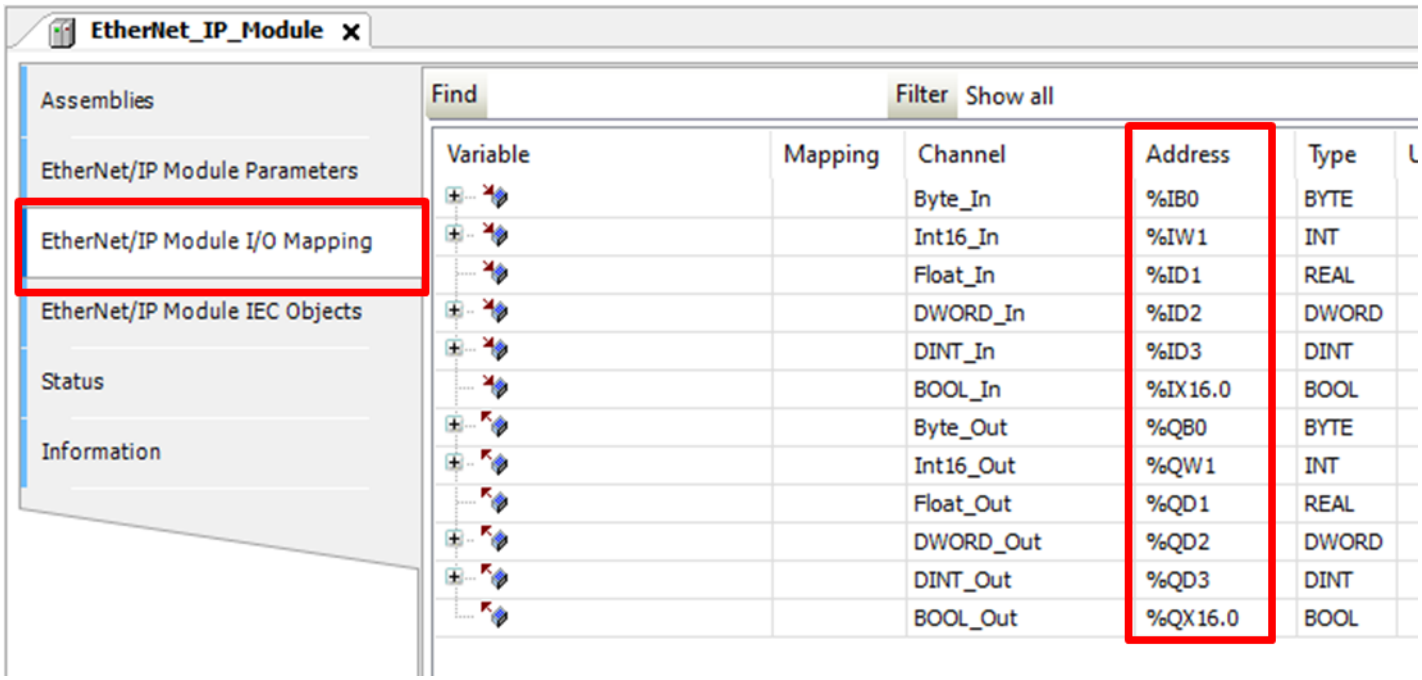


Теги можно создавать сразу с названием и любого из поддерживаемых типов данных.

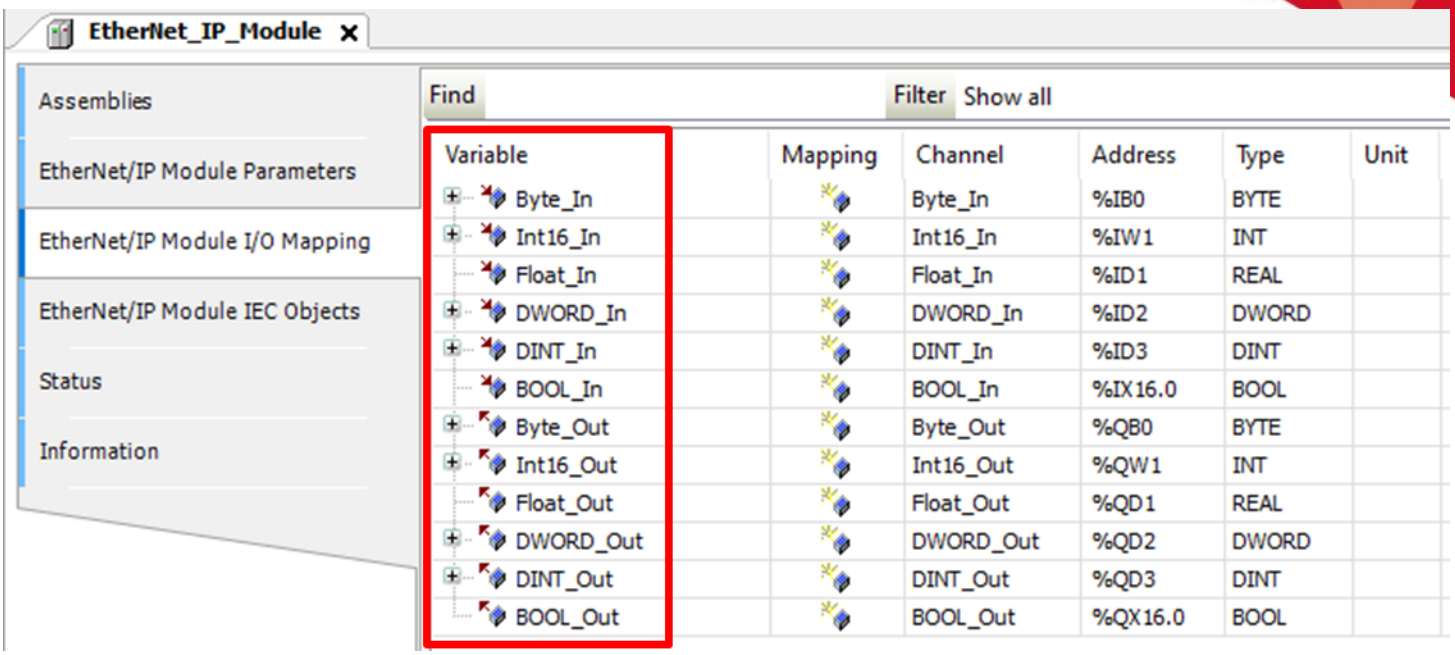
Для добавления регистров в пакет на отправку Мастеру нажимайте кнопку **Add** в поле **Producing Assembly**:



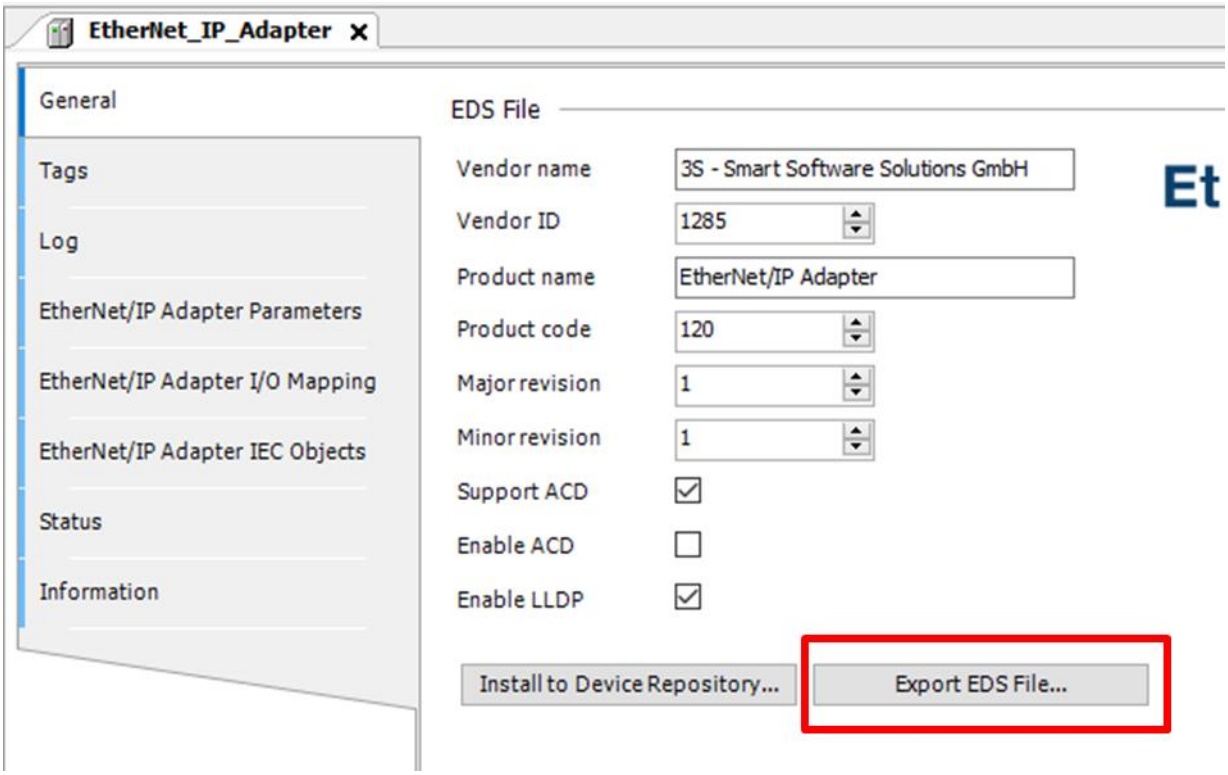
Общий список регистров можно увидеть в разделе **Ethernet/IP Module I/O Mapping**. Здесь же можно присвоить регистрам теги. Регистры типа %I** будут содержать данные от Мастера (Consuming Tags), а из регистров типа %Q** данные будут передаваться Мастеру (Producing Tags).



Для удобства лучше сразу присвоить тегам имена, которые можно будет использовать непосредственно в программе как Ведомого устройства (ME200):



Для экспорта таблицы тегов в виде стандартизированного XML файла необходимо дважды щёлкнуть левой кнопкой мышки на устройстве **EtherNet/IP_Adapter** и в открывшейся вкладке выбрать пункт **General**:



Нажмите кнопку **Export EDS File** и получите XML файл с тегами.

В файле лучше изменить поле **ProdName** со стандартного на своё название. Это поможет избежать путаницы со стандартным адаптером при выборе устройства из репозитария.

Для изменения поля откройте файл в Блокноте (Notepad) и измените название поле **ProdName**

```
[Device]
        VendCode = 1285;
        VendName = "3S - Smart Software Solutions GmbH";
        ProdType = 12;
        ProdTypeStr = "Communications Adapter";
        ProdCode = 120;
        MajRev = 1;
        MinRev = 1;
        ProdName = "EtherNet/IP Adapter";
```

например на такое:

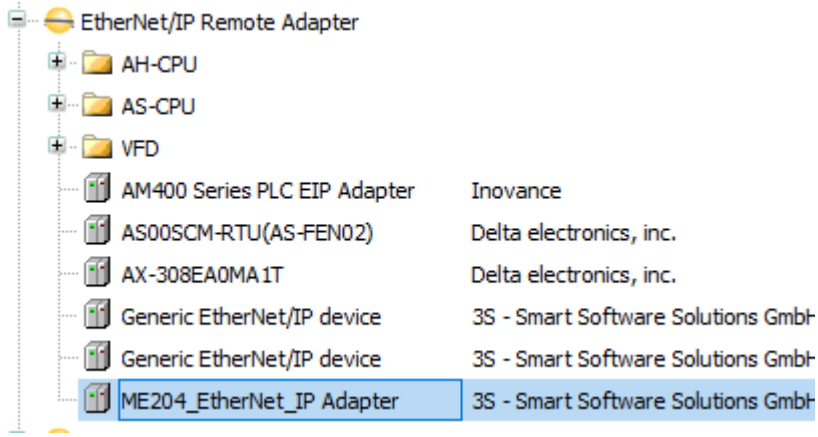
```
[Device]
        VendCode = 1285;
        VendName = "3S - Smart Software Solutions GmbH";
        ProdType = 12;
        ProdTypeStr = "Communications Adapter";
        ProdCode = 120;
        MajRev = 1;
        MinRev = 1;
        ProdName = "ME204_EtherNet_IP Adapter";
```

И можно изменить имя файла:

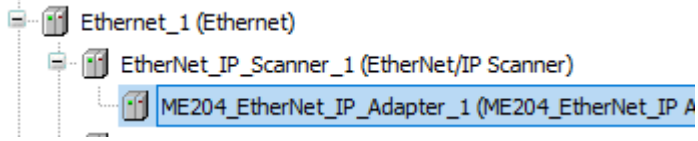


*Данный файл можно импортировать в любой **Ethernet/IP Scanner (Master).***

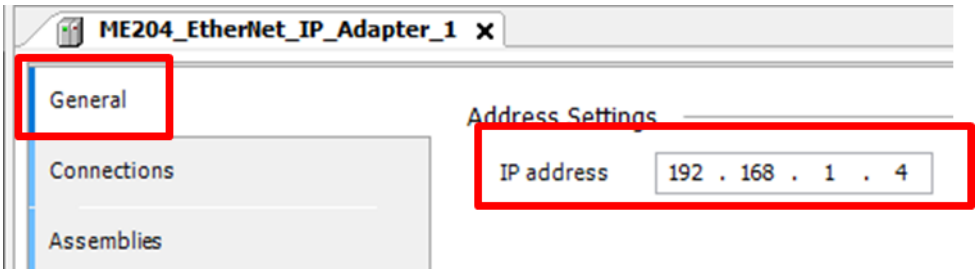
В качестве **Ethernet/IP Scanner (Master)** рассмотрим контроллер Delta AX-308E.
Создайте проект. При необходимости используйте документацию на данный контроллер.
Импортируйте в репозиторий устройств полученный на предыдущем шаге XML файл с тегами от контроллера MX308 (Adapter).



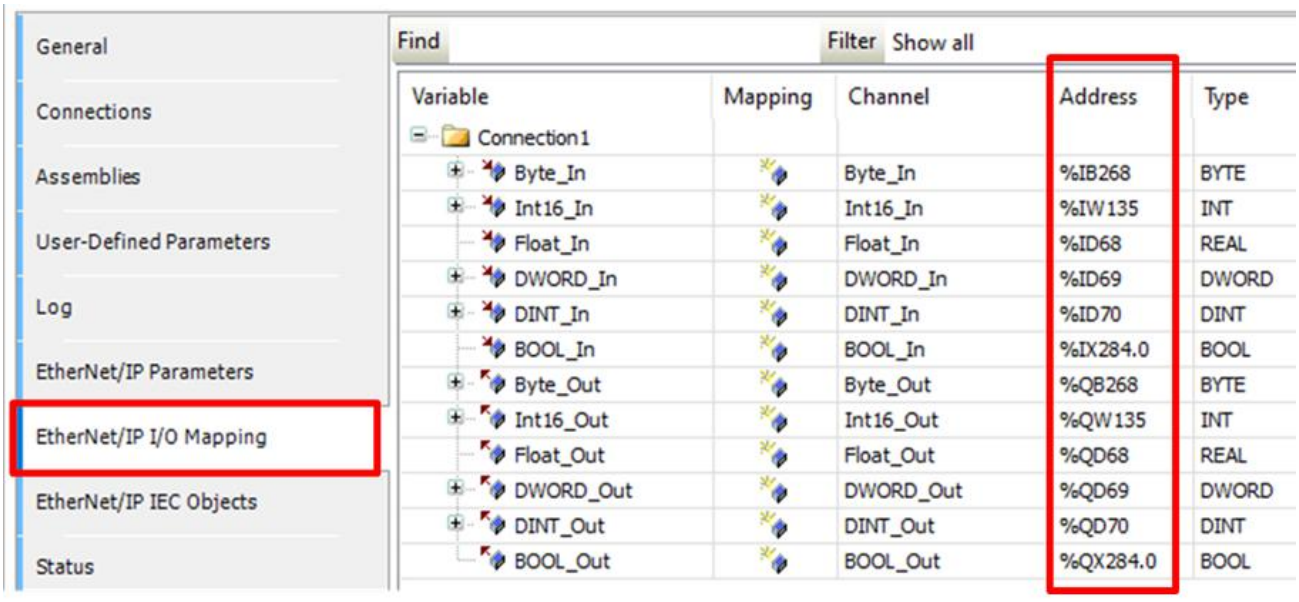
Добавьте устройство в проект:



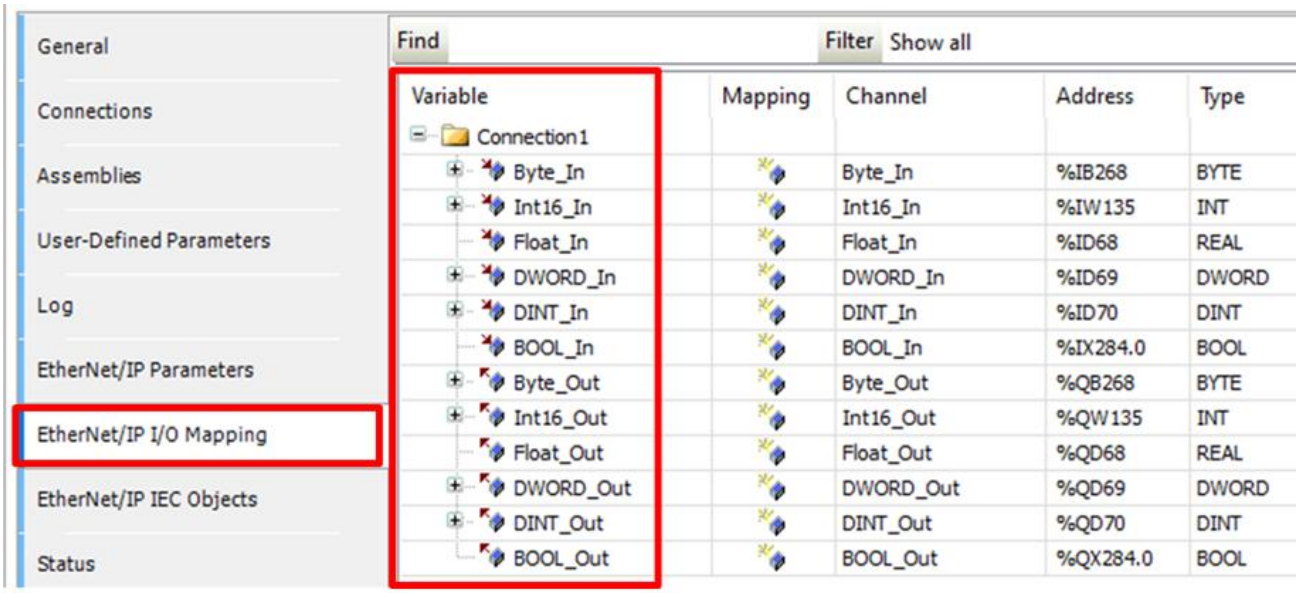
Щёлкните на нём двойным щелчком левой кнопкой мышки и в отрывшейся вкладке выберите пункт **General**, где необходимо установить IP адрес Адаптера:



Система автоматически раздаст адреса тегам, которые содержались в импортированном XML файле Адаптера (контроллера ME200). Адреса можно посмотреть в пункте **EtherNet/IP I/O Mapping**:



Также, необходимо объявить переменные для обращения к импортированным тегам в программе Мастера. Это является обязательным, так как при обращении к физическим регистрам они трактуются как целочисленные. А при объявлении переменной тип данных объявляется явным образом и система будет правильно трактовать данные:



Variable	Mapping	Channel	Address	Type
Connection1				
Byte_In		Byte_In	%IB268	BYTE
Int16_In		Int16_In	%IW135	INT
Float_In		Float_In	%ID68	REAL
DWORD_In		DWORD_In	%ID69	DWORD
DINT_In		DINT_In	%ID70	DINT
BOOL_In		BOOL_In	%IX284.0	BOOL
Byte_Out		Byte_Out	%QB268	BYTE
Int16_Out		Int16_Out	%QW135	INT
Float_Out		Float_Out	%QD68	REAL
DWORD_Out		DWORD_Out	%QD69	DWORD
DINT_Out		DINT_Out	%QD70	DINT
BOOL_Out		BOOL_Out	%QX284.0	BOOL

Обращение в программе нужно осуществлять именно через объявленные переменные. Например, в программе Мастера задаются следующие значения переменным:

Watch 1

Expression	Application	Type	Value
MX308_EtherNet_IP_Adapter.eState	Device.Application	ADAPTERSTATE	RUNNING
IoConfig_Globals_Mapping.Byte_Out	Device.Application	BYTE	255
IoConfig_Globals_Mapping.Int16_Out	Device.Application	INT	32767
IoConfig_Globals_Mapping.Float_Out	Device.Application	REAL	1234.567
IoConfig_Globals_Mapping.DWORD_Out	Device.Application	DWORD	4022333445
IoConfig_Globals_Mapping.DINT_Out	Device.Application	DINT	1022333445
IoConfig_Globals_Mapping.BOOL_Out	Device.Application	BIT	TRUE
IoConfig_Globals_Mapping.Byte_In	Device.Application	BYTE	0
IoConfig_Globals_Mapping.Int16_In	Device.Application	INT	0
IoConfig_Globals_Mapping.Float_In	Device.Application	REAL	0
IoConfig_Globals_Mapping.DWORD_In	Device.Application	DWORD	0
IoConfig_Globals_Mapping.DINT_In	Device.Application	DINT	0
IoConfig_Globals_Mapping.BOOL_In	Device.Application	BIT	FALSE

Контроль состояния Ведомого устройства можно осуществлять через элемент структуры **Device_Name.eState**

ME204_EtherNet_IP_Adapter_1.eState

Данные автоматически будут попадать в регистры Ведомого устройства ME204-C (Consumed Tags):

Watch 1

Expression	Type	Value
EtherNet_IP_Module.eState	MODULESTATE	RUNNING
IoConfig_Globals_Mapping.Byte_In	BYTE	255
IoConfig_Globals_Mapping.Int16_In	INT	32767
IoConfig_Globals_Mapping.Float_In	REAL	1234.567
IoConfig_Globals_Mapping.DWORD_In	DWORD	4022333445
IoConfig_Globals_Mapping.DINT_In	DINT	1022333445
IoConfig_Globals_Mapping.BOOL_In	BIT	TRUE
IoConfig_Globals_Mapping.Byte_Out	BYTE	0
IoConfig_Globals_Mapping.Int16_Out	INT	0
IoConfig_Globals_Mapping.Float_Out	REAL	0
IoConfig_Globals_Mapping.DWORD_Out	DWORD	0
IoConfig_Globals_Mapping.DINT_Out	DINT	0
IoConfig_Globals_Mapping.BOOL_Out	BIT	FALSE

В программе Ведомого состояние связи с Мастером можно также контролировать через элемент структуры **Device_Name.eState**:

EtherNet_IP_Module.eState

И наоборот, если в программе Ведомого (ME204-C) задать значения для исходящих тегов (Produced Tags):

Watch 1

Expression	Type	Value
EtherNet_IP_Module.eState	MODULESTATE	RUNNING
IoConfig_Globals_Mapping.Byte_In	BYTE	0
IoConfig_Globals_Mapping.Int16_In	INT	0
IoConfig_Globals_Mapping.Float_In	REAL	0
IoConfig_Globals_Mapping.DWORD_In	DWORD	0
IoConfig_Globals_Mapping.DINT_In	DINT	0
IoConfig_Globals_Mapping.BOOL_In	BIT	FALSE
IoConfig_Globals_Mapping.Byte_Out	BYTE	127
IoConfig_Globals_Mapping.Int16_Out	INT	-32768
IoConfig_Globals_Mapping.Float_Out	REAL	-9876.321
IoConfig_Globals_Mapping.DWORD_Out	DWORD	4000222333
IoConfig_Globals_Mapping.DINT_Out	DINT	-20123555
IoConfig_Globals_Mapping.BOOL_Out	BIT	TRUE

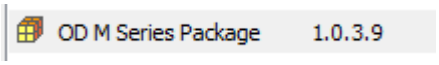
то данные автоматически попадут в регистры Мастера (Delta AX-308E):

Watch 1

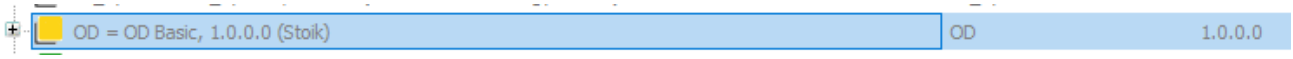
Expression	Application	Type	Value
MX308_EtherNet_IP_Adapter.eState	Device.Application	ADAPTERSTATE	RUNNING
IoConfig_Globals_Mapping.Byte_Out	Device.Application	BYTE	0
IoConfig_Globals_Mapping.Int16_Out	Device.Application	INT	0
IoConfig_Globals_Mapping.Float_Out	Device.Application	REAL	0
IoConfig_Globals_Mapping.DWORD_Out	Device.Application	DWORD	0
IoConfig_Globals_Mapping.DINT_Out	Device.Application	DINT	0
IoConfig_Globals_Mapping.BOOL_Out	Device.Application	BIT	FALSE
IoConfig_Globals_Mapping.Byte_In	Device.Application	BYTE	127
IoConfig_Globals_Mapping.Int16_In	Device.Application	INT	-32768
IoConfig_Globals_Mapping.Float_In	Device.Application	REAL	-9876.321
IoConfig_Globals_Mapping.DWORD_In	Device.Application	DWORD	4000222333
IoConfig_Globals_Mapping.DINT_In	Device.Application	DINT	-20123555
IoConfig_Globals_Mapping.BOOL_In	Device.Application	BIT	TRUE

Чтение и установка часов реального времени

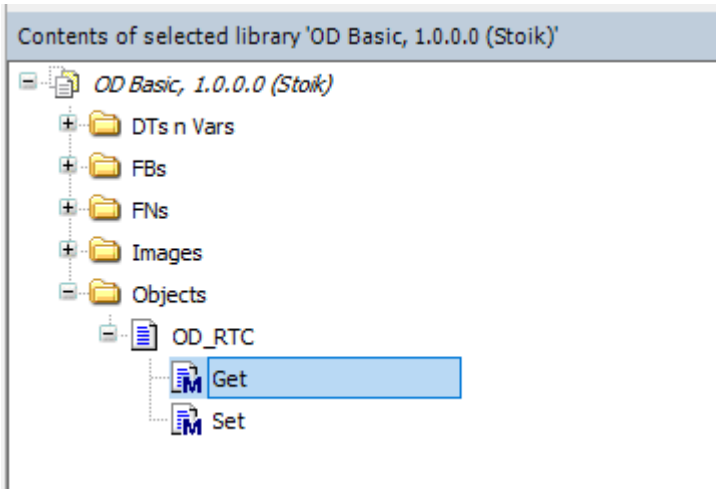
Контроллеры семейства ME200 имеют встроенные часы реального времени (RTC). Для поддержки работы часов используется специальная модификация батарейки CR1632 (с выводом проводов), установленная в отсеке в корпусе контроллера. В составе базового пэкиджа с описанием устройства



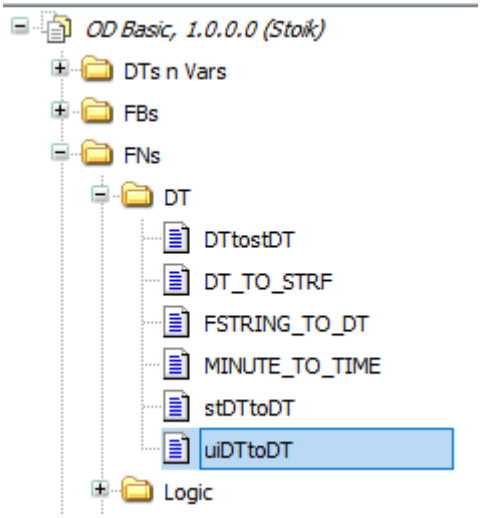
ставится библиотека **OD Basic**:



В данной библиотеке содержится Функциональный Блок **OD_RTC**, в котором есть два метода **Get** и **Set** для обеспечения доступа к функционалу данного ФБ. Метод **Get** позволяет получить текущее значение часов реального времени контроллера в формате DT, а метод **Set** позволяет установить часы реального времени из программы контроллера также в формате DT.



Для конвертации формата DT в тип UINT библиотека **OD Basic** содержит три функции **DTtostDT, stDTtoDT** и **uiDTtoDT**:

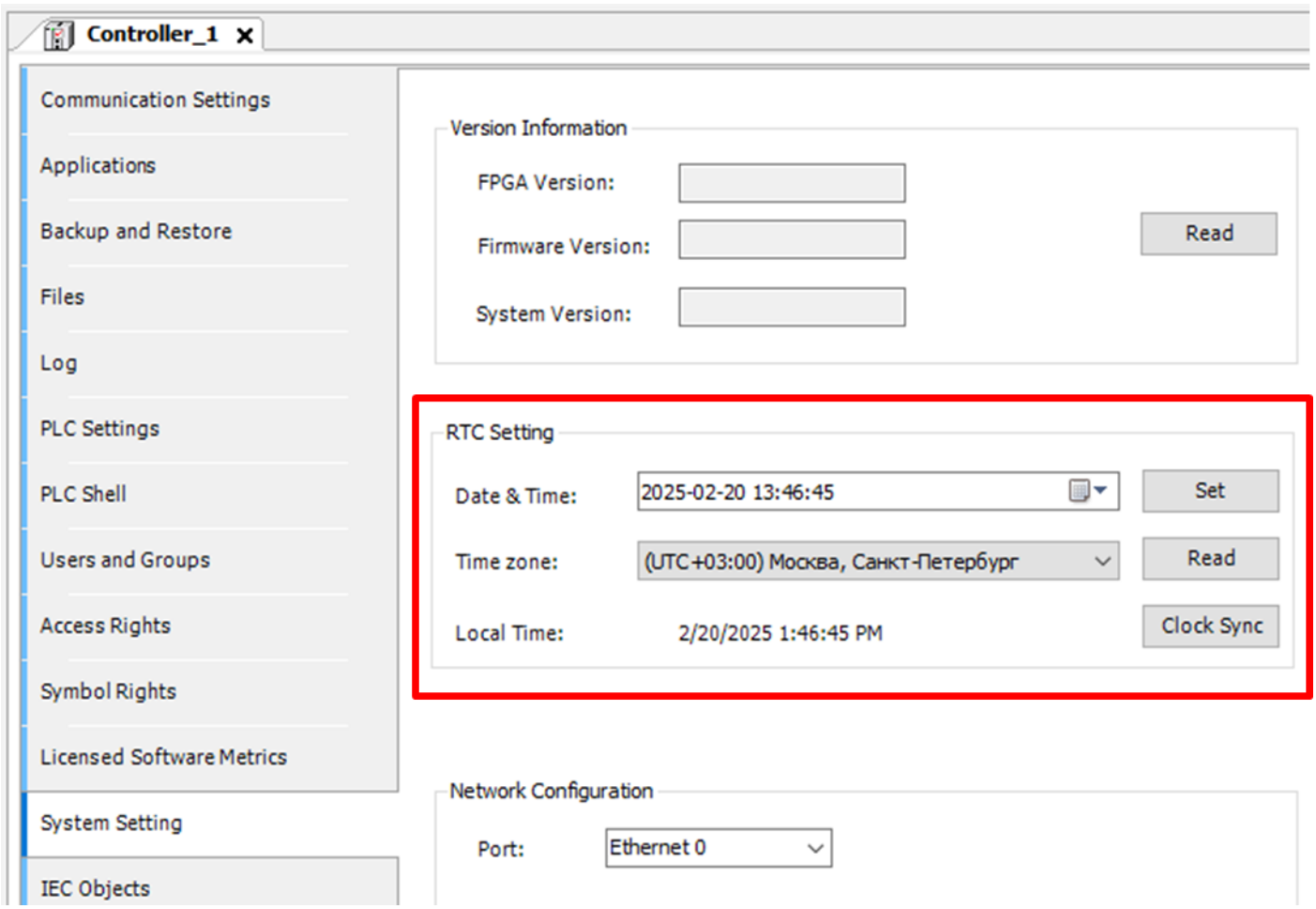


DTtostDT – функция конвертации DT формата даты, времени в структуру с элементами типа UINT

stDTtoDT – функция конвертации структуры даты, времени с элементами типа UINT в значение в DT формате

uiDTtoDT – функция конвертации UINT значений даты, времени в DT формат

Часы реального времени можно выставить также из среды программирования. Для этого необходимо открыть вкладку **Device** пункт **System Settings**:



Выставить нужные Дату и Время, а затем кнопкой **Set** записать в контроллер.
(связь с контроллером должна быть установлена заранее)

Кнопкой **Read** можно прочитать текущее значение часов реального времени контроллера.

Чтение часов реального времени контроллера методом OD_RTC.Get

В среде программирования CODESYS методы, как и функции, можно вызывать только на языке ST, поэтому вызов метода будет приведён на этом языке:

```
// Чтение часов реального времени (RTC)
dtNow DT#2025-2-21-20:6:7 := fbOD_RTC.Get(); // Постоянное чтение часов реального времени UTC+3
```

```
PROGRAM _RTC
VAR
	dtNow:		DT; //переменная для отображения считанной текущей Даты/Времени
	fbOD_RTC:	OD.OD_RTC; //объявление экземпляра ФБ OD_RTC
END_VAR
```

```
// Чтение часов реального времени (RTC)
dtNow := fbOD_RTC.Get(); // Постоянное чтение часов реального времени UTC+3: Moscow Time Zone
```

Запись часов реального времени контроллера методом OD_RTC.Set

```
// Запись часов реального времени (RTC)
IF xSetDT FALSE THEN
    fbOD_RTC.Set(dtSetDT DT#2025-2-21-11:3:0, 180);    // Запись. UTC+3: Moscow Time Zone  UTC+3 =
    xSetDT FALSE := FALSE;
END_IF RETURN
```

```
PROGRAM _RTC
VAR
	dtSetDT:	DT; //переменная для хранения Даты/Времени для записи в контроллер
	fbOD_RTC:	OD.OD_RTC; //объявление экземпляра ФБ OD_RTC
	xSetDT:		BOOL; //Тригер записи времени в контроллер
END_VAR
```

```
// Запись часов реального времени (RTC)
IF xSetDT THEN
	fbOD_RTC.Set(dtSetDT, 180);    // Запись. UTC+3: Moscow Time Zone  UTC+3 == iBias = 3*60 = 180
	xSetDT := FALSE;
END_IF
```

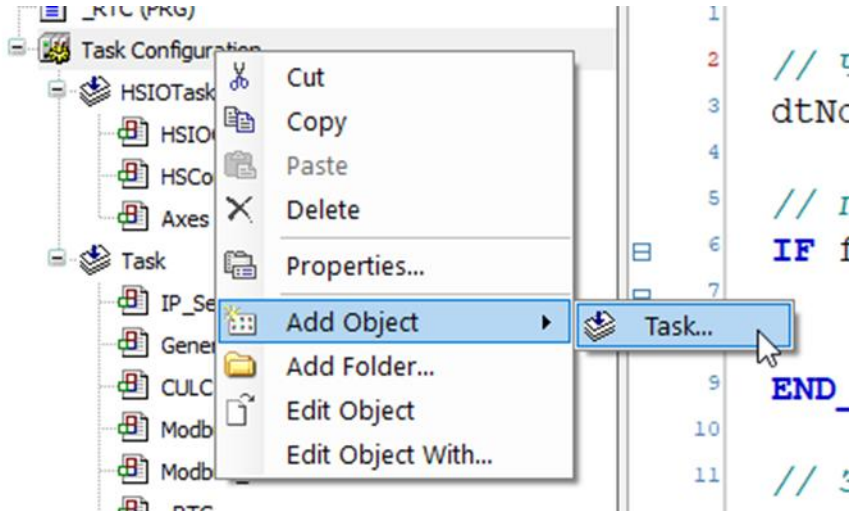
Примечание:
Минимальный год, который можно выставить – 2010 (1 января 00 00)

Прерывание по сигналу на входе контроллера

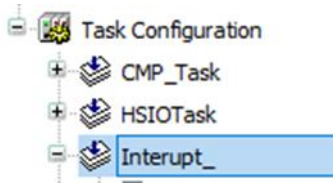
Контроллеры типа ME204-C имеют возможность отрабатывать назначенные программные блоки по фронту на дискретном входе контроллера. Прерывания могут быть назначены на входы 00 – 05 контроллера (всего 6 прерываний). Можно использовать передний, задний или оба фронта.

Для назначения прерываний по входам в древо проекта контроллера должен быть добавлен адаптер входов-выходов на ЦПУ – **MEHighSpeedIO** (см. соответствующую Главу данного Руководства).

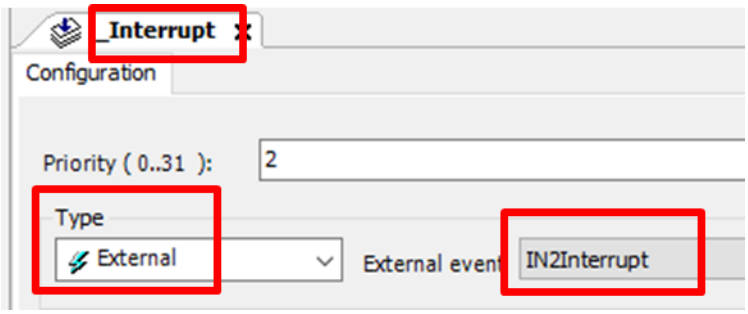
В начале необходимо создать Задачу (Task) обработки прерывания. Для этого встаньте в древе проекта на пункт **Task Configuration** и правой кнопкой мышки вызовите меню, в котором выберите пункт **Add Object – Task**:



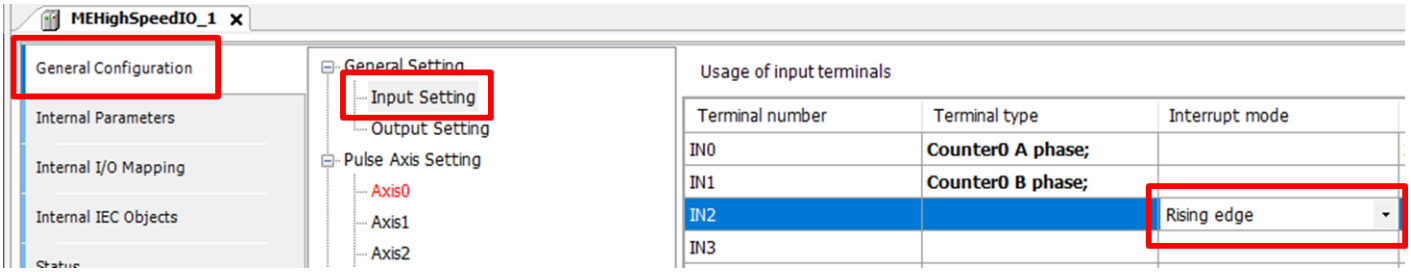
Дайте название Задаче, например **_Interrupt**. В древе проекта появится Задача **_Interrupt**:



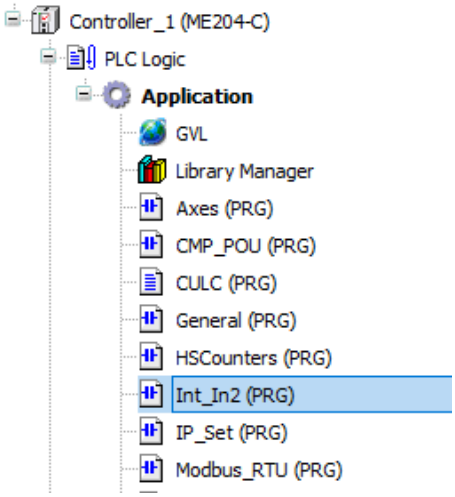
Щёлкните дважды левой кнопкой мышки на пункте **_Interrupt**, откроется вкладка с настройками параметров Задачи. Выберите тип задачи **External** и назначьте вход, например 02 (**IN2Interrupt**):



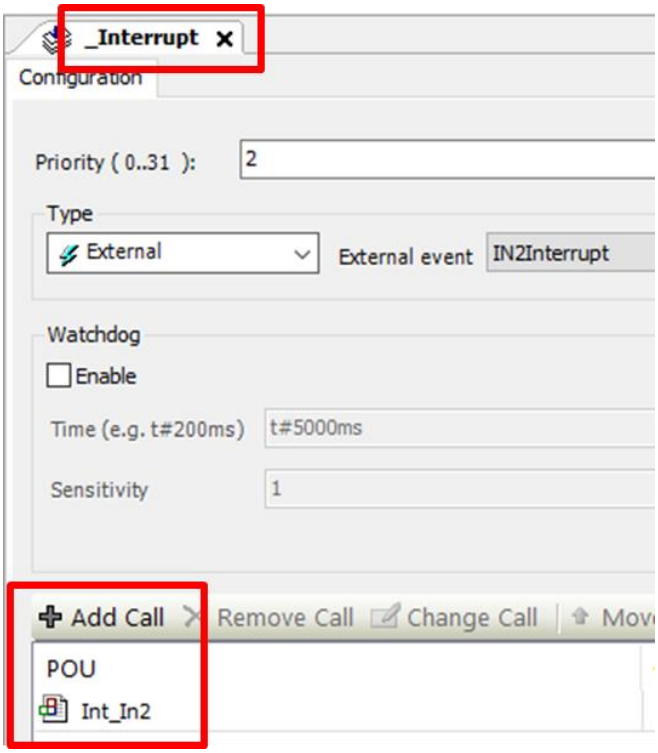
Щёлкните дважды на пункте **MEHighSpeedIO** и в открывшейся вкладке выберите раздел **General Configuration – Input Setting**. Напротив входа IN2 появится меню выбора фронта сигнала: Передний, задний или оба. На картинке ниже выбран передний фронт (**Rising edge**):



Далее создайте программный блок (POU), в котором необходимо поместить код, который должен быть исполнен по наступлению события, в нашем случае переднего фронта на входе 02. Дайте название, например **Int_In2**:

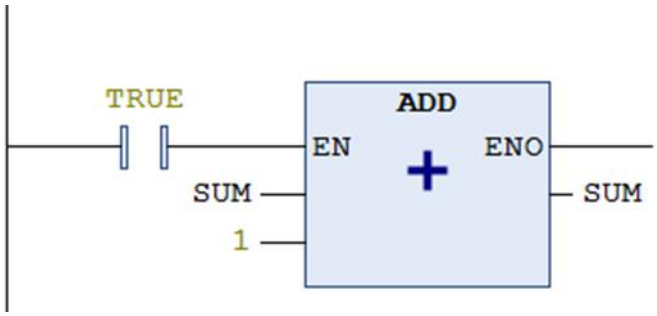


Созданный программный блок Int_In2 необходимо привязать к Задаче **_Interrupt**.

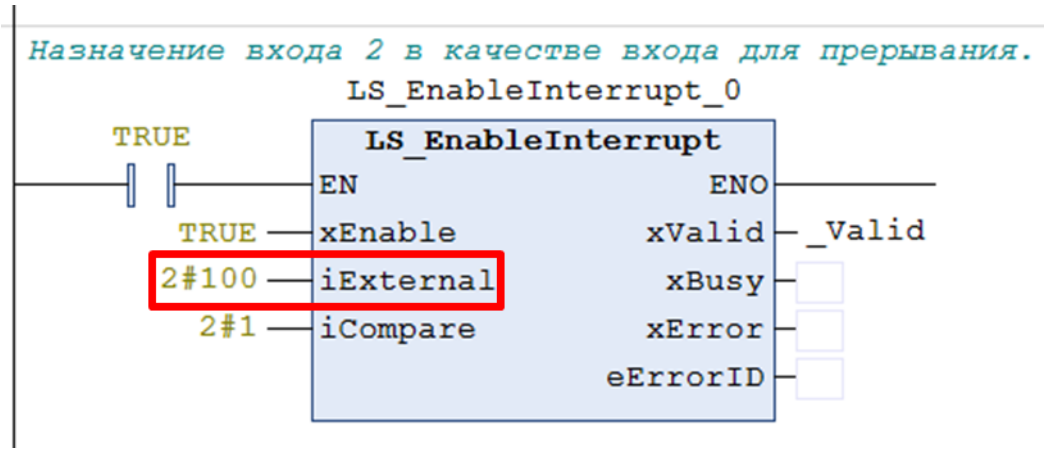


В теле POU **Int_In2** создайте одну строчку программы чисто в целях демонстрации работы прерывания:

Scope	Name	Address	Data type
VAR	**SUM**		INT



Далее необходимо зарегистрировать прерывание в команде **LS_EnableInterrupt**, которую необходимо поместить в циклической задаче (т.е. не в самом прерывании !!!):



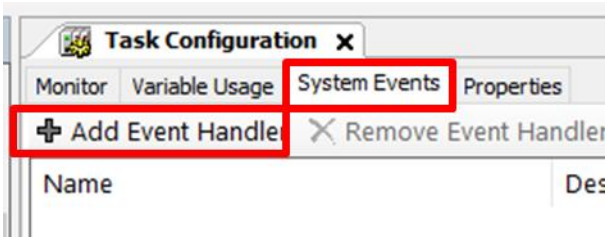
На ножке **iExternal** в двоичном коде указываются входы ЦПУ, которые назначены для работы в режиме прерывания. В нашем примере это вход 02, поэтому бит 2 взводится в 1. Нумерация справа на лево.

После загрузки в контроллер можно проверить работу вышеприведённой процедуры.
Если бы команда **ADD** стояла в обычной циклической Задаче, то она бы исполнялась в каждом скане и переменная **SUM** быстро бы увеличивалась. Но данный код привязан к Задаче по прерыванию, следовательно исполняться будет разово при наступлении соответствующего события – в нашем случае переднего фронта на входе 02.

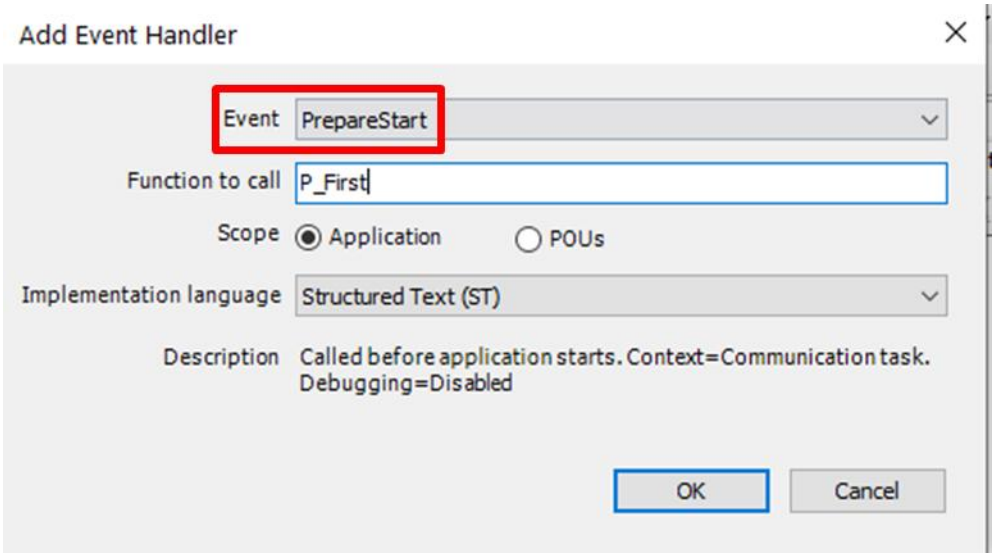
Импульс при переводе в состояние RUN

Практически во всех проектах возникает необходимость разового исполнения каких-либо действий при переводе контроллера в состояние Работа (RUN), например присвоить какие-либо константы и т.п. В среде CODESYS нет готового специального флага, который мог бы выполнить данную задачу. Поэтому ниже приводится процедура, которая позволяет реализовать данный функционал.

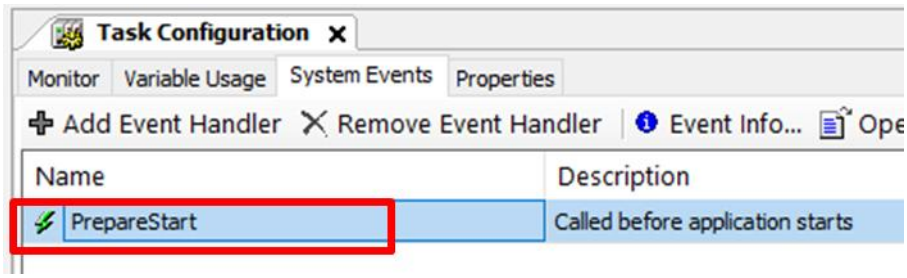
Для этого встаньте в древе проекта на пункт **Task Configuration** и двойным щелчком левой кнопки мышки откройте вкладку **Task Configuration** и выберите пункт **System Events** и нажмите кнопку **Add Event Handler**:



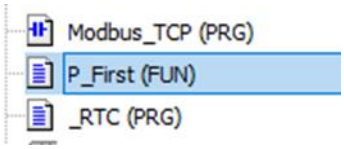
В открывшемся окне выберите тип события **Event => PrepareStart**, дайте имя, например **P_First** и нажмите **OK**.



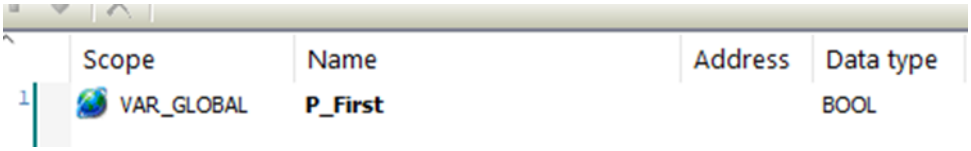
В списке обработчиков появится пункт **PrepareStart**:



а в древе проекта появится функция **P_First**:

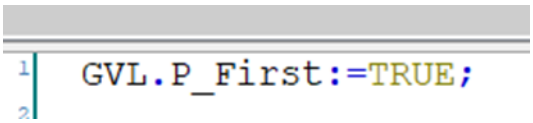


Далее откройте Таблицу Глобальных Переменных (**GVL**) и создайте глобальную переменную **P_First**:



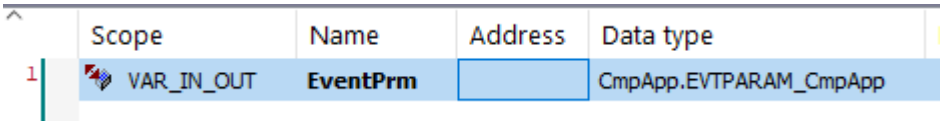
	Scope	Name	Address	Data type
1	VAR_GLOBAL	**P_First**		BOOL

Отройте двойным щелчком функцию **P_First** и напишите одну строчку кода:



```
1  GVL.P_First:=TRUE;
2
```

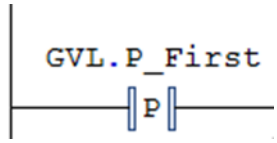
В таблице переменных система автоматически уже создала переменную:



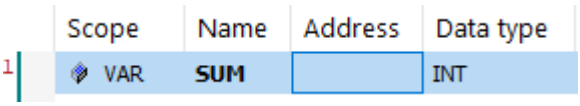
	Scope	Name	Address	Data type
1	VAR_IN_OUT	**EventPrm**		CmpApp.EVTPARAM_CmpApp

никаких других переменных в функции-обработчике создавать не надо!

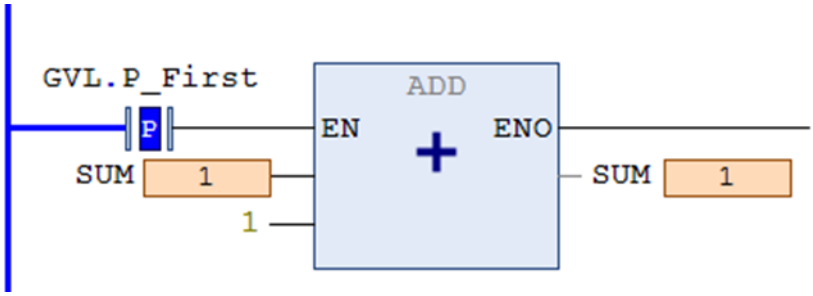
На этом процедура оформления контакта, замкнутого один раз при переводе контроллера в работу, окончена. В качестве контакта выступает глобальная переменная **P_First**, которую необходимо оформить как контакт по переднему фронту:



Для проверки работоспособности данной процедуры можно в любой циклической задаче написать простейший код типа:



	Scope	Name	Address	Data type
1	VAR	**SUM**		INT

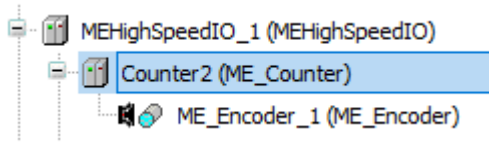


Переменная SUM будет равна 1 на всё время работы контроллера.

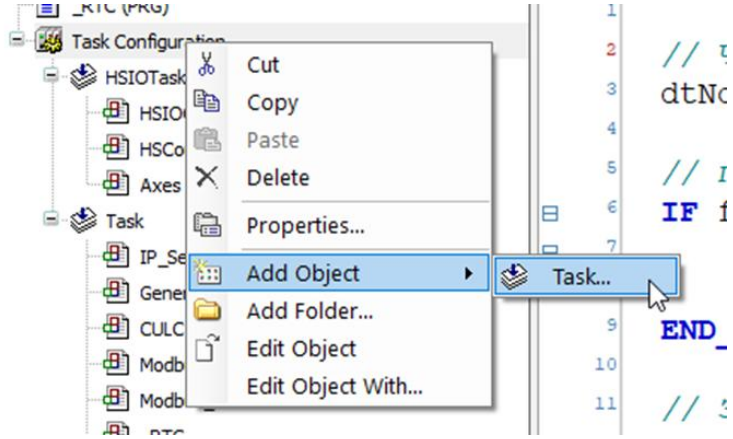
Высокоскоростное сравнение

В ряде задач требуется выполнить сравнение текущего значения высокоскоростного счётчика с заданным порогом и исполнить в режиме прерывания определённый код или включить выход на контроллере. Для реализации подобной процедуры необходимо выполнить ряд действий.

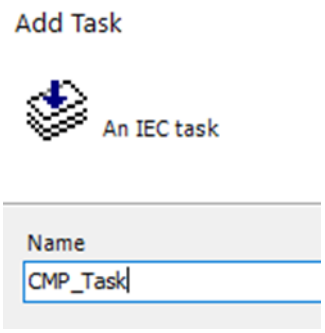
Для реализации высокоскоростного сравнения в древо проекта контроллера должен быть добавлен адаптер входов-выходов на ЦПУ – **MEHighSpeedIO** (см. соответствующую Главу данного Руководства). Это необходимо для организации высокоскоростного счётчика, текущее значение оси которого и будет сравниваться с заданным порогом. Создайте счётчик **Counter2** (см. Работа с высокоскоростными счётчиками, стр. 91 настоящего Руководства).



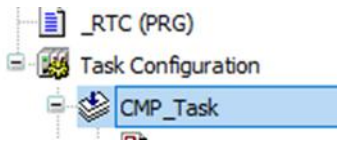
В начале необходимо создать Задачу (Task) обработки прерывания. Для этого встаньте в древе проекта на пункт **Task Configuration** и правой кнопкой мышки вызовите меню, в котором выберите пункт **Add Object – Task**:



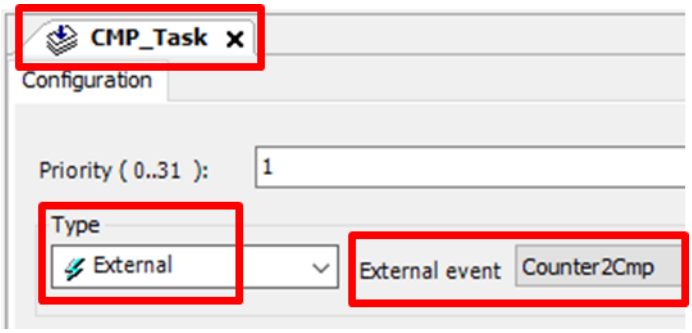
Дайте название Задаче, например **CMP_Task**.



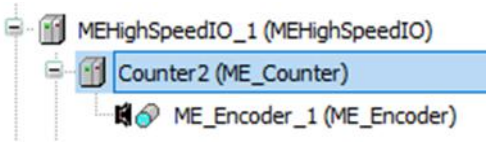
В древе проекта появится Задача **CMP_Task**:



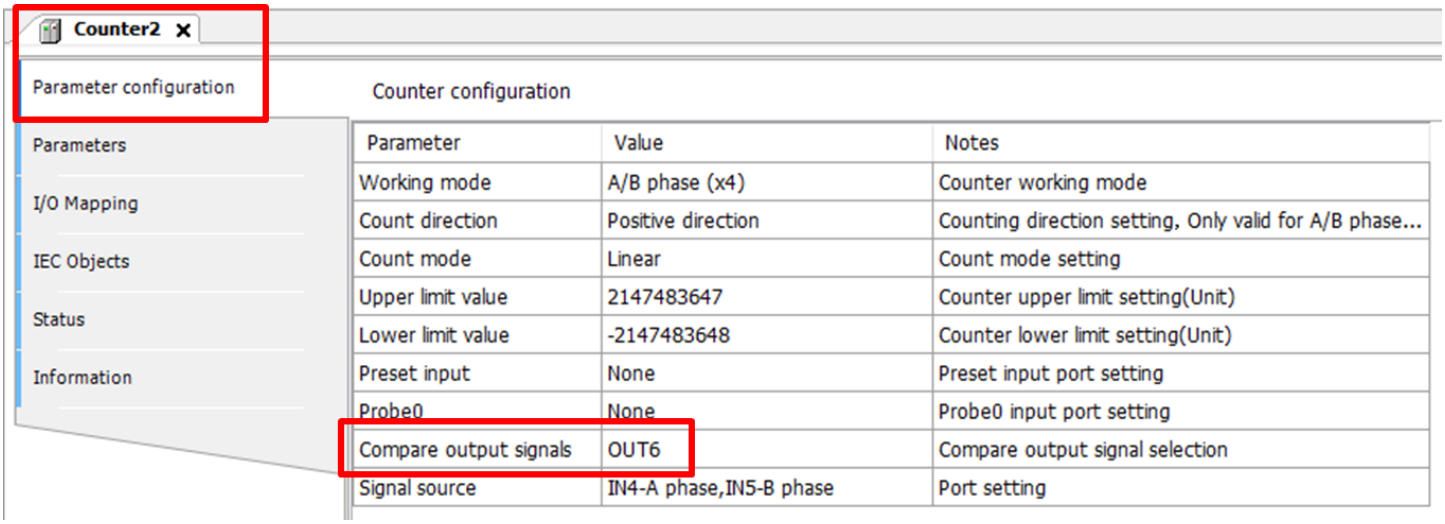
Щёлкните дважды левой кнопкой мышки на пункте **CMP_Task**, откроется вкладка с настройками параметров Задачи. Выберите тип задачи **External** и назначьте счётчик, в нашем примере **Counter2Cmp**:



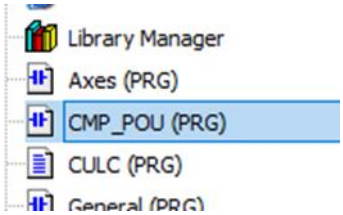
Далее щёлкните дважды левой кнопкой мышки на пункте **Counter2** в древе проекта:



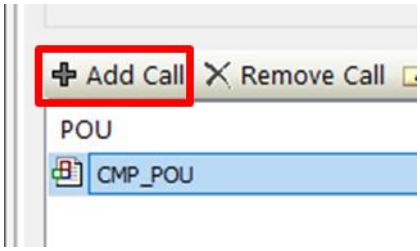
Откроется вкладка с настройками счётчика, в которой необходимо выбрать раздел **Parameter Configuration** и в параметре **Compare Output Signals** выбрать выход, который будет включаться при наступлении события прерывания по достижению порога высокоскоростного счёта. На рисунке ниже выбран выход 06 (**OUT6**):



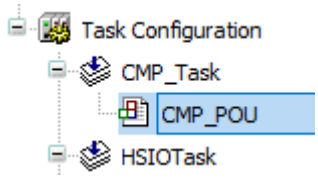
Далее создайте программный блок, например **CMP_POU**:



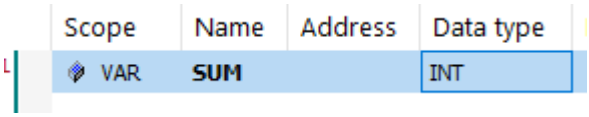
и привяжите его к Задаче **CMP_Task**:

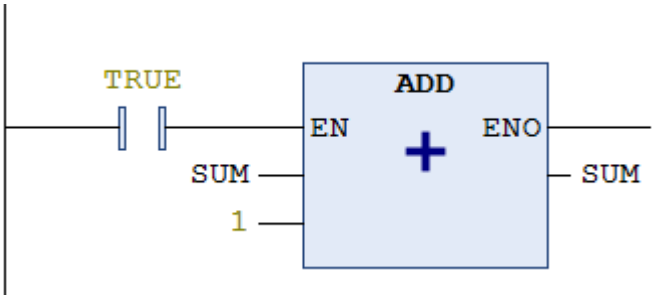


В древе проекта появится пункт:



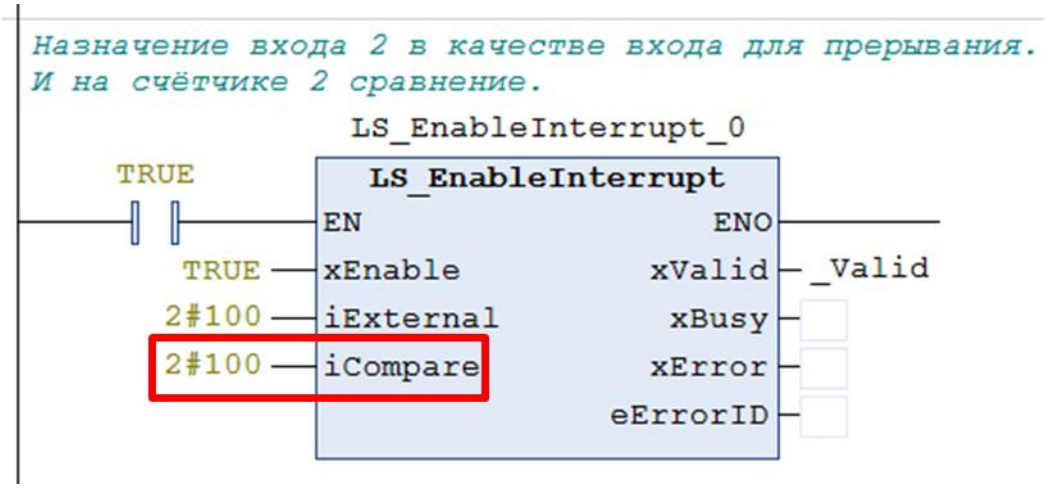
Далее в программном блоке CMP_POU для целей демонстрации можно создать простой код, который будет исполняться только по наступлению события достижения порога высокоскоростного счёта:



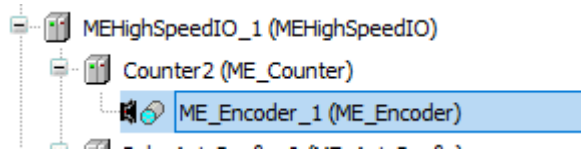


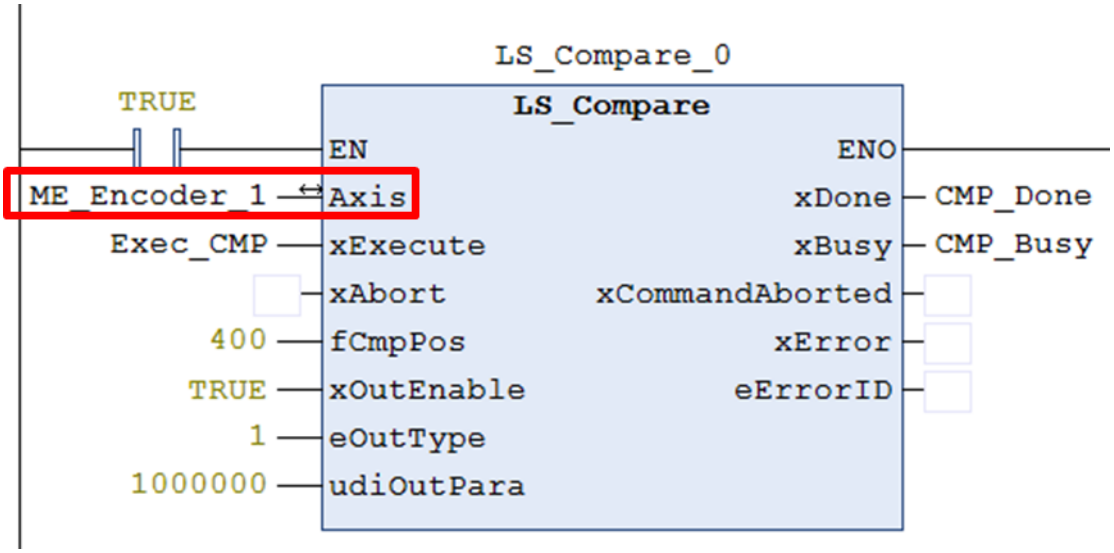
(если бы данная команда стояла в циклической Задаче, то исполнялась бы в каждом скане, а так только по наступлению события).

Прерывание необходимо зарегистрировать в команде **LS_EnableInterrupt** в двоичном коде по номеру счётчика:



Далее в любой циклической задаче необходимо поставить команду **LS_Compare** и привязать по ***имени оси*** счётчик:



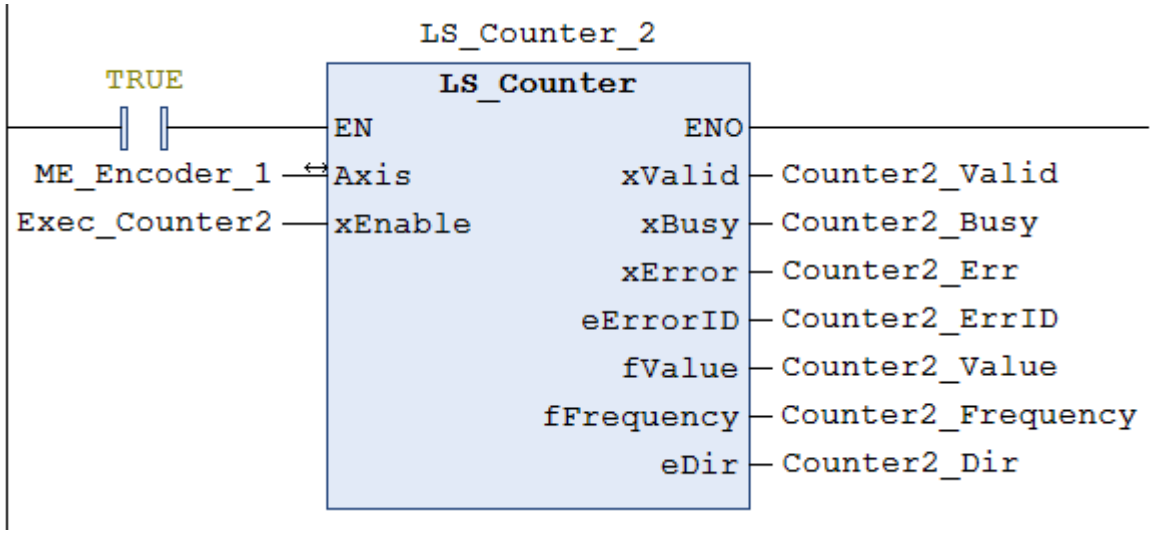


Описание основных ножек ФБ:
Axis – имя энкодерной оси, привязанной к счетчику в древе проекта
xExecute – запускает ФБ (необходимо взводить заново каждый раз после срабатывания ФБ)
fCmpPos – порог для сравнения. Количество импульсов в user units LREAL
eOutType – режим сравнения, поставить 1 (выход включен на установленное время по наступлению события)
xOutEnable – разрешение установки выхода при достижении порога, поставить TRUE
udiOutPara – время удержания выхода при срабатывании прерывания, мкс (1000000 = 1 секунда)
CMP_Done – флаг окончания сравнения
CMP_Busy – флаг готовности ФБ к сравнению

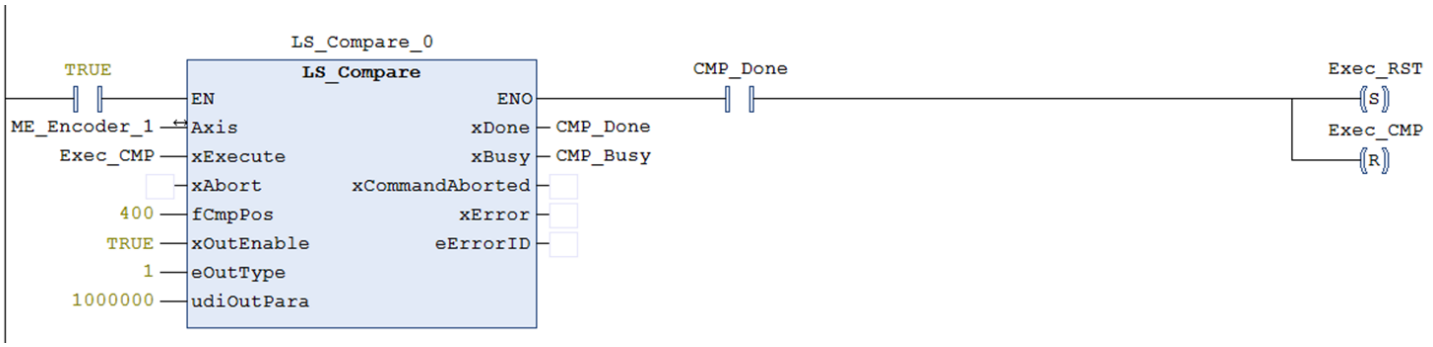
При работе с командой **LS_Compare** необходимо учитывать, что она срабатывает по фронту сигнала на ножке **xExecute**, загорается флаг готовности xBusy и команда готова к высокоскоростному сравнению. После выполнения сравнения флаг **xBusy** гаснет и загорается флаг **xDone**. Для осуществления следующей операции сравнения необходимо погасить и снова взвести ножку **xExecute**. Также, в программе у Вас должна быть включена инструкция высокоскоростного счёта для соответствующего счётчика **LS_Counter**. Сброс счётчика осуществляется командой **LS_PresetValue** со значением 0.

Таким образом, для реализации повторяющегося цикла работы инструкции сравнения необходимо:
1. Включить в программе высокоскоростной счётчик (включен постоянно)
2. Включить команду сравнения для первого цикла ножкой xExecute
3. По флагу xDone сбросить xExecute команды сравнения и взвести xExecute команды сброса счётчика
4 После выполнения сброса счётчика по флагу xDone включить ножку xExecute команды сравнения

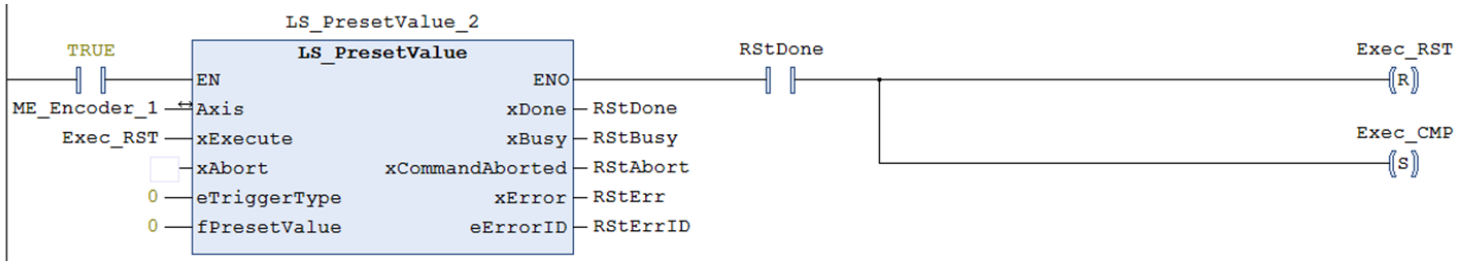
Команда активации счётчика:



Команда высокоскоростного сравнения:



Команда сброса счётчика:



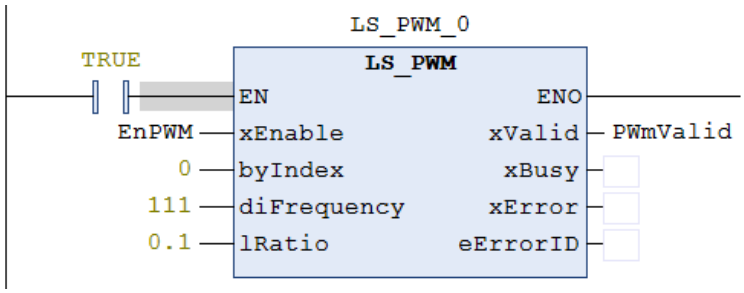
Обратите внимание, что во всех инструкциях привязка идёт к имени оси, а не к номеру счётчика.

Работа в режиме ШИМ импульсов

Контроллеры типа ME204-C имеют аппаратную возможность формировать ШИМ импульсы на выходах 14 и 15. Для организации данного функционала в проекте должен быть адаптер входов-выходов **MEHighSpeedIO**. Щёлкните дважды левой кнопкой мышки на пункте **MEHighSpeedIO** и в открывшейся вкладке выберите пункт **General Configuration – PWM Setting – PWM0(OUT14)**. Поставьте флажок **Enable PWM0(OUT14)**:



Добавьте в программу команду **LS_PWM**:



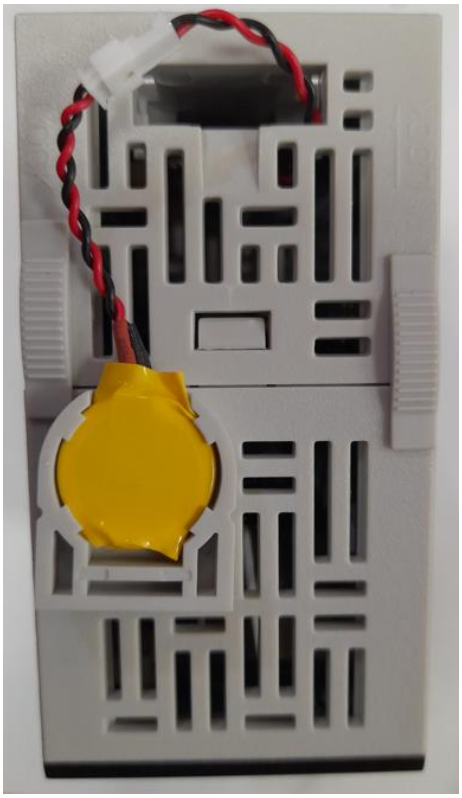
Команда **LS_PWM** должна обязательно стоять в программном блоке, привязанном к основной Задаче **HSIOTask**.

Описание ножек ФБ:
xEnable – запуск ФБ. ШИМ выдаётся пока есть TRUE
byIndex – номер выхода, 0 – OUT14, 1 – OUT15
diFrequency – частота ШИМ импульсов. Диапазон 100 Гц – 200 кГц
lRatio – скважность 0 -1 REAL. 0 – выход полностью закрыт, 1 – выход полностью открыт
xValid – флаг, что импульсы выдаются
xBusy – флаг, что ФБ в работе
xError – флаг ошибки
eErrorID – код ошибки

Замена батарейки часов реального времени

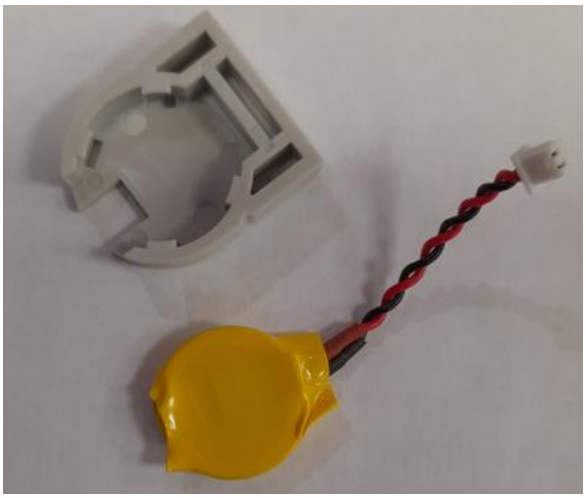
В контроллерах типа ME204-C используется батарейка CR1632 с коннектором. Идёт в комплекте с контроллером. Срок службы примерно 3 года. Используется только для работы часов реального времени (для поддержания памяти не требуется). Если Вы не используете часы реального времени в контроллере, то наличие батарейки не требуется.

Отсек с батарейкой находится на верхней плоскости контроллера в специальном держателе, который легко выдвигается наружу:



Коннектор со стороны батарейки отсоединяется от коннектора в корпусе контроллера:







Примечание:

Данную позицию можно найти в магазинах по названию:

Батарейка CMOS CR1632 с коннектором

(Разъём питания шаг 1.25 мм, 2 контакта)

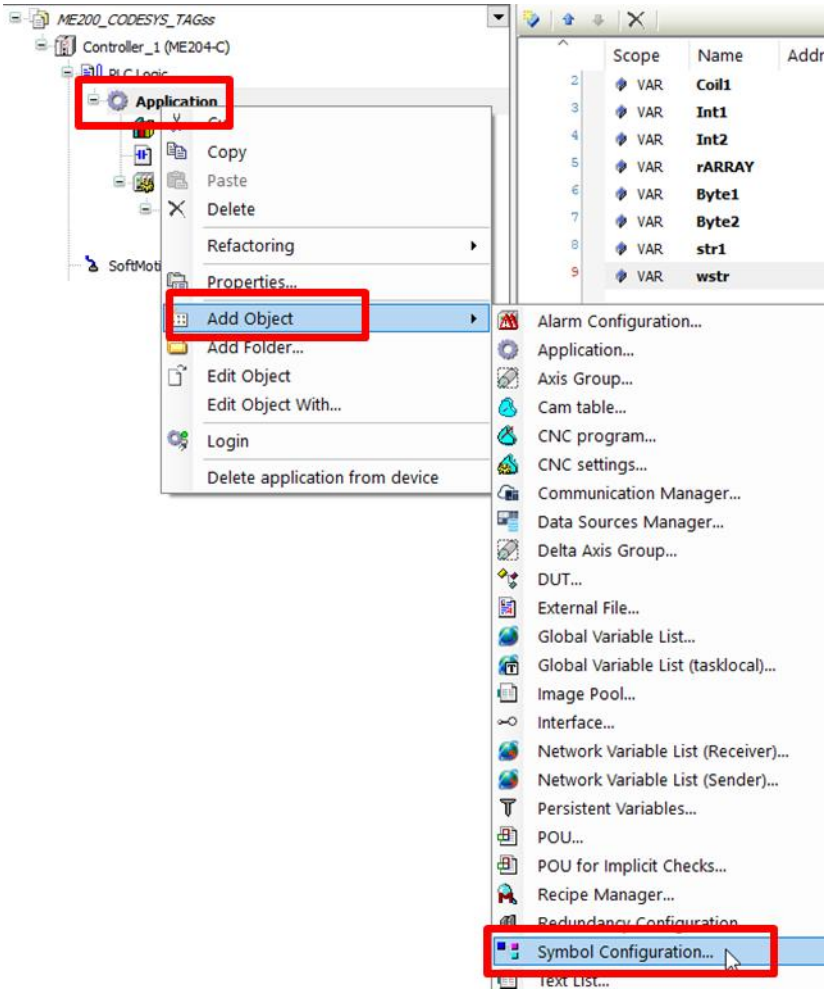
Обмен тегами CODESYS с панелью оператора

В среде программирования CODESYS существует стандартная процедура экспорта тегов в виде XML файла для дальнейшего использования в панели оператора или SCADA, которые поддерживают данный протокол связи с устройствами CODESYS. Панель (SCADA) будет выступать в роли Мастера (Клиента), а контроллер в роли Ведомого (Сервера).

Откройте любой проект для контроллера ME204-C, в котором есть созданные переменные.

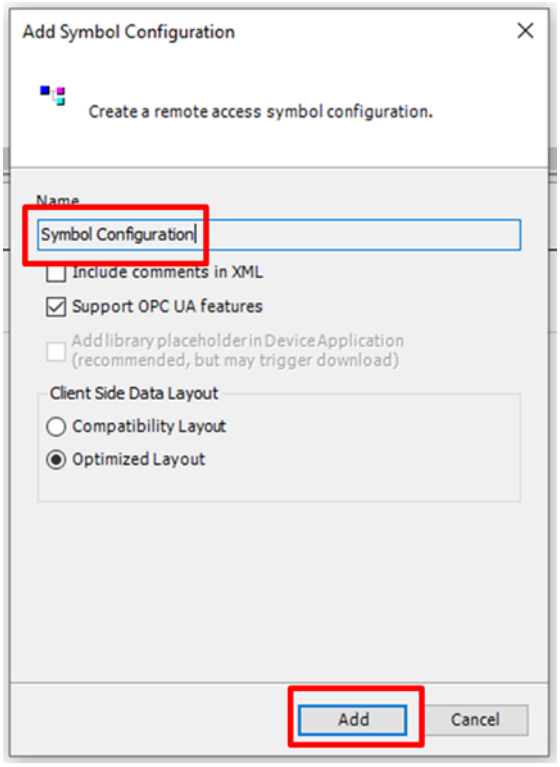
	Scope	Name	Address	Data type
2	VAR	**Coil1**		BOOL
3	VAR	**Int1**		INT
4	VAR	**Int2**		INT
5	VAR	**rARRAY**		ARRAY[0..2999] OF REAL
6	VAR	**Byte1**		BYTE
7	VAR	**Byte2**		BYTE
8	VAR	**str1**		string(20)
9	VAR	**wstr**		wstring(20)

Щёлкните правой кнопкой мышки на вкладки **Application** и в открывшемся меню выберите пункт **Add Object** и в следующем меню пункт **Symbol Configuration**.

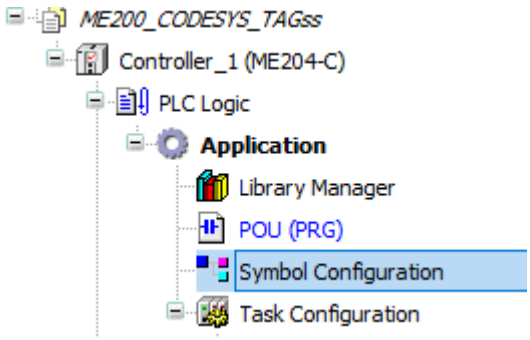


Внимание: ***В составе одного приложения (Application) может быть только одна таблица символов.***

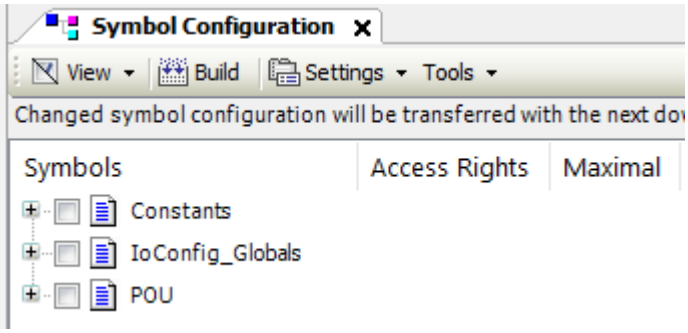
В появившемся окне определите название и нажмите **Add**:



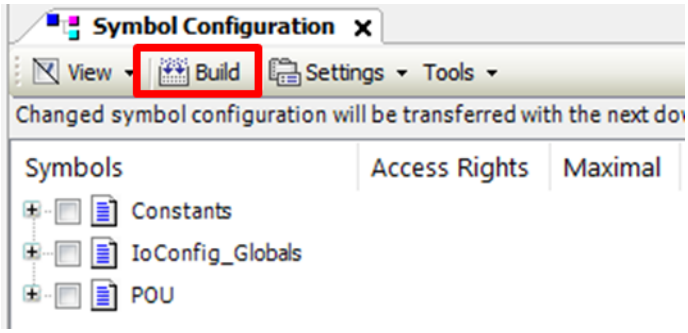
В разделе Application будет создан пункт с этим названием (в нашем примере используется название по умолчанию **Symbol Configuration**):



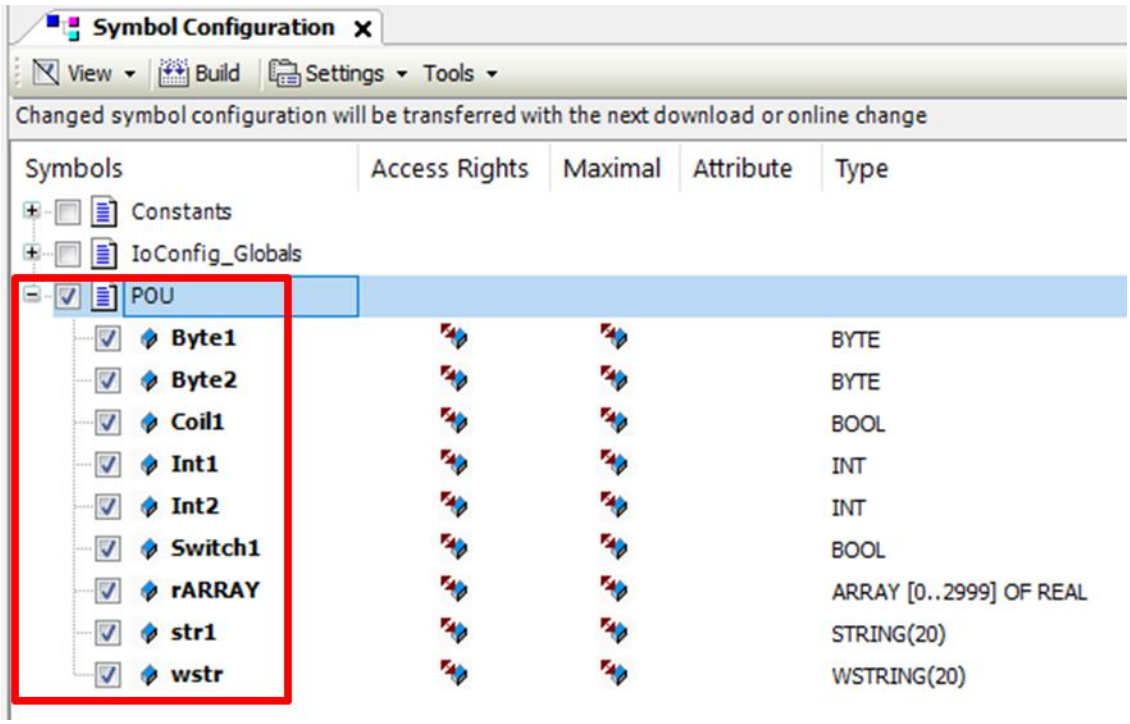
Также система сразу откроет одноименную вкладку:



Нажмите кнопку **Build** для построения проекта и формирования общего списка переменных:

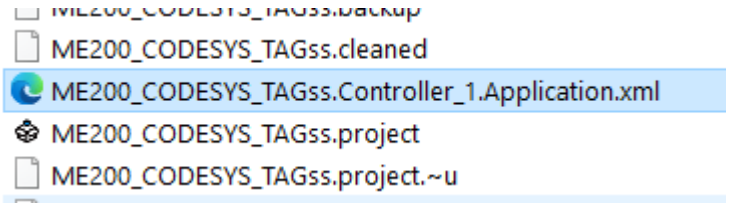


После компиляции отметьте флажками необходимы переменные:



Нажмите ещё раз кнопку **Build.**
В папке проекта будет создан XML файл вида:

ME200_CODESYS_TAGss.Controller_1.Application.xml



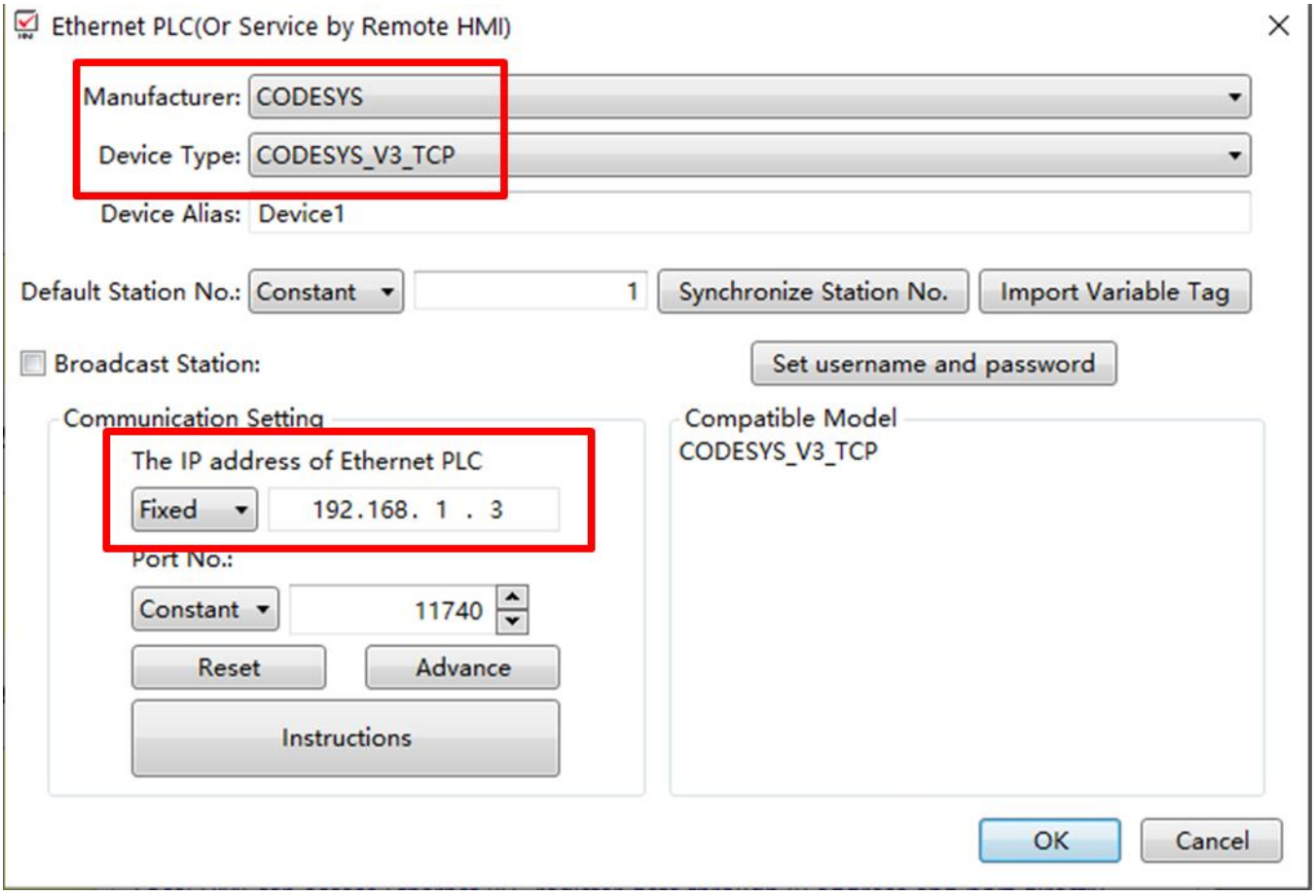
В созданном XML файле будут содержаться выбранные переменные:

```
  <NodeList>
    <Node name="Application">
      <Node name="POU">
        <Node name="Byte1" type="T_BYTE" access="ReadWrite" />
        <Node name="Byte2" type="T_BYTE" access="ReadWrite" />
        <Node name="Coil1" type="T_BOOL" access="ReadWrite" />
        <Node name="Int1" type="T_INT" access="ReadWrite" />
        <Node name="Int2" type="T_INT" access="ReadWrite" />
        <Node name="rARRAY" type="T_ARRAY__0__2999__OF_REAL" access="ReadWrite" />
        <Node name="str1" type="T_STRING_20_" access="ReadWrite" />
        <Node name="Switch1" type="T_BOOL" access="ReadWrite" />
        <Node name="wstr" type="T_WSTRING_20_" access="ReadWrite" />
      </Node>
    </Node>
  </NodeList>
</Symbolconfiguration>
```

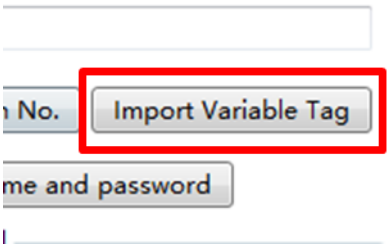
Загрузите проект в контроллер, чтобы таблица с тегами стала доступна для Мастера (Клиента)!

Данный XML файл можно импортировать в среду разработки проекта панели оператора или SCADA, которые поддерживают технологию связи через теги CODESYS. Для примера рассмотрим панель оператора Optimus Drive OTP210-070E, среда разработки Optimus Drive HMI Soft 3.

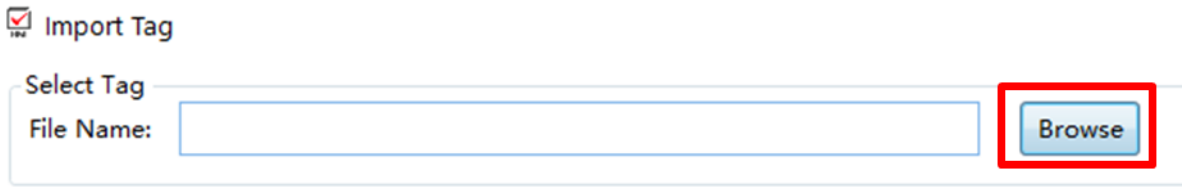
Создайте проект для панели OTP210-070E и в разделе настройки связи выберите Ethernet PLC и драйвер CODESYS. Задайте IP адрес (в нашем примере 192.168.1.3):



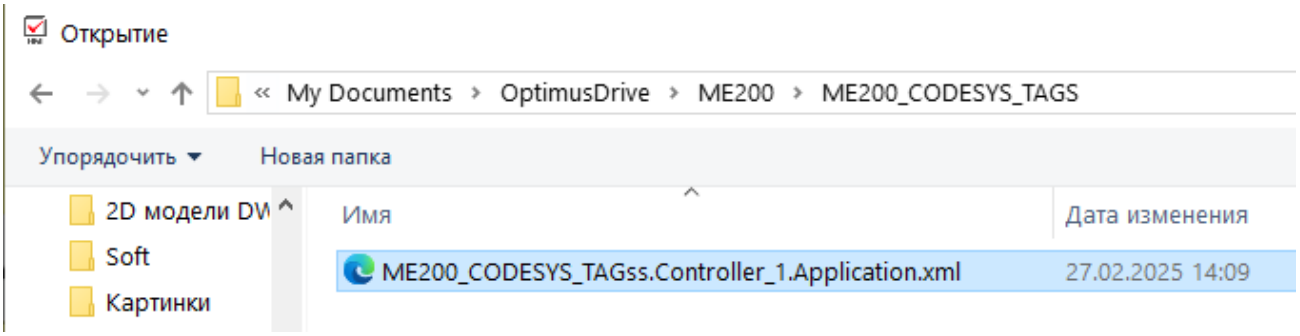
Нажмите кнопку **Import Variable Tag** и импортируйте XML файл с тегами:



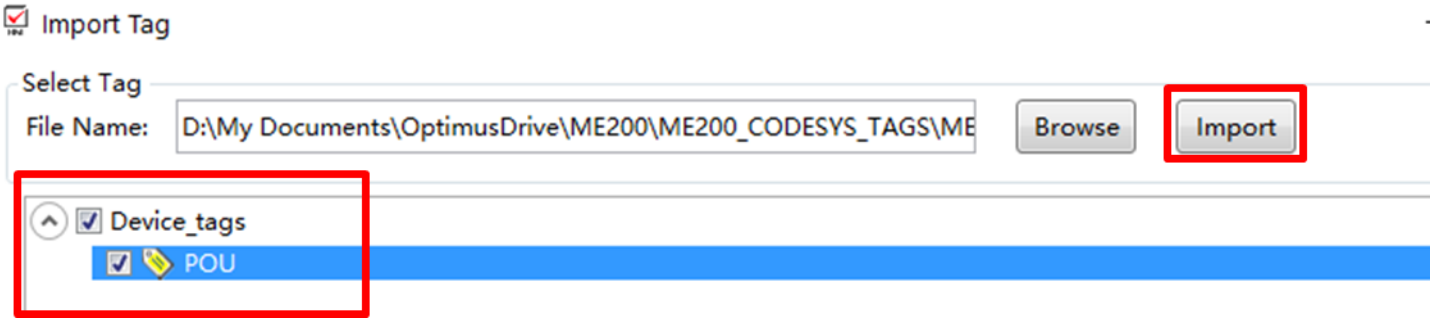
В открывшемся окне нажмите кнопку **Browse**:



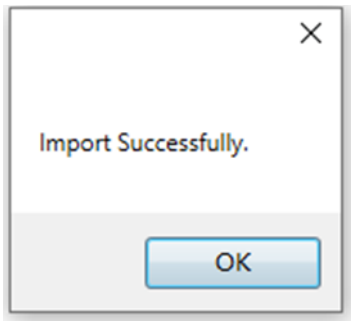
выберите XML файл:



Разверните список тегов и поставьте флажки, затем нажмите кнопку **Import**:

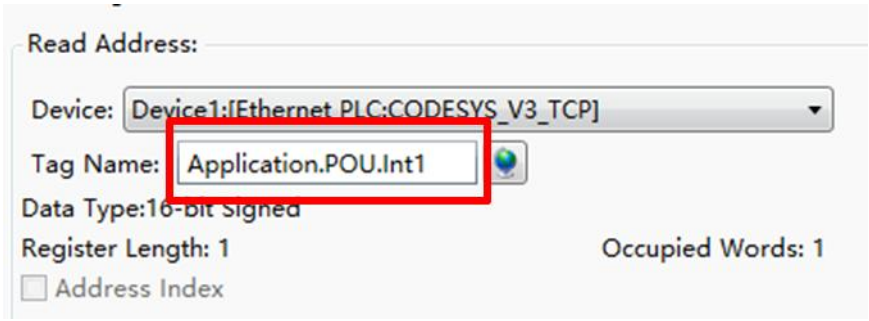


При удачном импорте тегов появится окошко с надписью **Import Successfully**:

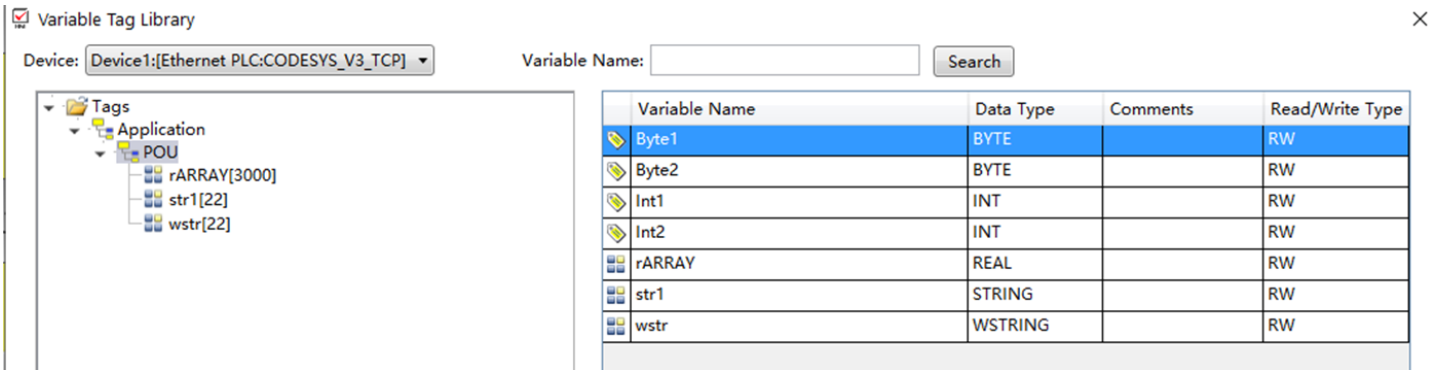


Нажмите **ОК**.

Теперь теги будут доступны как адреса у экранных объектах:

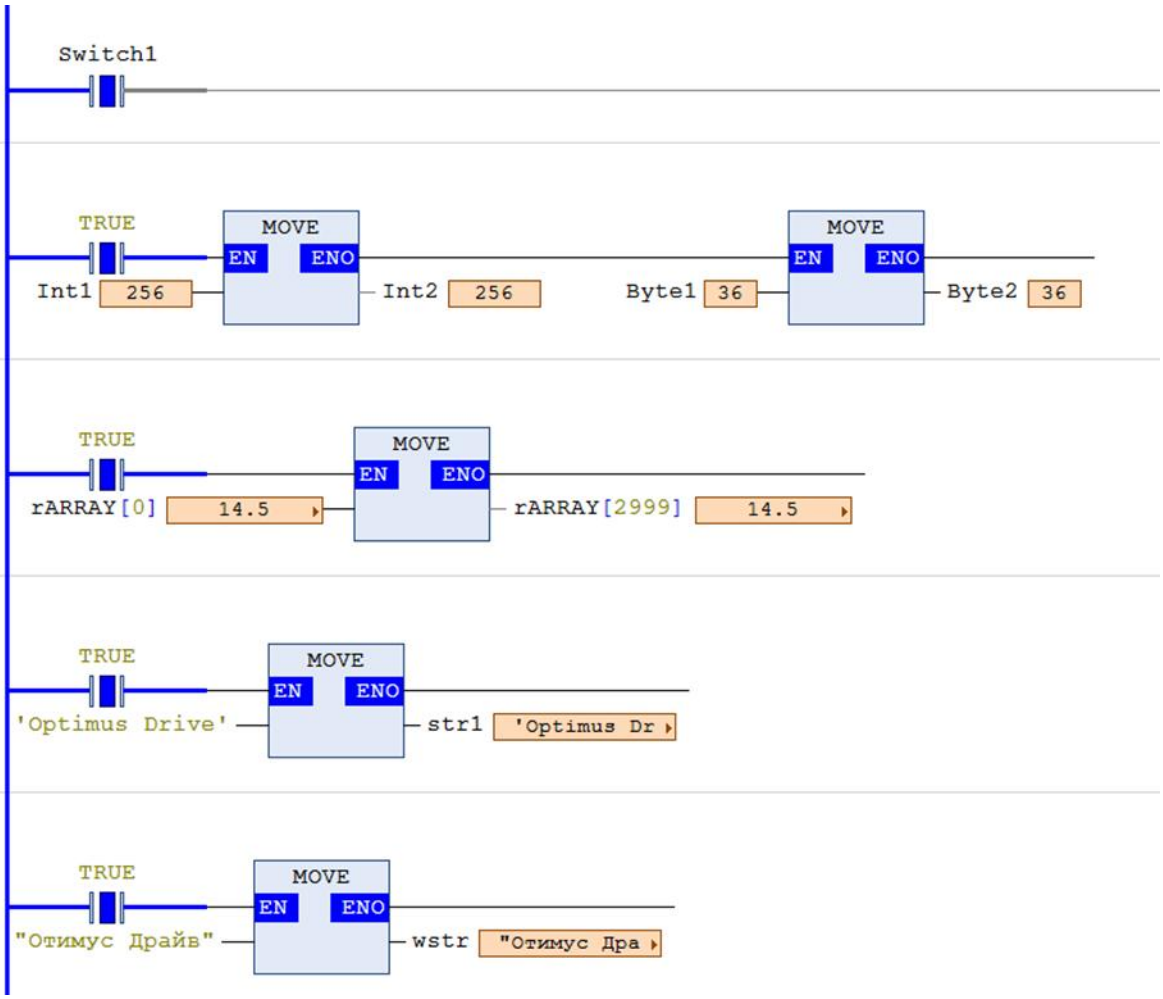


Создайте необходимое количество экранных объектов, указывая в качестве адресов импортированные теги:



Загрузите в панель или запустите онлайн симулятор. На экране будут отображаться те же данные, что и в программе контроллера.


256
256
14.450
14.450
Кнопка
Лампа
36
36
Optimus Drive
Отимус Драйв


Switch1
TRUE
MOVE
EN
ENO
Int1
256
Int2
256
MOVE
EN
ENO
Byte1
36
Byte2
36
TRUE
MOVE
EN
ENO
rARRAY[0]
14.5
rARRAY[2999]
14.5
TRUE
MOVE
EN
ENO
'Optimus Drive'
str1
'Optimus Dr
TRUE
MOVE
EN
ENO
"Отимус Драйв"
wstr
"Отимус Дра