Контроллеры промышленные серии MX300

Руководство по эксплуатации

Редакция от марта 2025 года

optimusdrive.ru

Оглавление

Введение

Настоящее Руководство описывает порядок и особенности использования контроллеров серии MX300 в среде программирования DesignerAX 1.7 и выше или CODESYS 3.5.18.30 и содержит информацию по техническим характеристикам контроллеров и модулей расширения. 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 оси сервопривода, оси энкодера, виртуальной оси, импульсной оси. Работы с портами по различным протоколам связи. Применение станции удалённого ввода-вывода R2EC по сети EtherCAT.

Редакция от марта 2024 года содержит следующие сведения:

- технические характеристики контроллеров и модулей расширения
- установка конфигурационных файлов
- подключение контроллера к ПК
- работа с модулями ввода-вывода

В Редакцию от июня 2024 года добавлены следующие сведения:

- Применение штурвального энкодера в качестве мастер-оси
- Работа с импульсными входами-выходами (счётчики, импульсные оси)
- Создание виртуальной оси
- Последовательная связь по протоколу Modbus RTU Master/Slave
- Связь по протоколу Modbus TCP Master/Slave
- Работа с преобразователем частоты серии Optimus Drive AD800 по сети EtherCAT

В Редакцию от августа 2024 года добавлены следующие сведения:

- Работа по протоколу Ethernet/IP
- Чтение и установка часов реального времени

В Редакцию от ноября 2024 года добавлены следующие сведения:

- Работа с температурными модулями
- Добавлены таблицы с EtherCAT адресами для модулей расширения
- Работа со станцией удалённого ввода-вывода (каплером) R2EC по сети EtherCAT
- Работа по протоколу OPC UA
- Копирование файлов на SD карту

В Редакции от апреля 2025 года приведены в соответствие с новым пэкиджем **OD M Series Pack 1.0.3.10.package** (и выше версией) описания конфигурационных форм и процедур.

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

Меры предосторожности при эксплуатации

Контроллеры семейства MX300 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

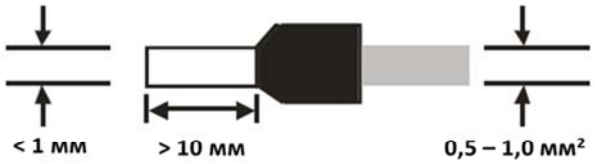
Меры предосторожности при подключении и работе

⚠ 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 MX300 и модулях расширения установлены пружинные клеммники, которые позволяют производить монтаж проводов без наконечников. Рекомендуются строго гибкие (многопроволочные) провода сечением 0,5 – 1,0 мм².
В случае использования наконечников рекомендуются наконечники без юбки. Если Вы принципиально используете наконечники с юбкой, то длина рабочей части наконечника должна быть более 10 мм (оптимально 12 мм). Диаметр после обжима не более 1 мм.



Функциональное назначение

Приборы семейства MX300 являются многофункциональными контроллерами, предназначенными для решения широкого круга задач как общей автоматизации в промышленности, так и задач управления сложным движением по шине EtherCAT с большой скоростью процессов. Обладают высокими программными возможностями с поддержкой сложных вычислений, логики и операций с данными. Имеют встроенные порты Ethernet, EtherCAT, RS232, 2xRS485, mini USB. Поддерживают протоколы связи EtherCAT, Ethernet/IP, Modbus, OPC UA.

Для программирования контроллеров семейства MX300 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 и выше версией. 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;

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, аналоговых и температурных входов-выходов. Далее приводится перечень оборудования, входящего в состав системы на основе контроллеров MX300, 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)	MX308-CE	Контроллер, EtherCAT 8 осей движения, 6 импульсных осей 200 кГц, 16DI/16DO (NPN), Ethernet, RS232, 2xRS485, CAN, микро SD, USB-C, 24 VDC
	MX316-CE	Контроллер, EtherCAT 16 осей движения, 6 импульсных осей 200 кГц, 16DI/16DO (NPN), Ethernet, RS232, 2xRS485, CAN, микро SD, USB-C, 24 VDC
	MX332-CE	Контроллер, EtherCAT 32 осей движения, 6 импульсных осей 200 кГц, 16DI/16DO (NPN), Ethernet, RS232, 2xRS485, CAN, микро SD, USB-C, 24 VDC
Модули дискретных входов/выходов	MX16DI	24 В DC 5 мА 16 входов NPN/PNP Пружинный клеммный блок
	MX16DOP	5 ~ 30 В DC 0.5 А 16 выходов Выходы: PNP Пружинный клеммный блок
	MX16DOR	240 В AC / 24 В DC 2 А 16 выходов Выходы: Реле Пружинный клеммный блок
	MX16DOT	5 ~ 30VDC 0.5A 16 выходов Выходы: NPN Пружинный клеммный блок
Модули аналоговых входов-выходов	MX04AD	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
	MX04DA	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
Модули температурных входов	MX04TC	4-канальный температурный модуль под термопары Аппаратное разрешение: 16 бит
	MX04RC	4-канальный температурный модуль под термосопротивления Аппаратное разрешение: 16 бит
Станция удалённого ввода-вывода EtherCAT	R2EC	Использует те же модули расширения, что и ЦПУ
Патч-корды EtherCAT	CABLE-TX0M3-BUS CABLE-TX0M5-BUS CABLE-TX1M0-BUS CABLE-TX2M0-BUS CABLE-TX5M0-BUS CABLE-TX10M0-BUS CABLE-TX20M0-BUS	0.3 м 0.5 м 1 м 2 м 5 м 10 м 20 м

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	MX308-CE	MX316-CE	MX332-CE
Кол-во поддерживаемых осей	8 осей EtherCAT + импульсное управление 6 осями 200 кГц	16 осей EtherCAT + импульсное управление 6 осями 200 кГц	16 осей EtherCAT + импульсное управление 6 осями 200 кГц
Процессор	ARM Cortex-A9 Dual Core 866 МГц		
Макс. кол-во модулей расширения	32		
EtherNET	1* EtherNET port, Modbus TCP, Socket, загрузка и выгрузка программы, отладка		
EtherCAT	EtherCAT master, поддержка до 128 ведомых устройств		
Порт последовательной связи	RS232, 2xRS485, пользовательский протокол, MODBUS RTU Ведущий/Ведомый		
Поддержка устройств по шине CANopen	до 31 Ведомого		
Память под программу	20 Мб		
Память данных	40 Мб		
Энергонезависимая память	512 Кбайт		
Порт USB	Тип С, загрузка и выгрузка программы, отладка		
Слот карты памяти	карта micro SD card, FAT32, до 32 Гб, для переноса программы пользователя		
Управление движением	Точка-точка, электронный кулачок (E-CAM), интерполяция		
Высокоскоростные счетчики	6 входов AB (200 кГц)		
Встроенные входы/выходы	16 входов (12 входов 200 кГц/4 входа 1 кГц(NPN/PNP)) 16 выходов (12 выходов 200 кГц/4 выхода 10 кГц (NPN))		
Часы реального времени	Да (встроенная батарейка CR2032 с выводами и разъёмом)		
Среда разработки	Designer-AX 1.5, CODESYS 3.5.18.30		
Языки программирования	ST, LD, CFC, SFC, FBD		
Библиотеки	SM3_Basics/Robotics/CNC v.4.10		
Напряжение питания	24 В постоянного тока		
Потребляемая мощность	3.6 Вт		
Рабочая температура	0~55□		
Габаритные размеры	ВхШхГ: 100.00 x 81.75 x 98,5 мм		

Спецификация дискретных входов-выходов на ЦПУ

Спецификация дискретных входов:

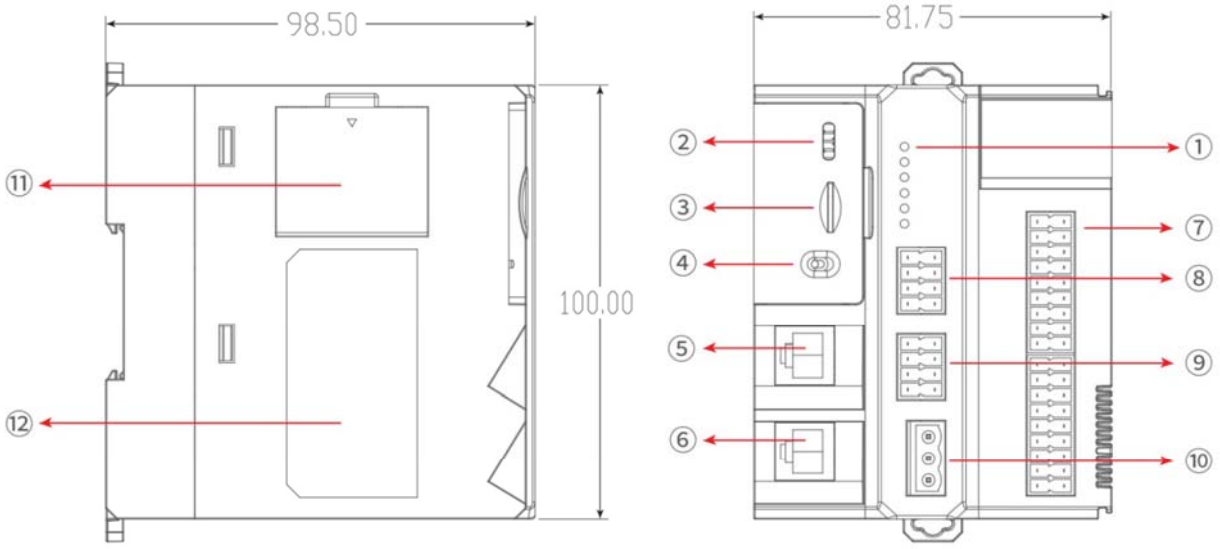
Параметр / Модель		MX308-CE MX316-CE MX332-CE
Количество входов		16
Тип соединения		Съёмный пружинный клеммник
Тип входа		Дискретный вход
Форма входа		Постоянный ток (NPN/PNP, две общие точки SS0/SS1) SS0 для IN0~IN7,SS1 для IN8~IN15
Напряжение/ ток		24 VDC, 5 mA
Уровень вкл/выкл	OFF→ON	>15 VDC
	ON→OFF	<5 VDC
Максимальная входная частота		200 KHz
Входное сопротивление		4.7 kΩ обычные входы, 3.3 kΩ импульсные входы
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

Спецификация дискретных выходов:

Параметр / Модель		MX308-CE MX316-CE MX332-CE
Количество выходов		16
Тип соединения		Съёмный пружинный клеммник
Тип выхода		NPN (Sinking)
Напряжение		5~30 VDC
Макси-мальная нагрузка	Активная	0.5A/выход
	Индуктивная	-
	Лампочка	-
Максимальная выходная частота		200 KHz
Общая точка COM		Одна общая точка на все выходы, 2.4A/COM

Внешний вид и размеры ЦПУ

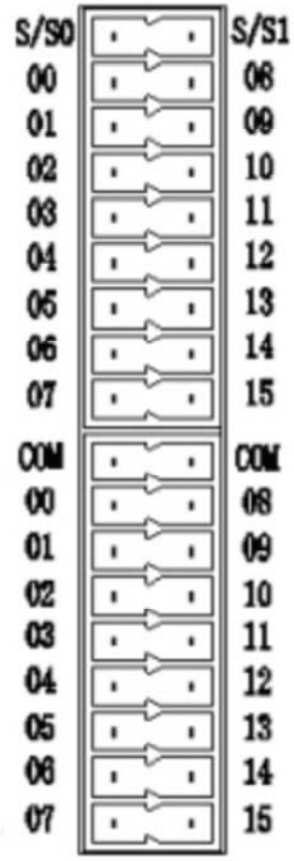
MX308-CE/MX316-CE/MX332-CE



1 Светодиоды:
POWER: Питание (зеленый), SYS: Индикатор работы (зеленый), RUN: шина EtherCAT (зеленый), ERR: ошибка сети EtherCAT (красный)
2 Порт USB Type-C
3 Карта Micro SD
4 Переключатель RUN/STOP
5 Шина EtherNet: OPC UA, EtherNet/IP, MODBUS TCP
6 Шина EtherCAT
7 Встроенные входы/выходы
8 Порты последовательной связи 2xRS485
9 Порты CANopen и RS232
10 Разъем питания
11 Батарейный отсек
12 Шильдик

Расположение клемм ЦПУ

MX308-CE/MX316-CE/MX332-CE



Входы 00 – 11 являются высокоскоростными, входы 12 – 15 являются обычными

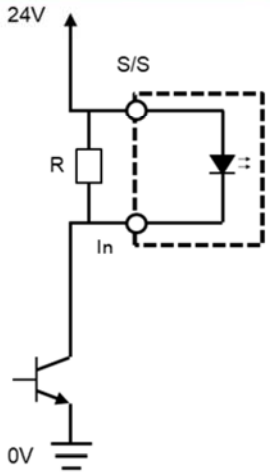
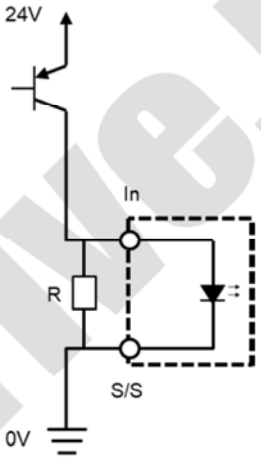
Выходы 00 – 11 являются высокоскоростными, выходы 12 -15 обычными
(нечётный выход – импульсы, чётный – направление)

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).

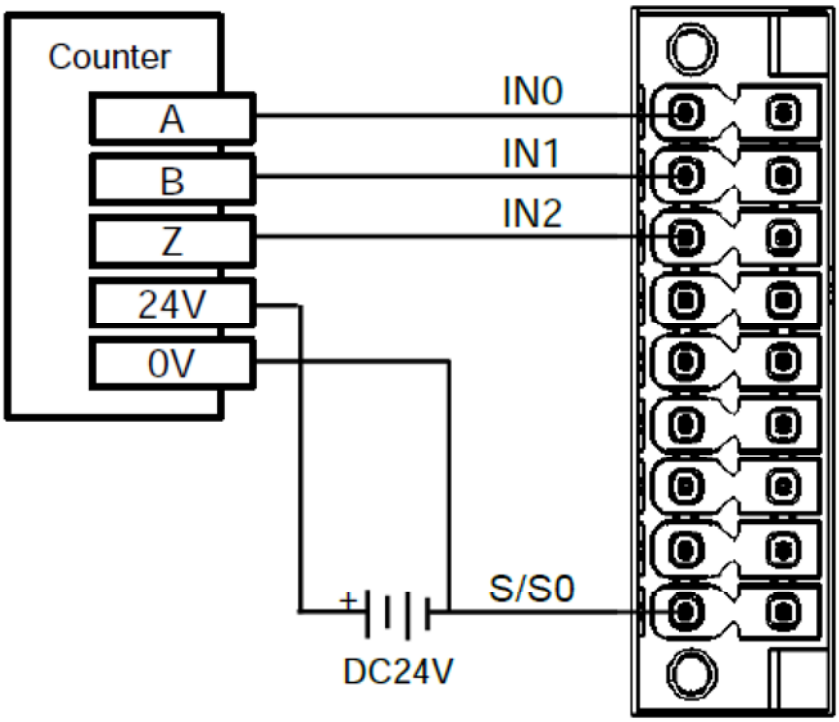
Схемы подключения MX308-CE/MX316-CE/MX332-CE

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.

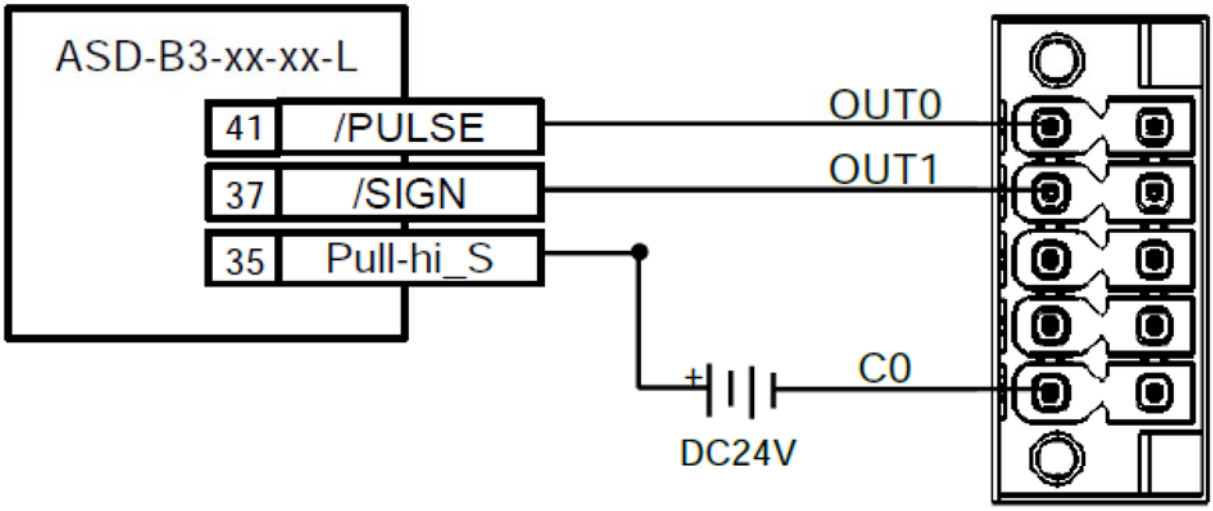
NPN (общий плюс)	PNP (общий минус)
	

Подключение входов в режиме высокоскоростного счётчика

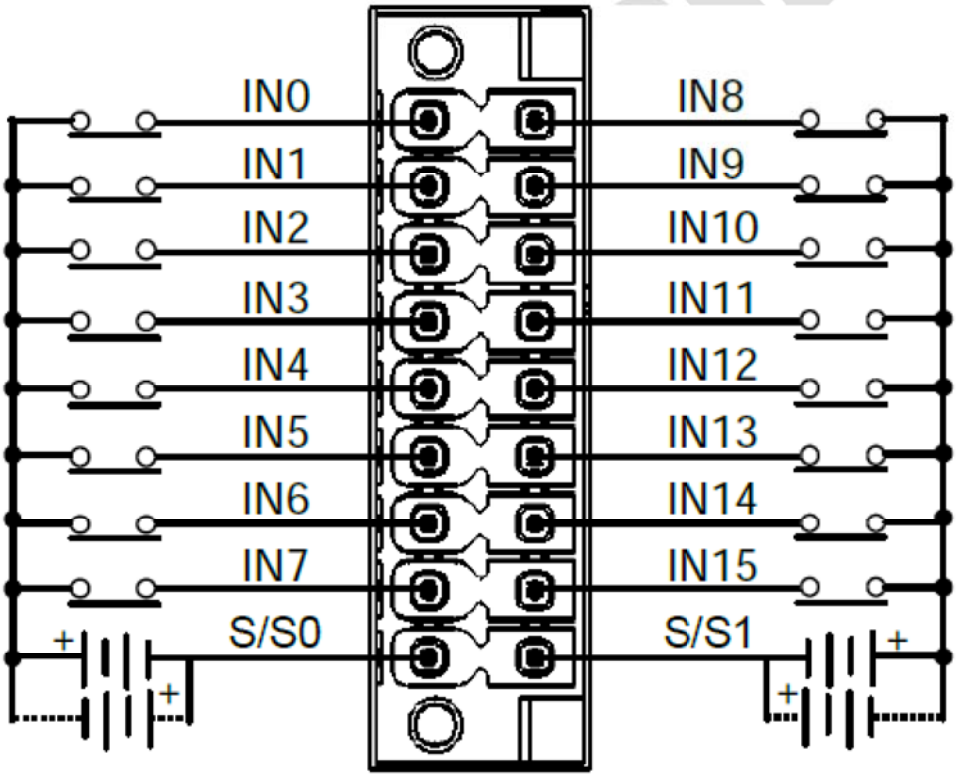


Подключение выходов в импульсном режиме

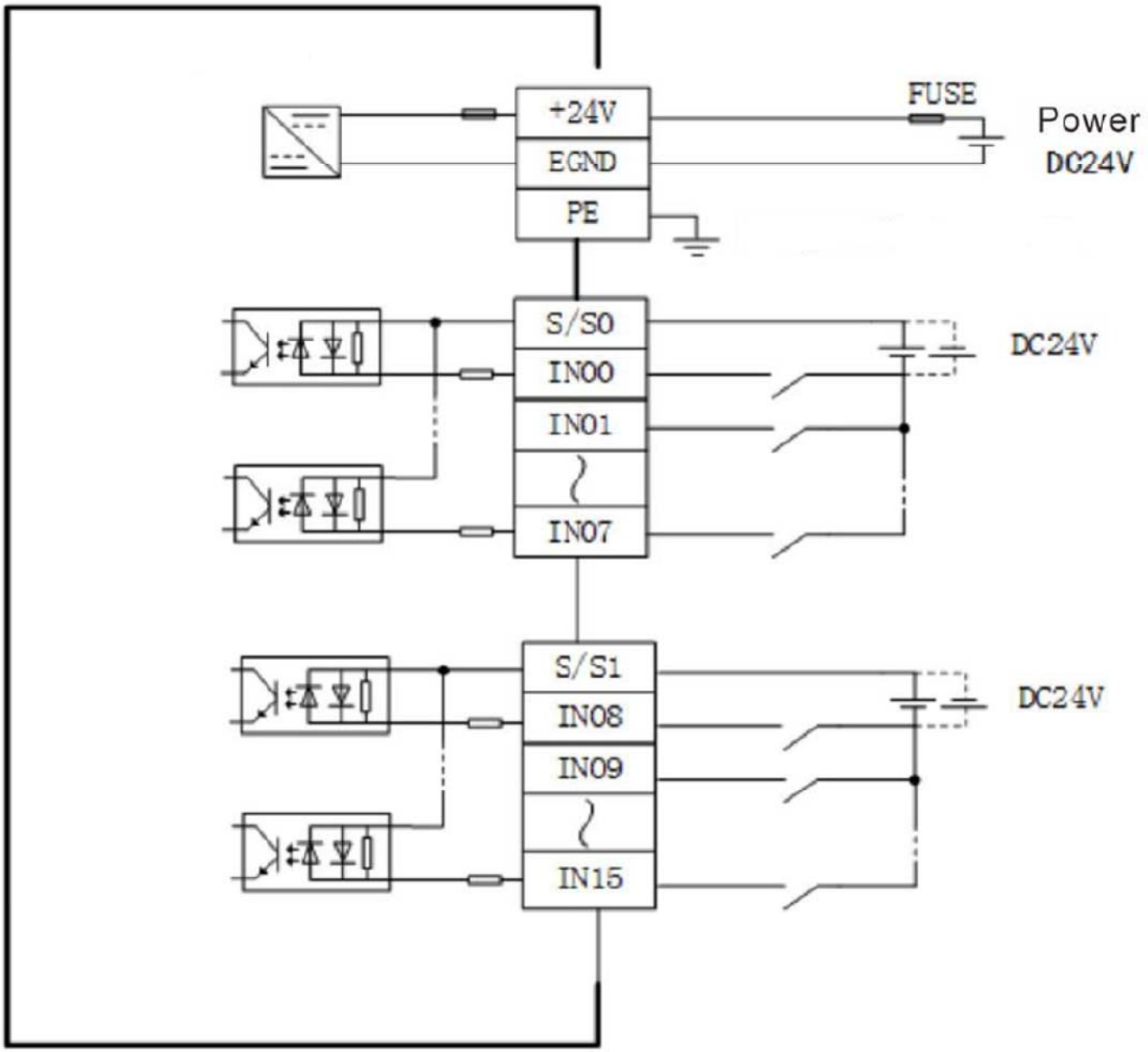
(на примере выдачи импульсов на сервопривод Delta ASD)



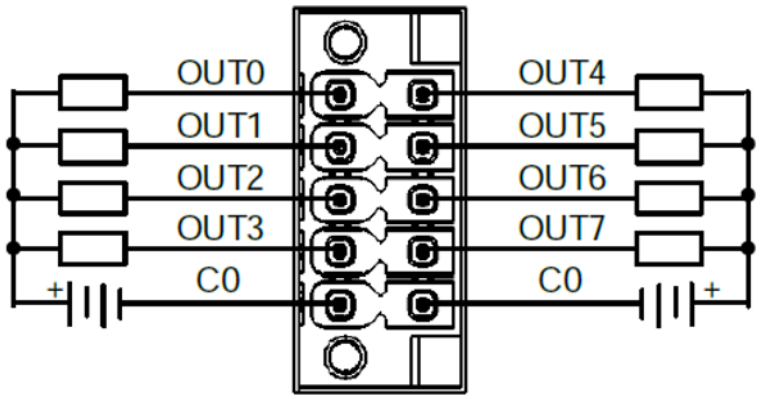
Подключение входов в обычном режиме



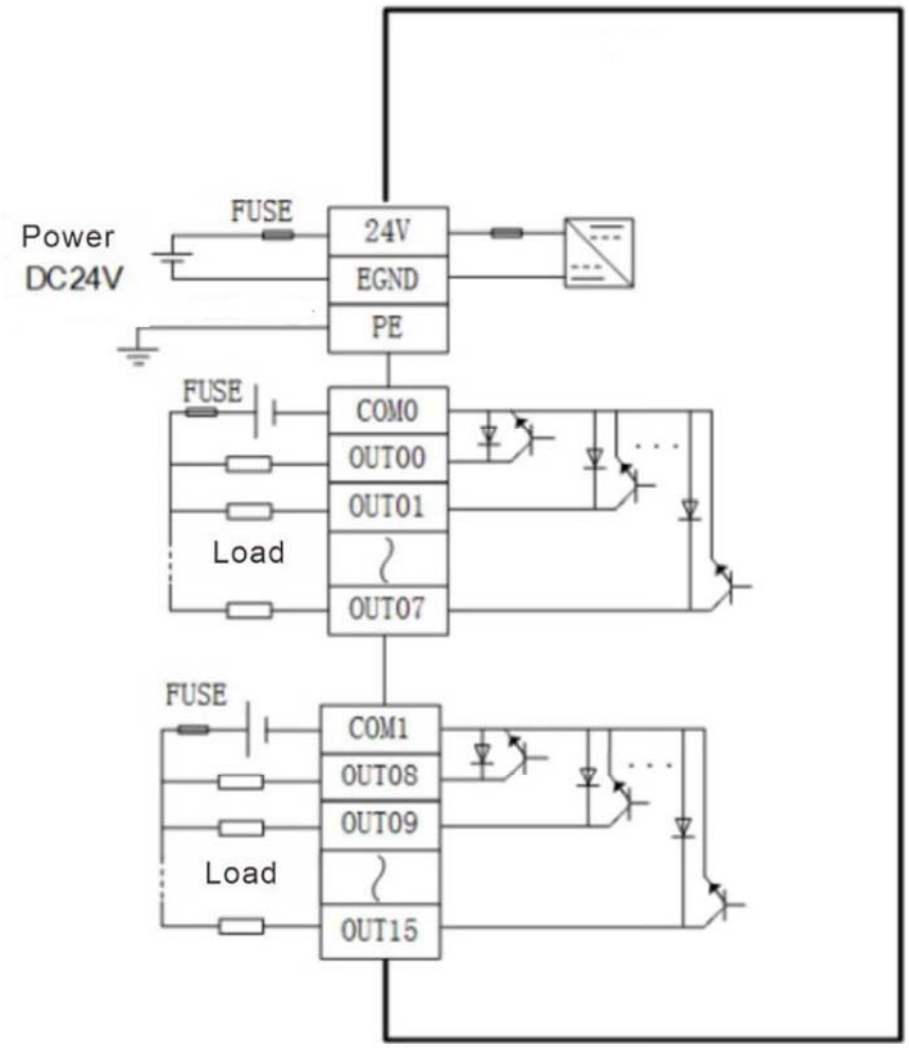
Эквивалентная схема входов:



Подключение выходов типа NPN в обычном режиме



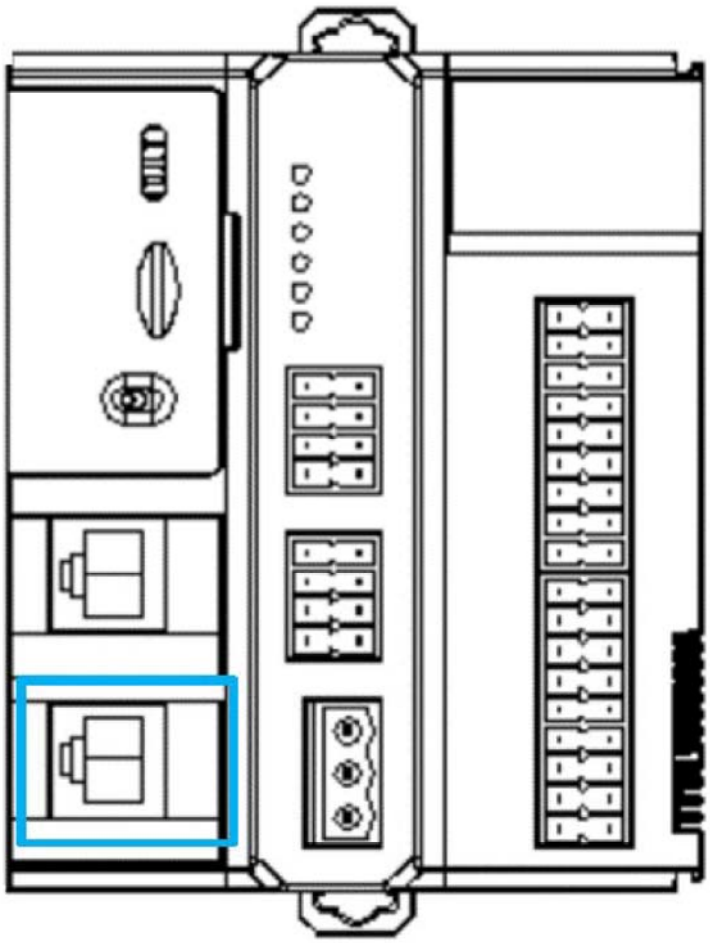
Эквивалентная схема выходов:



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).

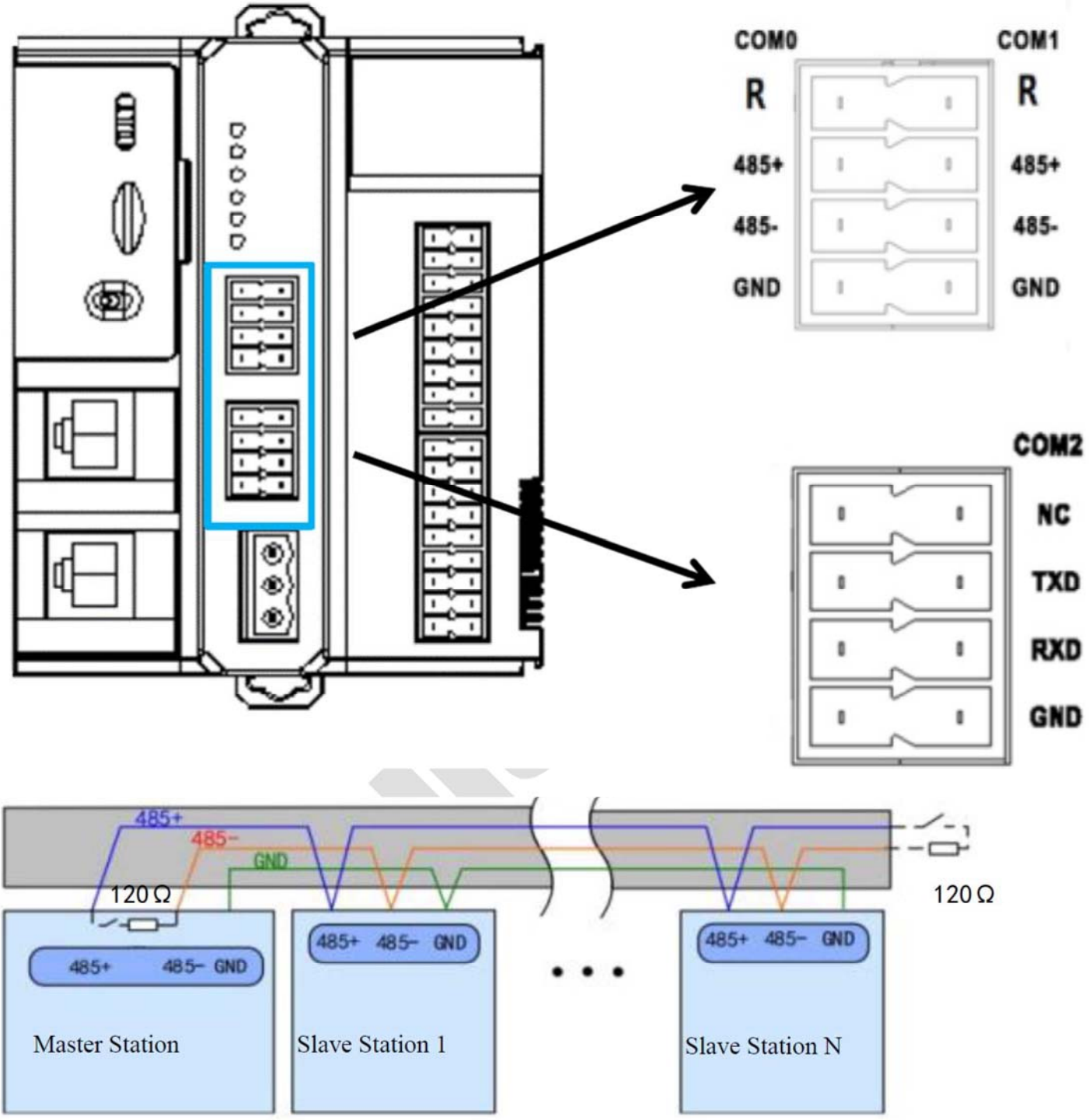
Расположение интерфейсов MX308-CE/MX316-CE/MX332-CE

EtherCAT

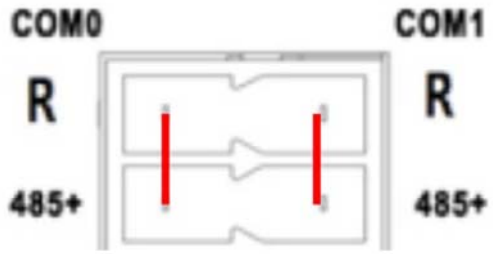


No.	Наименование	Описание
1	TX data+	Send data+
2	TX data-	Send data-
3	RX data+	Receive data+
4	/	/
5	/	/
6	RX data-	Receive data-
7	/	/
8	/	/

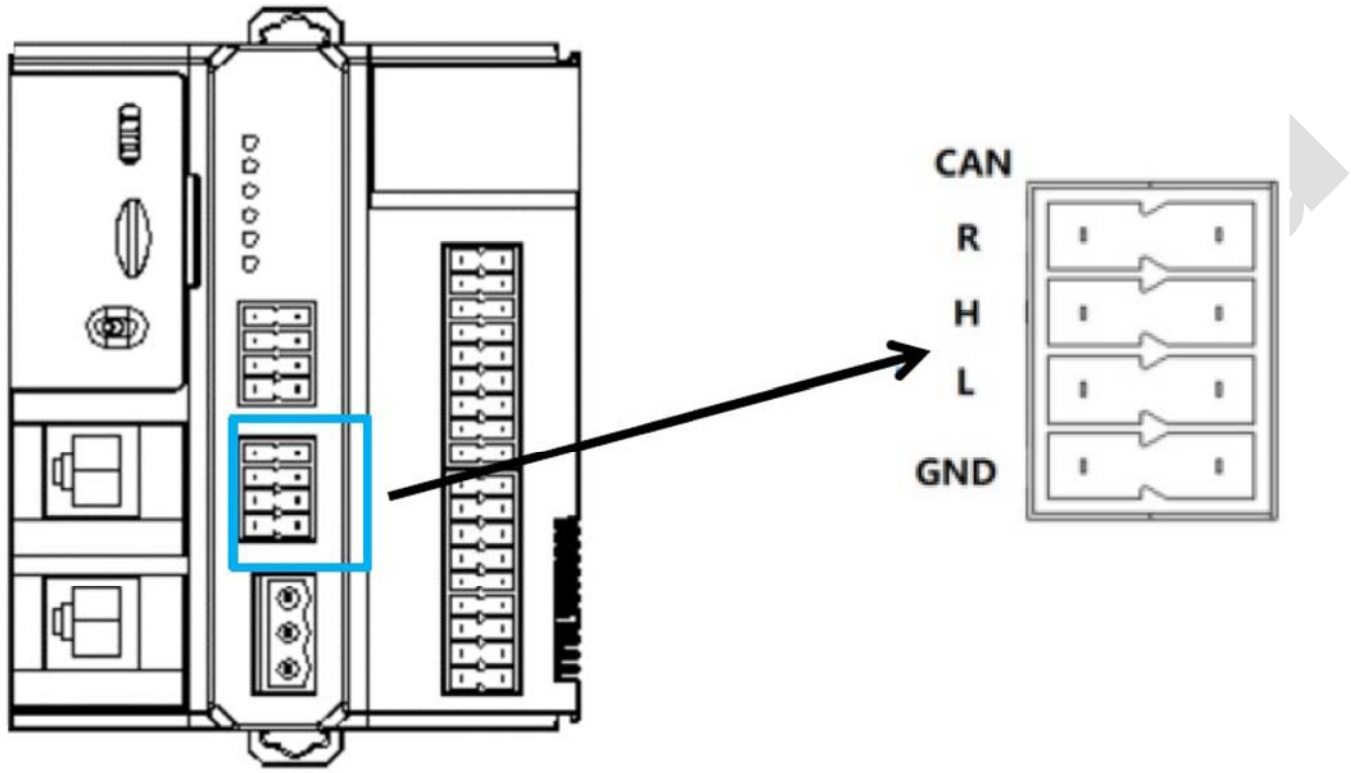
RS232/RS485

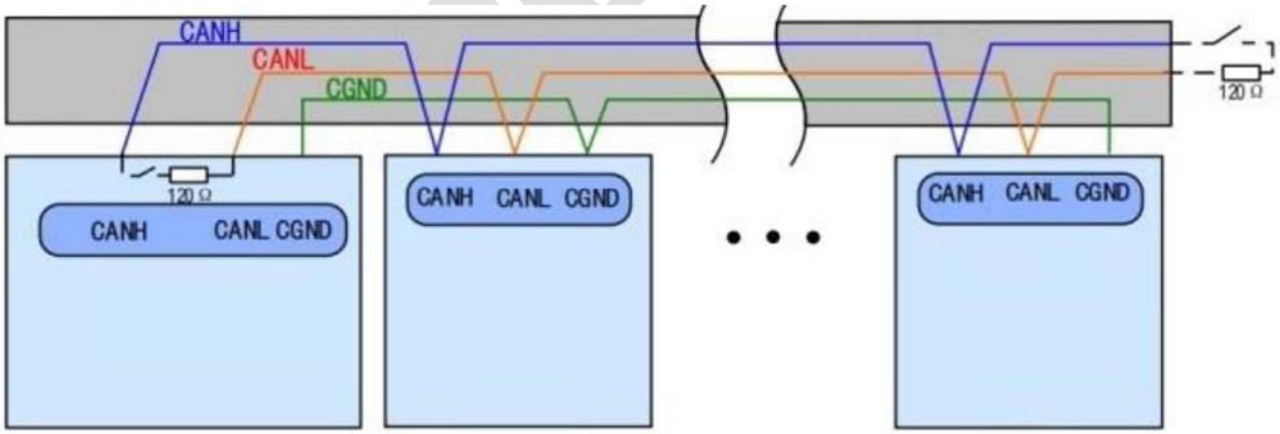


ЦПУ содержит встроенные резисторы 120 Ом для портов RS485 COM0 и COM1. Для подключения резистора необходимо перемкнуть перемычкой клеммы 485+ и R.

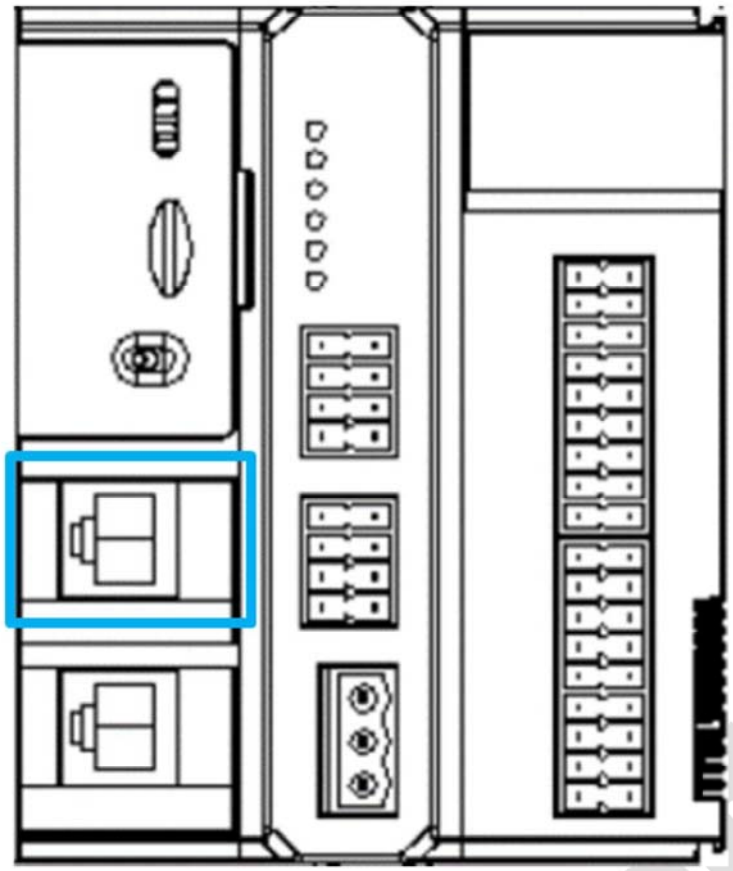


CAN Bus

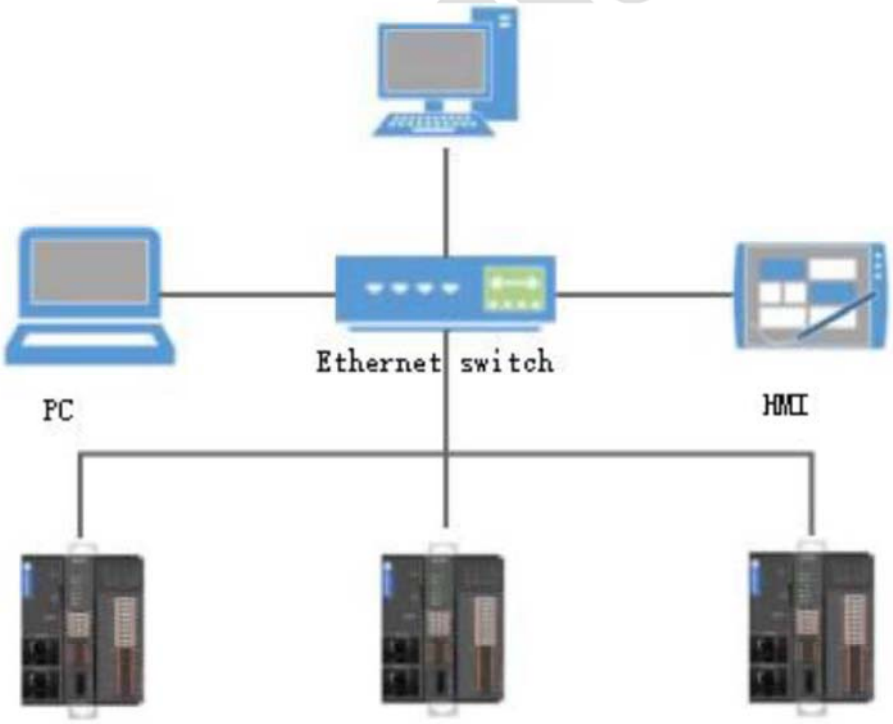

CAN
R
H
L
GND


CANH
CANL
CGND
120 Ω
120 Ω
CANH CANL CGND
CANH CANL CGND
CANH CANL CGND

Ethernet

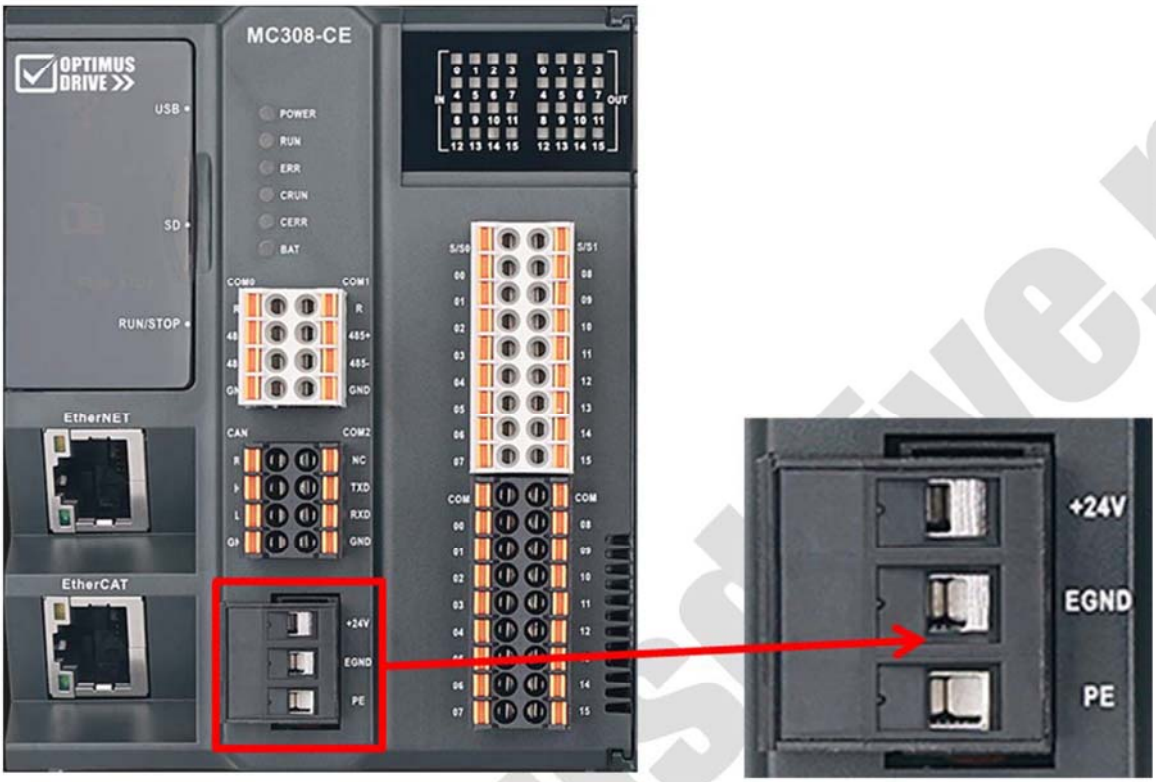


Контроллер может быть подключен к сети Ethernet напрямую или через коммутатор.



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.



Клемма	Назначение
+24V	Плюс источника питания 24 VDC
EGND	Минус источника питания 24 VDC
PE	Защитное заземление линии блоков питания

Установка модулей расширения

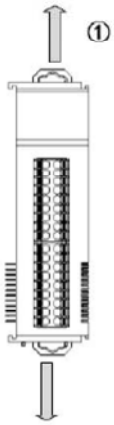
Модули расширения устанавливаются по направляющим вплотную к ЦПУ и далее модуль к модулю. Благодаря такому монтажу, для смены модуля не требуется раздвигать всю сборку. Достаточно просто откинуть разъём извлечь старый модуль вставить новый и установить обратно разъём с проводами. Для монтажа сигнальных проводов на модулях используются съёмные клеммники с пружинными зажимами.

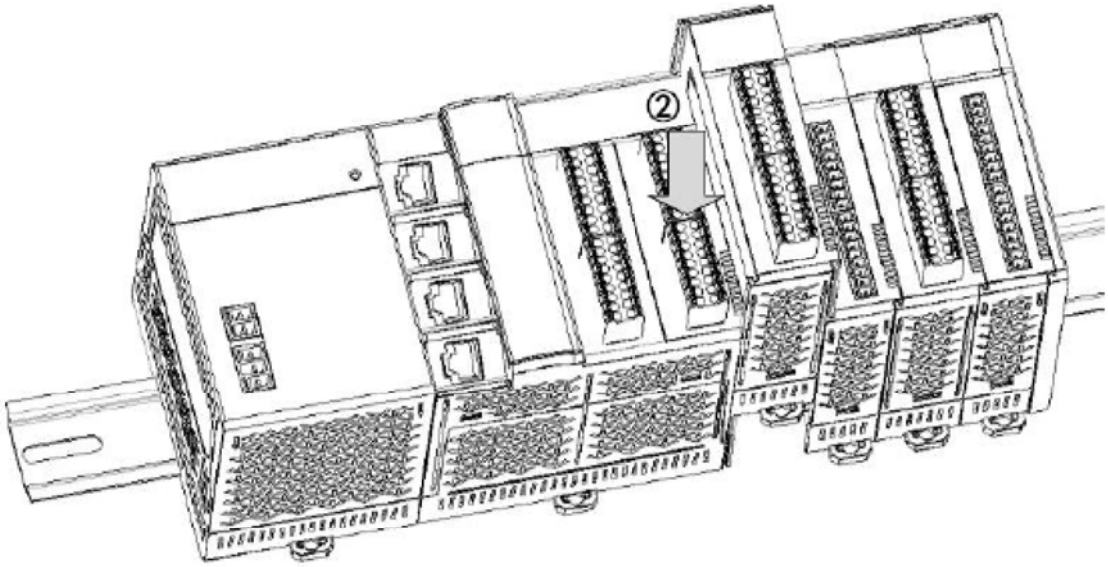
На модулях, как и на ЦПУ, используется удобная двойная защёлка для монтажа на ДИН-рейку. Для установки модуля необходимо:

Раздвинуть фиксаторы вверх и вниз соответственно

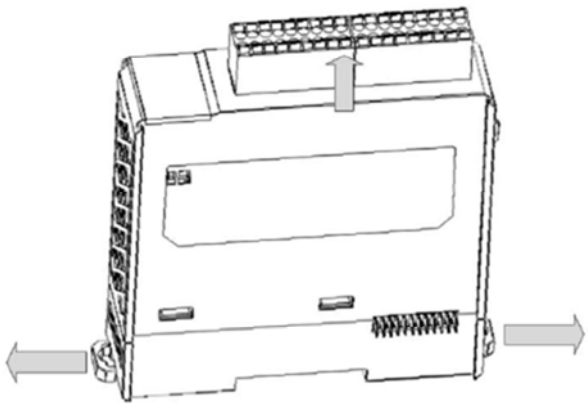
Вставить модуль по направляющим до упора на ДИН-рейку

Защёлкнуть оба фиксатора (сверху и снизу)



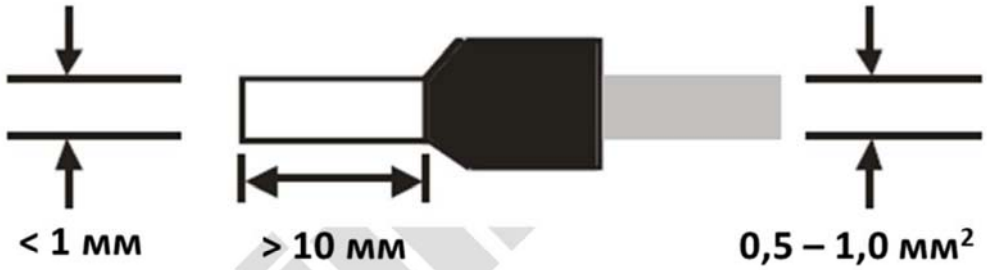


Для извлечения модуля необходимо раздвинуть фиксаторы и потянуть модуль на себя:



Рекомендации по использованию проводов и наконечников

На контроллерах MX300 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 10 мм (оптимально 12 мм). Диаметр после обжима не более 1 мм.



Спецификация модулей дискретных входов-выходов

Спецификация дискретных входов на модулях расширения:

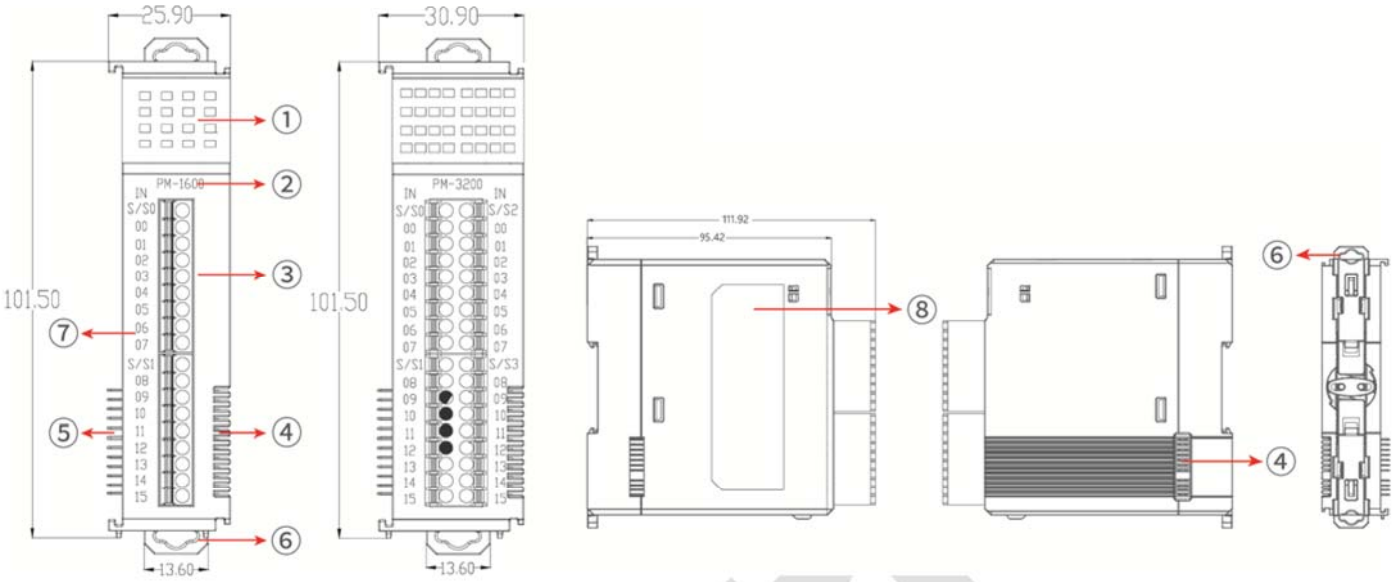
Параметр \ Модель		MX16DI
Количество входов		16
Тип соединения		Съёмный пружинный клеммник
Тип входа		Дискретный вход
Форма входа		Постоянный ток (NPN/PNP, две общие точки SS0/SS1) SS0 для IN0~IN7,SS1 для IN8~IN15
Напряжение/ ток		24 VDC, 5 mA
Уровень вкл/выкл	OFF→ON	>15 VDC
	ON→OFF	<5 VDC
ON/OFF		20мкс/50мкс
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: 100.00 x 25.90 x 98.5 мм

Спецификация дискретных выходов на модулях расширения:

Параметр \ Модель		MX16DOT	MX16DOP	MX16DOR
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: 100.00 x 25.90 x 98.5 мм		

Внешний вид и размеры дискретных модулей расширения

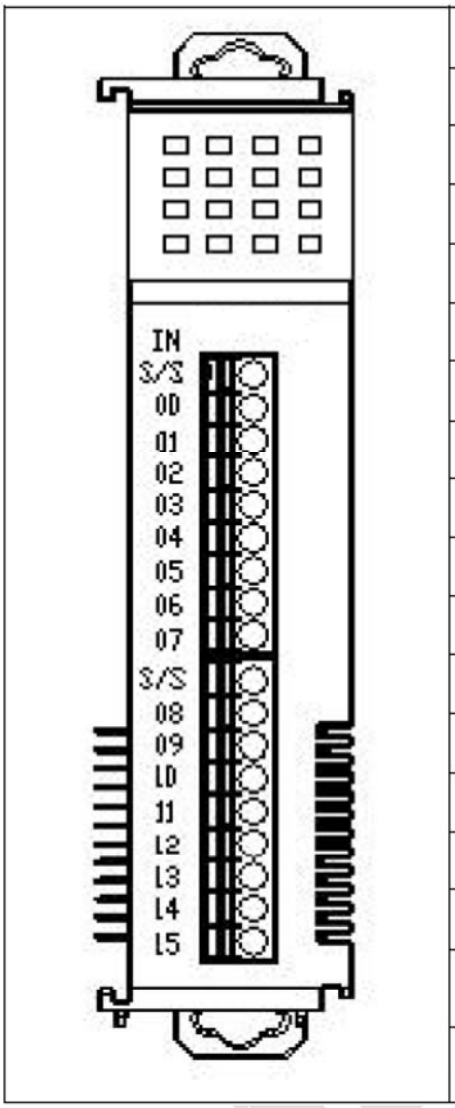
MX16DI/MX16DOT/MX16DOP/MX16DOR



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

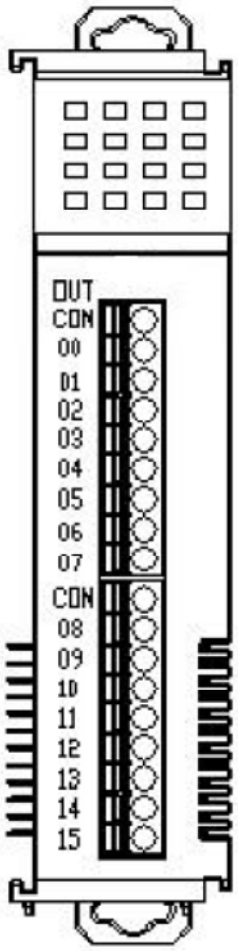
Расположение клемм дискретных модулей расширения

MX16DI



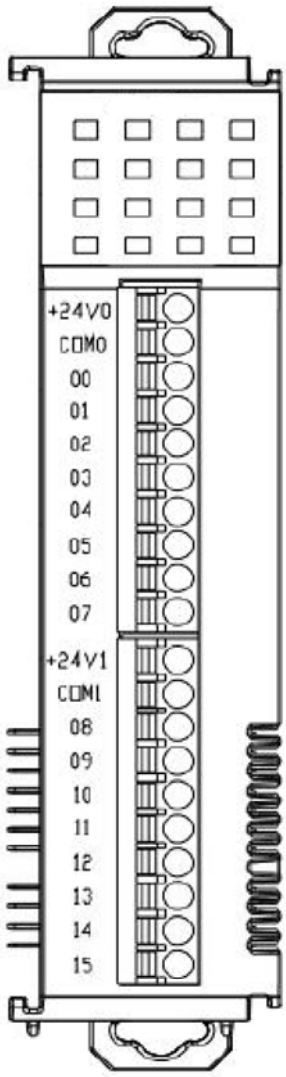
S/S 0
00
01
02
03
04
05
06
07
S/S 1
08
09
10
11
12
13
14
15

MX16DOT



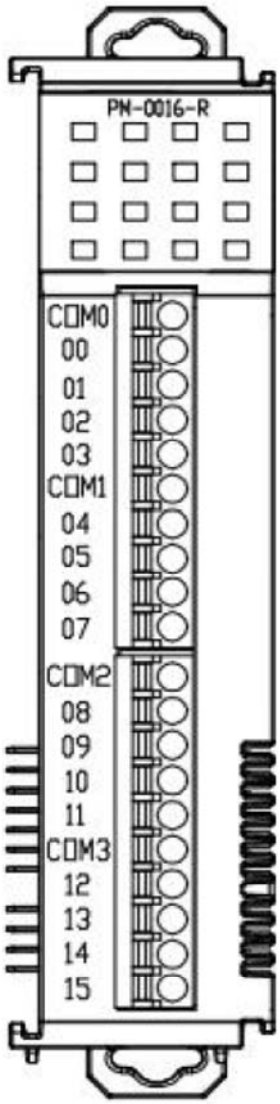
COM
00
01
02
03
04
05
06
07
COM
08
09
10
11
12
13
14
15

MX16DOP



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

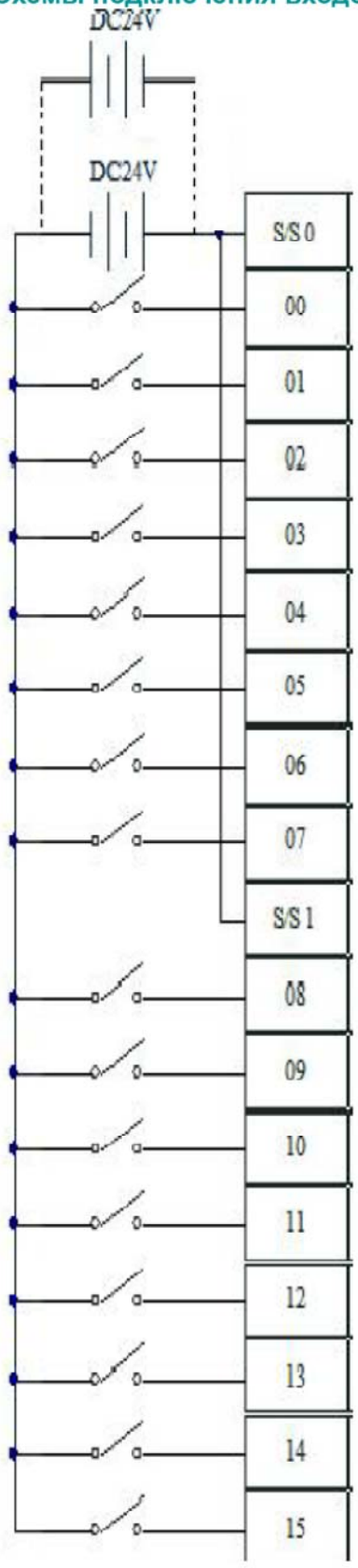
MX16DOR



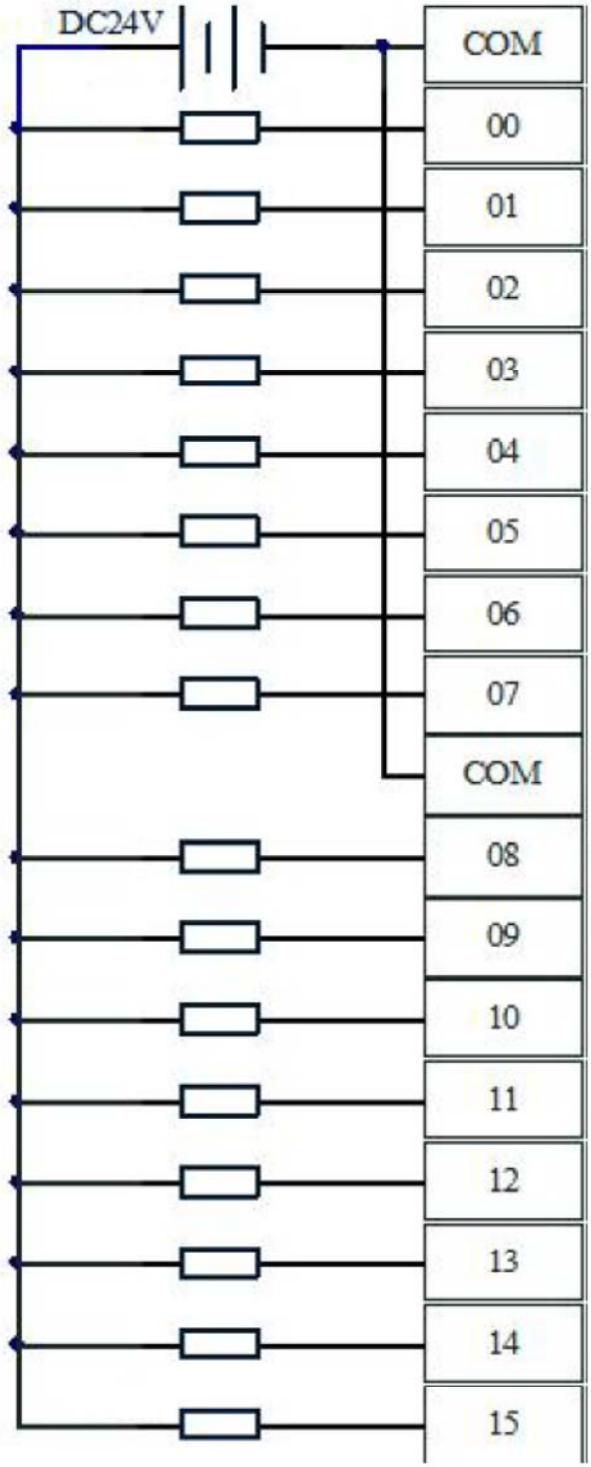
COM0
00
01
02
03
COM1
04
05
06
07
COM2
08
09
10
11
COM3
12
13
14
15

Схемы подключения дискретных входов-выходов модулей расширения

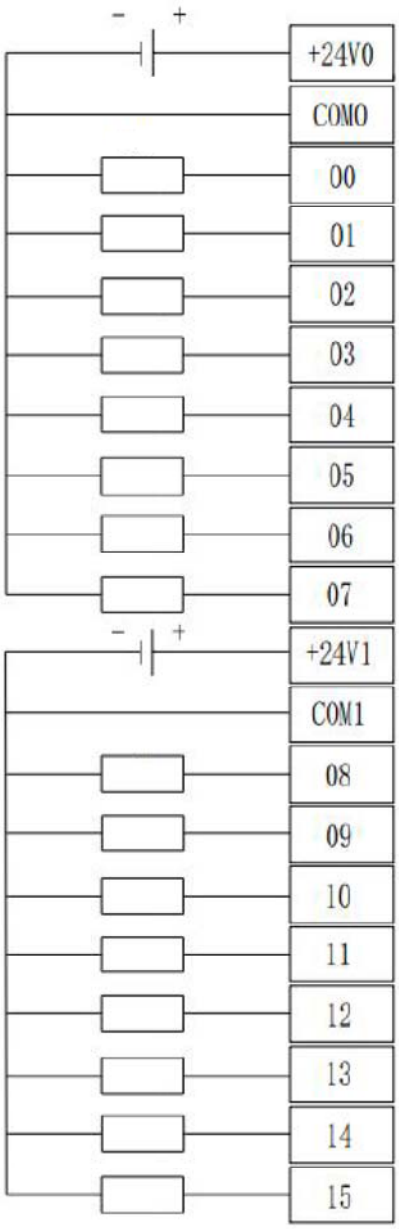
Схемы подключения входов на модуле расширения MX16DI



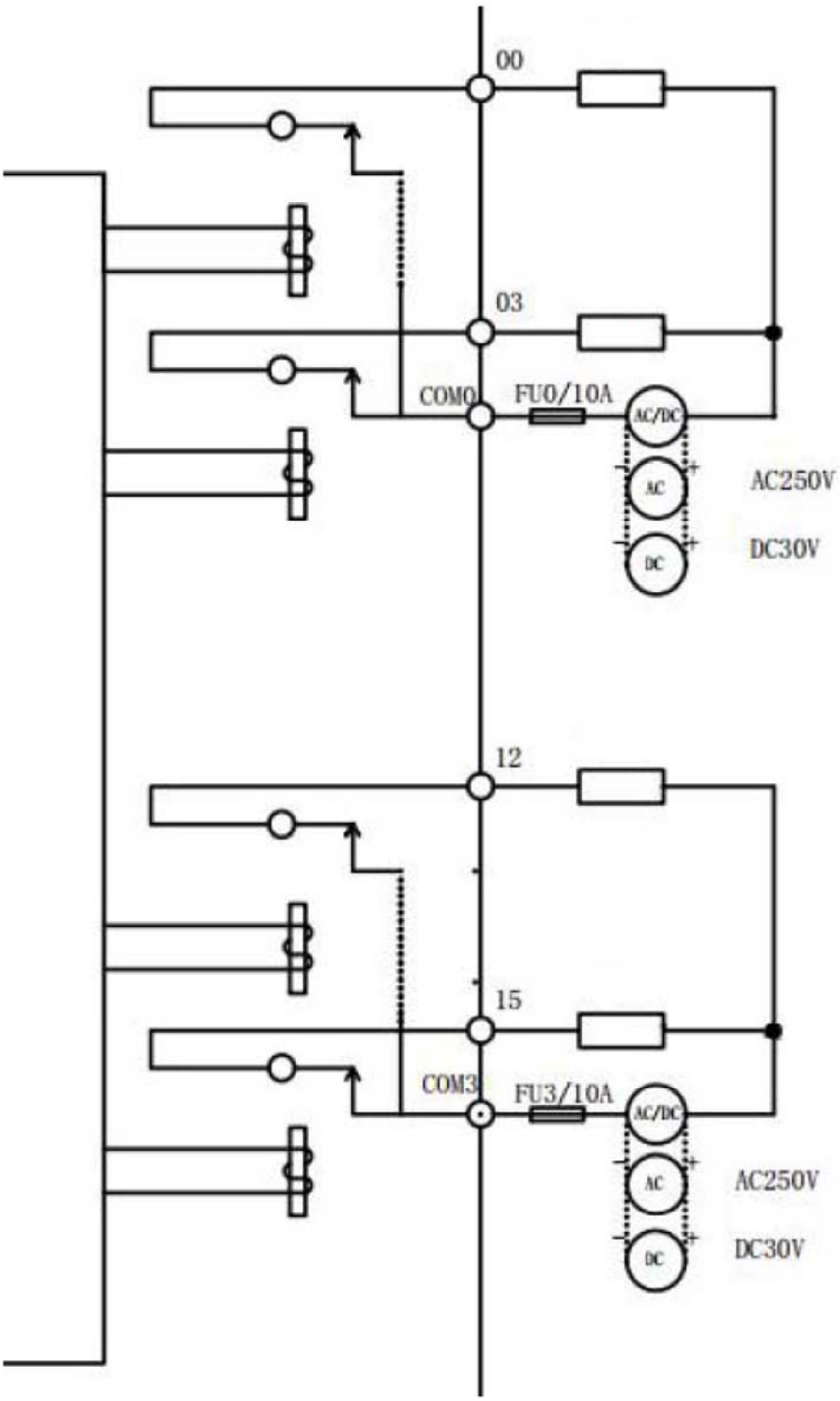
Схемы подключения выходов на модуле расширения MX16DOT



Схемы подключения выходов на модуле расширения MX16DOP



Схемы подключения выходов на модуле расширения MX16DOR



Объекты EtherCAT дискретных модулей

Объекты EtherCAT модуля MX16DI (TxPDO) (состояние входов)

Индекс	Субиндекс	Содержание	Тип данных	Доступ	Описание
6000-61F0H	00H	Дискретные входы			На шине может быть до 32-х модулей. Нумерация 0-31. Индекс объекта EtherCAT будет нумероваться с шагом 1: 0x6000 – 0x61F0
	01H	Input bit[1-16]	Unsigned 16	RO	
	02H	Input bit[17-32]	Unsigned 16	RO	

Объекты EtherCAT модулей MX16DOT, MX16DOR, MX16DOP (SDO) (конфигурация и статус)

Индекс	Субиндекс	Содержание	Тип данных	Доступ	Описание
8000-81F0H	00H	Конфигурация			На шине может быть до 32-х модулей. Нумерация 0-31. Индекс объекта EtherCAT будет нумероваться с шагом 1: 0x8000 – 0x81F0
	01H	Состояние выходов при потере связи bit[1-16]	Unsigned 16	RW	1: Выход сохраняет состояние 0: Выход сбрасывается
	02H	Состояние выходов при потере связи bit[17-32]	Unsigned 16	RW	1: Выход сохраняет состояние 0: Выход сбрасывается

Объекты EtherCAT модулей MX16DOT, MX16DOR, MX16DOP (TxPDO) (состояние выходов)

Индекс	Субиндекс	Содержание	Тип данных	Доступ	Описание
6000-61F0H	00H	Состояние дискретных выходов			На шине может быть до 32-х модулей. Нумерация 0-31. Индекс объекта EtherCAT будет нумероваться с шагом 1: 0x6000 – 0x61F0
	01H	Input bit[1-16]	Unsigned 16	RO	
	02H	Input bit[17-32]	Unsigned 16	RO	

Объекты EtherCAT модулей MX16DOT, MX16DOR, MX16DOP (RxPDO) (установка выходов)

Индекс	Субиндекс	Содержание	Тип данных	Доступ	Описание
7000-71F0H	00H	Установка дискретных выходов			На шине может быть до 32-х модулей. Нумерация 0-31. Индекс объекта EtherCAT будет нумероваться с шагом 1: 0x7000 – 0x71F0
	01H	Output bit[1-16]	Unsigned 16	RW	
	02H	Output bit[17-32]	Unsigned 16	RW	

RO – Read Only (только для чтения)
RW – Read – Write (чтение и запись)

Спецификация модулей аналоговых входов-выходов

MX04AD

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: 100.0 x 23.0 x 98.5 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	MX04DA
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: 100.0 x 23.0 x 98.5 мм
Потребляемая мощность	2.0 Вт
Рабочая температура	0-55℃

Характеристики преобразования входных сигналов

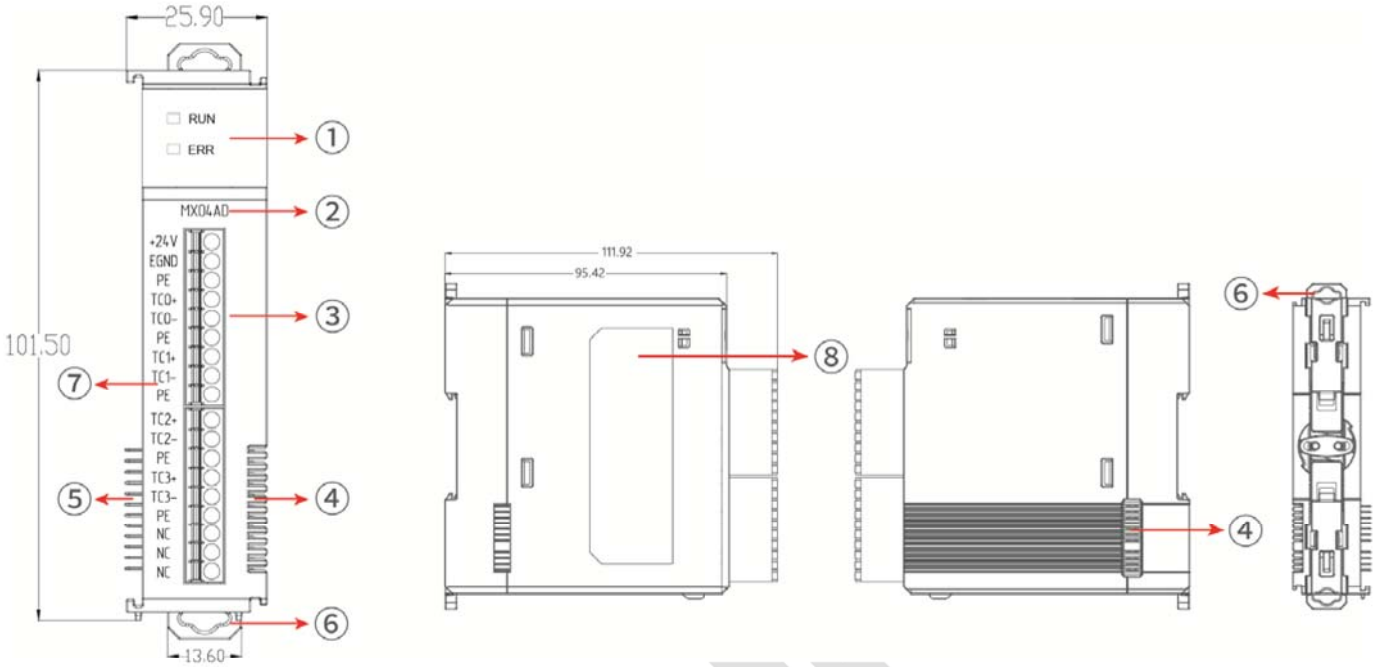
ЦАП	Выход по напряжению				
Рабочий диапазон	-10 В~10 В	0 В~10 В	±5 В	0 В~5 В	1 В~5 В
Аппаратный диапазон	-10.2 В ~10.2 В	-0.2 В ~10.2 В	-5.1 В ~5.1 В	-0.1 В ~5.1 В	0.2 В ~5.1 В
Диапазон цифровой шкалы	-32000 ~ 32000				
Погрешность осреднения при температуре 25℃	±0.2% FS				
Погрешность осреднения при 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

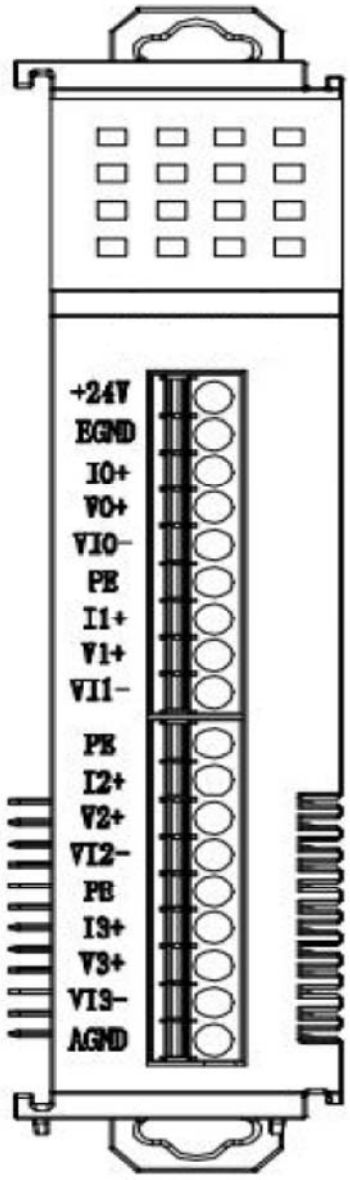
MX04AD/MX04DA



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

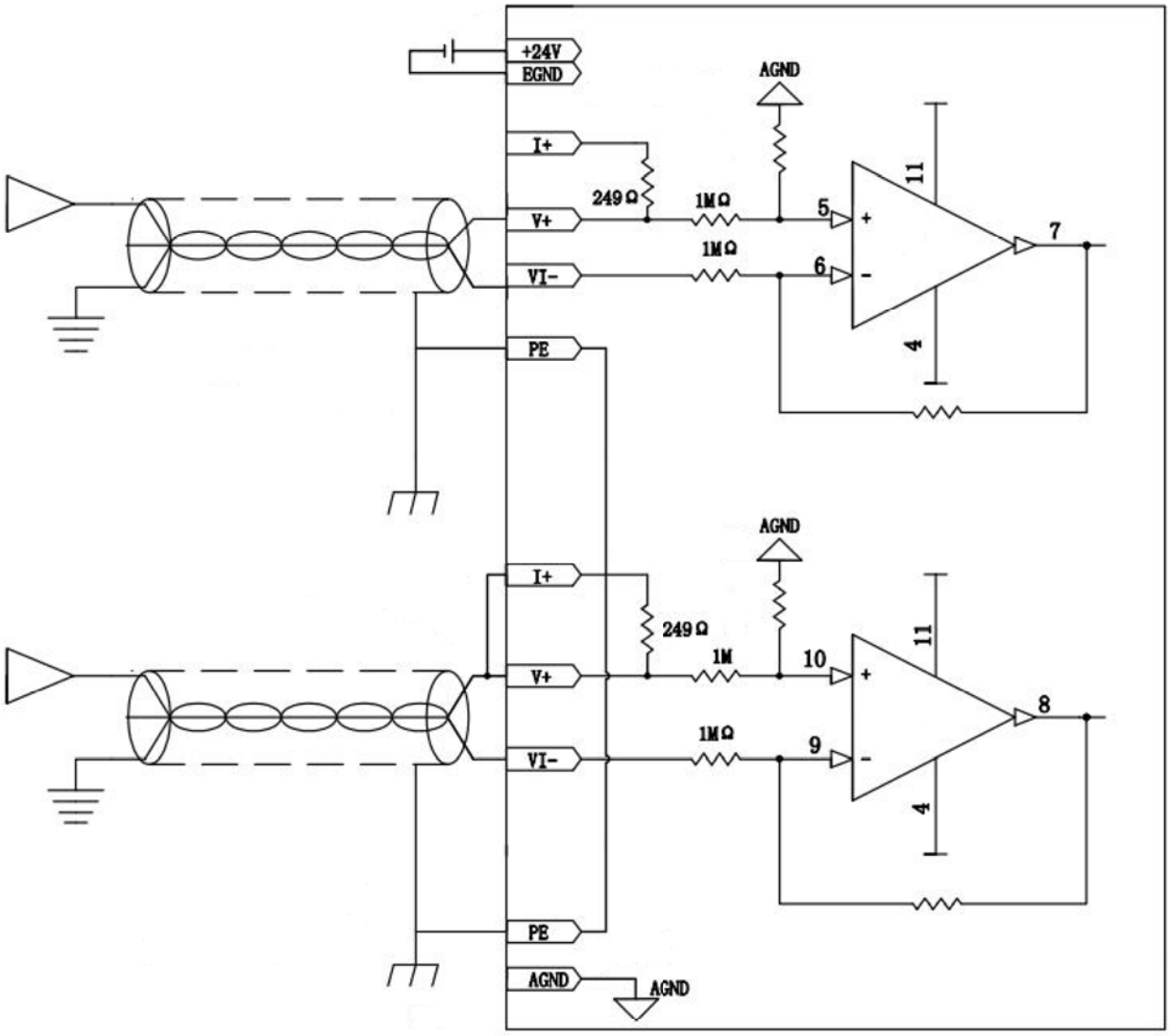
MX04AD/MX04DA



+24V
EGND
I0+
V0+
VI0-
PE
I1+
V1+
VI1-
PE
I2+
V2+
VI2-
PE
I3+
V3+
VI3-
AGND

Схемы подключения аналоговых входов-выходов

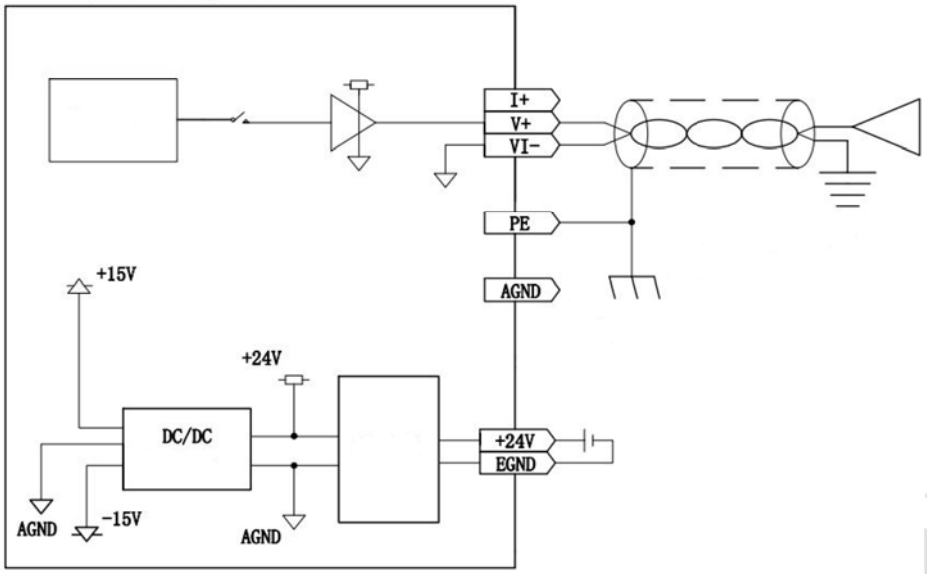
Схемы подключения входов на модуле расширения MX04AD



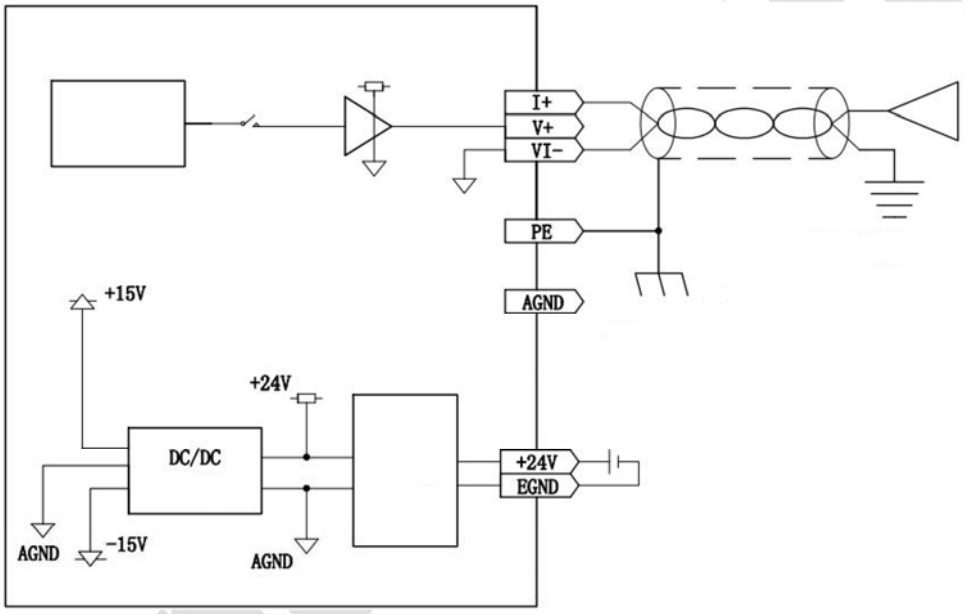
Модуль требует внешнего питания. Подключите 24 VDC к клеммам +24V и EGND.
Для подключения датчика используйте экранированную витую пару, экран которой нужно подключить к сигнальному заземлению.
Модуль необходимо установить на хорошо заземленную металлическую ДИН-рейку. В верхней части защёлки имеется 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.

Объекты EtherCAT аналоговых модулей

Объекты EtherCAT модулей MX04AD и MX04DA (SDO) (параметры модуля)

Индекс	Субиндекс	Содержание	Тип данных	Доступ	Описание
9000-91F0H	00H	Параметры модуля			На шине может быть до 32-х модулей. Нумерация 0-31. Индекс объекта EtherCAT будет нумероваться с шагом 1: 0x9000 – 0x91F0
	01H	Зарезервировано	Unsigned 32	RO	
	02H	Зарезервировано	Unsigned 32	RO	
	03H	Зарезервировано	Unsigned 32	RO	
	04H	Зарезервировано	Unsigned 32	RO	
	05H	Vendor ID	Unsigned 32	RO	Код производителя
	06H	Product Code	Unsigned 32	RO	Код продукта
	07H	Revision Number	Unsigned 32	RO	Номер версии встроенного ПО
	08H	Serial Number	Unsigned 32	RO	Серийный номер
	09H	Fpga Revison	Unsigned 32	RO	Номер версии ПЛИС

Объекты EtherCAT модуля MX04AD (SDO) (настройки модуля)

Индекс	Субиндекс	Содержание	Тип данных	Доступ	Описание
8000-81F0H	00H	Настройки модуля			На шине может быть до 32-х модулей. Нумерация 0-31. Индекс объекта EtherCAT будет нумероваться с шагом 1: 0x8000 – 0x81F0
	01H	Режим работы канала AD0	Unsigned 8	RW	Bit2-Bit0: 000: ±5V 001 : 1-5V 010 : ±10V 011 : 0-10V 100 : 0-20mA 101 : 4-20mA 110 : 0-5V 111 : ±20mA
	02H	Режим работы канала AD1	Unsigned 8	RW	см. канал AD0
	03H	Режим работы канала AD2	Unsigned 8	RW	см. канал AD0
	04H	Режим работы канала AD3	Unsigned 8	RW	см. канал AD0
8001-81F1H	00H	Цикл опроса (мс)			Нумерация модулей 0-31 0x8001-0x81F1H
	01H	Канал AD0	Unsigned 8	RW	1 – 255 мс
	02H	Канал AD1	Unsigned 8	RW	1 – 255 мс
	03H	Канал AD2	Unsigned 8	RW	1 – 255 мс
	04H	Канал AD3	Unsigned 8	RW	1 – 255 мс
8002-81F2H	00H	Сброс и сохранение параметров			Нумерация модулей 0-31 0x8002-0x81F2H
	01H	Сохранение параметров	Unsigned 8	RW	Запишите число 1
	02H	Сброс параметров на заводс.	Unsigned 8	RW	Запишите число 1
8003-81F3H	00H	Разрешение работы каналов			Нумерация модулей 0-31 0x8003-0x81F3H
	01H	AD0 EN	Unsigned 8	RW	1: Вкл. 0: Выкл.
	02H	AD1 EN	Unsigned 8	RW	1: Вкл. 0: Выкл.
	03H	AD2 EN	Unsigned 8	RW	1: Вкл. 0: Выкл.
	04H	AD3 EN	Unsigned 8	RW	1: Вкл. 0: Выкл.

Объекты EtherCAT модуля MX04AD (TxPDO) (данные процесса)

Индекс	Субиндекс	Содержание	Тип данных	Доступ	Описание
TxPDO0	1A00H				
6000~61F0H	00H	Измеренные значения на каналах			На шине может быть до 32-х модулей. Нумерация 0-31. Индекс объекта EtherCAT будет нумероваться с шагом 1: 0x6000 – 0x61F0
	01H	Канал AD0	Unsigned 16	RO	
	02H	Канал AD1	Unsigned 16	RO	
	03H	Канал AD2	Unsigned 16	RO	
	04H	Канал AD3	Unsigned 16	RO	
TxPDO1	1A01H				
A000~A1F0H	00H	Статус каналов			Нумерация модулей 0-31 0xA000-0xA1F0H
	01H	Канал AD0	Unsigned 8	RO	Bit: 4 1: Значение вне диапазона 0: Значение нормальное
	02H	Канал AD1	Unsigned 8	RO	см. канал AD0
	03H	Канал AD2	Unsigned 8	RO	см. канал AD0
	04H	Канал AD3	Unsigned 8	RO	см. канал AD0

Объекты EtherCAT модуля MX04DA (SDO) (настройки модуля)

Индекс	Субиндекс	Содержание	Тип данных	Доступ	Описание
8000-81F0H	00H	Настройки модуля			На шине может быть до 32-х модулей. Нумерация 0-31. Индекс объекта EtherCAT будет нумероваться с шагом 1: 0x8000 – 0x81F0
	01H	Режим работы канала DA0	Unsigned 8	RW	Bit2-Bit0: 000 : 0-5V 001 : 1-5V 010 : ±5V 011 : 0-10V 100 : ±10V 101 : 0-20mA 110 : 4-20mA
	02H	Режим работы канала DA1	Unsigned 8	RW	см. канал DA0
	03H	Режим работы канала DA2	Unsigned 8	RW	см. канал DA0
	04H	Режим работы канала DA3	Unsigned 8	RW	см. канал DA0
8001-81F1H	00H	Разрешение работы каналов			Нумерация модулей 0-31 0x8001-0x81F1H
	01H	DA0 EN	Unsigned 8	RW	1: Вкл. 0: Выкл.
	02H	DA1 EN	Unsigned 8	RW	1: Вкл. 0: Выкл.
	03H	DA2 EN	Unsigned 8	RW	1: Вкл. 0: Выкл.
	04H	DA3 EN	Unsigned 8	RW	1: Вкл. 0: Выкл.
8002-81F2H	00H	Состояние при потере связи			Нумерация модулей 0-31 0x8002-0x81F2H
	01H	DA0	Unsigned 8	RW	0 : сохранение текущего значения; 1 : сброс выхода на ноль; 2 : предустановленное выходное значение в 8003-81F3H
	02H	DA1	Unsigned 8	RW	см. канал DA0

	03H	DA2	Unsigned 8	RW	см. канал DA0
	04H	DA3	Unsigned 8	RW	см. канал DA0
8003-81F3H	00H	Предустановленное значение каналов при потере связи			Нумерация модулей 0-31 0x8003-0x81F3H
	01H	DA0	Unsigned 16	RW	-32000 ~ + 32000
	02H	DA1	Unsigned 16	RW	-32000 ~ + 32000
	03H	DA2	Unsigned 16	RW	-32000 ~ + 32000
	04H	DA3	Unsigned 16	RW	-32000 ~ + 32000
8004-81F4H	00H	Сброс и сохранение параметров			Нумерация модулей 0-31 0x8004-0x81F4H
	01H	Сохранение параметров	Unsigned 8	RW	Запишите число 1 (кроме параметров калибровки)
	02H	Сброс параметров на заводские настройки	Unsigned 8	RW	Запишите число 1 (кроме параметров калибровки)

Объекты EtherCAT модуля MX04DA (TxPDO) (данные процесса)

Индекс	Субиндекс	Содержание	Тип данных	Доступ	Описание
A000~A1F0H	00H	Статус каналов			На шине может быть до 32-х модулей. Нумерация 0-31. Индекс объекта EtherCAT будет нумероваться с шагом 1: 0xA000-0xA1F0H
	01H	DA0	Unsigned 8	RW	Bit: 4 1: Значение вне диапазона 0: Значение нормальное
	02H	DA1	Unsigned 8	RW	см. канал DA0
	03H	DA2	Unsigned 8	RW	см. канал DA0
	04H	DA3	Unsigned 8	RW	см. канал DA0

Объекты EtherCAT модуля MX04DA (RxPDO) (данные процесса)

Индекс	Субиндекс	Содержание	Тип данных	Доступ	Описание
7000~71F0H	00H	Установка значений на каналах			На шине может быть до 32-х модулей. Нумерация 0-31. Индекс объекта EtherCAT будет нумероваться с шагом 1: 0x7000 – 0x71F0
	01H	DA0	Unsigned 16	RW	-32000 ~ + 32000
	02H	DA1	Unsigned 16	RW	-32000 ~ + 32000
	03H	DA2	Unsigned 16	RW	-32000 ~ + 32000
	04H	DA3	Unsigned 16	RW	-32000 ~ + 32000

Спецификация температурных модулей

MX04TC

Общие характеристики

Модуль	MX04TC
Число входов	4 входа под термопары
Разрешение	0.1°C / 0.1°F
Напряжение питания	24 VDC
Разъем	Разъемный клеммный блок
Время преобразования	250 мс, 500 мс или 1000 мс /4 канала
Габаритные размеры	ВхШхГ: 100.0 x 23.0 x 98.5 мм
Потребляемая мощность	2.0 Вт
Рабочая температура	0-55℃

Характеристики преобразования входных сигналов

Параметр	MX04TC
Типы и рабочие диапазоны датчиков температуры (термопар)	J : -80℃~1200℃/-112°F~2192°F K : -1000℃~1350℃/-148°F~2462°F R : 0℃~1750℃/32°F~3182°F S : 0℃~1750℃/32°F~3182°F N : -150℃~1300℃/-238°F~2372°F B : 200℃~1800℃/392°F~3272°F T : -150℃~400℃/-238°F~752°F E : -150℃~980℃/-238°F~1796°F -100mv~100mv (точность ±0.5%)
Точность компенсации холодного спая	±1℃
Погрешность измерения при температуре 25℃	±0.1% FS+ 1℃ (компенсация холодного спая)
Погрешность измерения при полном температурном диапазоне 0-55℃	±0.3% FS + 1℃ (компенсация холодного спая)
Аппаратное разрешение	16 бит -32000 ~ +32000

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

MX04RC

Общие характеристики

Модуль	MX04RC
Число входов	4 входа под термосопротивления
Разрешение	0.1°C / 0.1°F
Напряжение питания	24 VDC
Разъем	Разъемный клеммный блок
Время преобразования	250 мс, 500 мс или 1000 мс /4 канала
Габаритные размеры	ВхШхГ: 100.0 x 23.0 x 98.5 мм
Потребляемая мощность	2.0 Вт
Рабочая температура	0-55℃

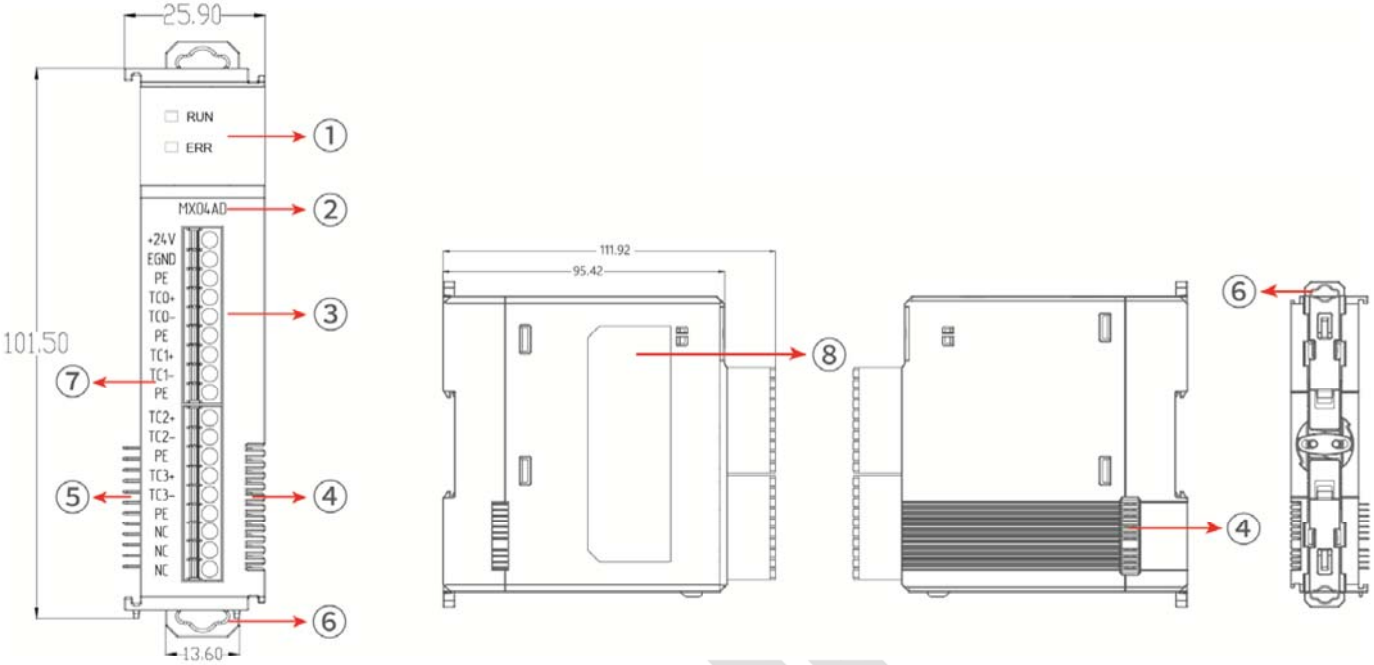
Характеристики преобразования входных сигналов

Параметр	MX04RC
Типы и рабочие диапазоны датчиков температуры (термопар)	Pt100 : -180℃~800℃/-292°F~1472°F Pt1000 : -180℃~800℃/-292°F~1472°F JPt100 : -180℃~500℃/-292°F~932°F Ni100 : -80℃~170℃/-112°F~332°F Ni1000 : -80℃~170℃/-112°F~332°F LG-Ni1000 : -50℃~180℃/-58°F~356°F Cu50 : -50℃~150℃/-58°F~302°F Cu100 : -50℃~150℃/-58°F~302°F
Погрешность измерения при температуре 25℃	Pt100, Pt1000, JPt100, Ni100, Ni1000, 0~300Ω, 0~3000Ω : ±0.1% FS
Погрешность измерения при температуре 25℃	LG-Ni1000 : ±0.2% FS
Погрешность измерения при температуре 25℃	Cu50 : ±4℃
Погрешность измерения при температуре 25℃	Cu100 : ±2℃
Аппаратное разрешение	16 бит -32000 ~ +32000

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

Внешний вид и размеры температурных модулей

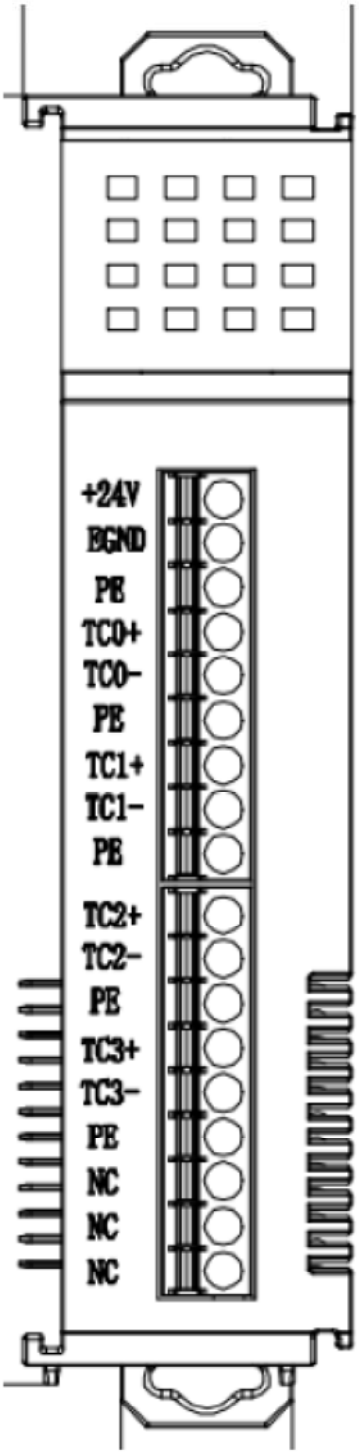
MX04TC/MX04RC



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

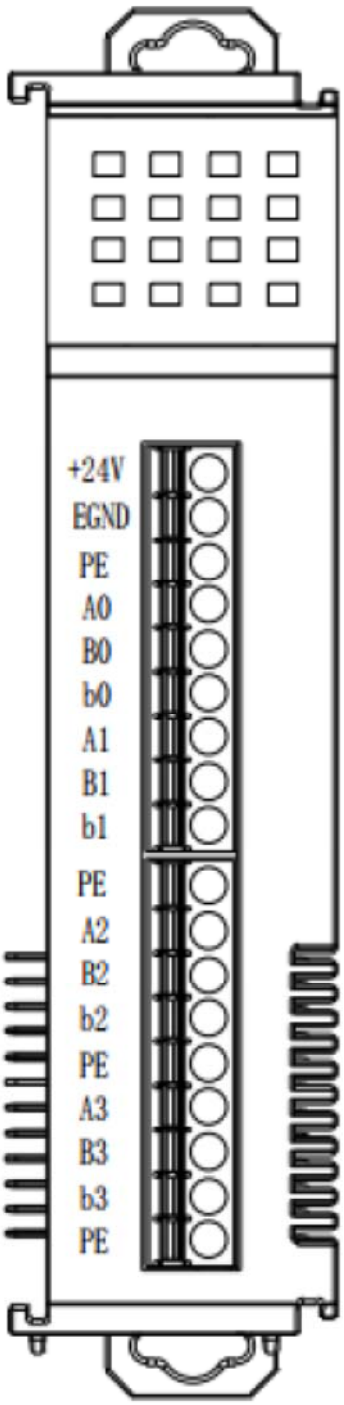
Расположение клемм температурных модулей

MX04TC



+24V
EGND
PE
TC0+
TC0-
PE
TC1+
TC1-
PE
TC2+
TC2-
PE
TC3+
TC3-
PE
NC
NC
NC

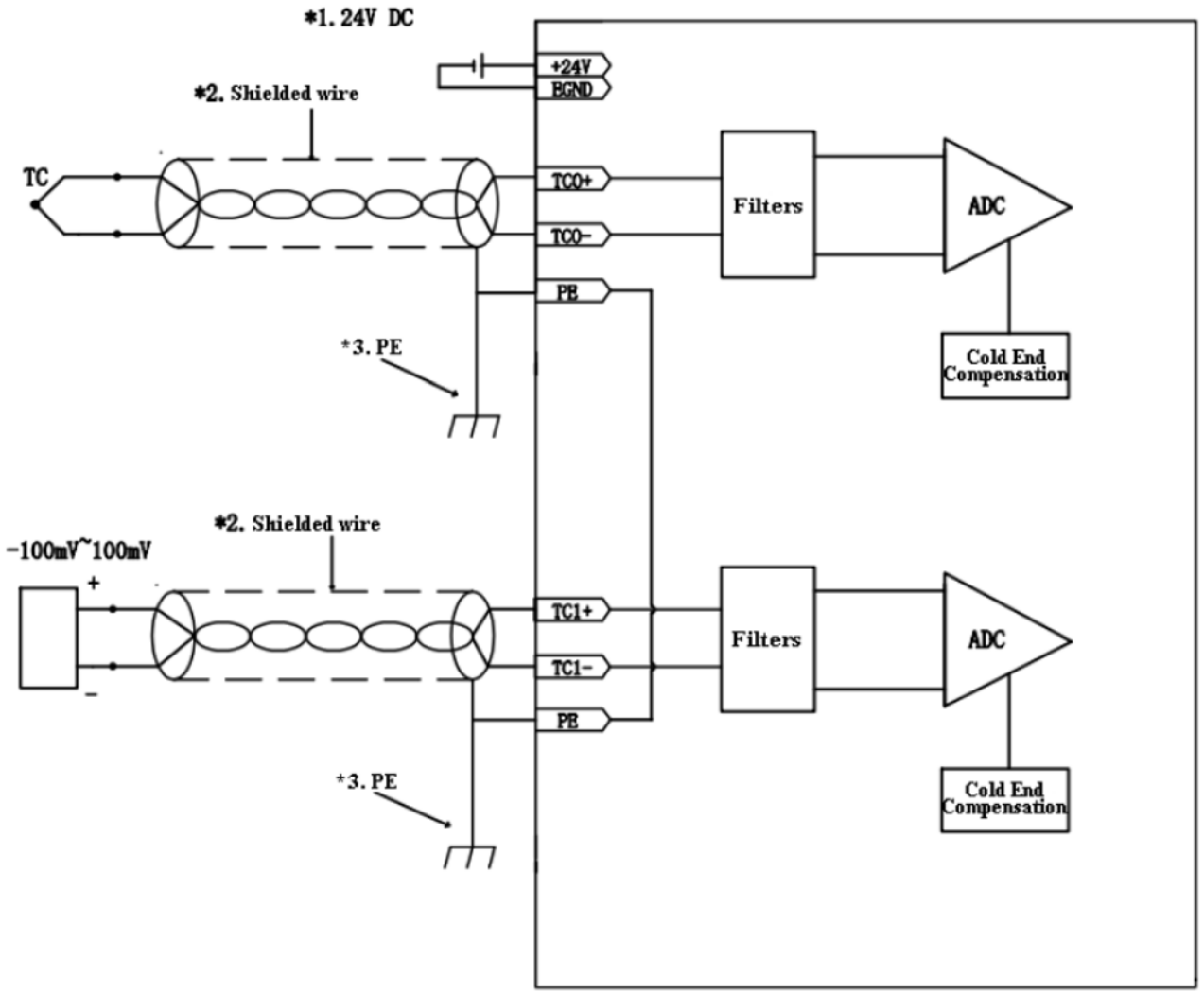
MX04RC



+24V
EGND
PE
A0
B0
b0
A1
B1
b1
PE
A2
B2
b2
PE
A3
B3
b3
PE

Схемы подключения температурных модулей

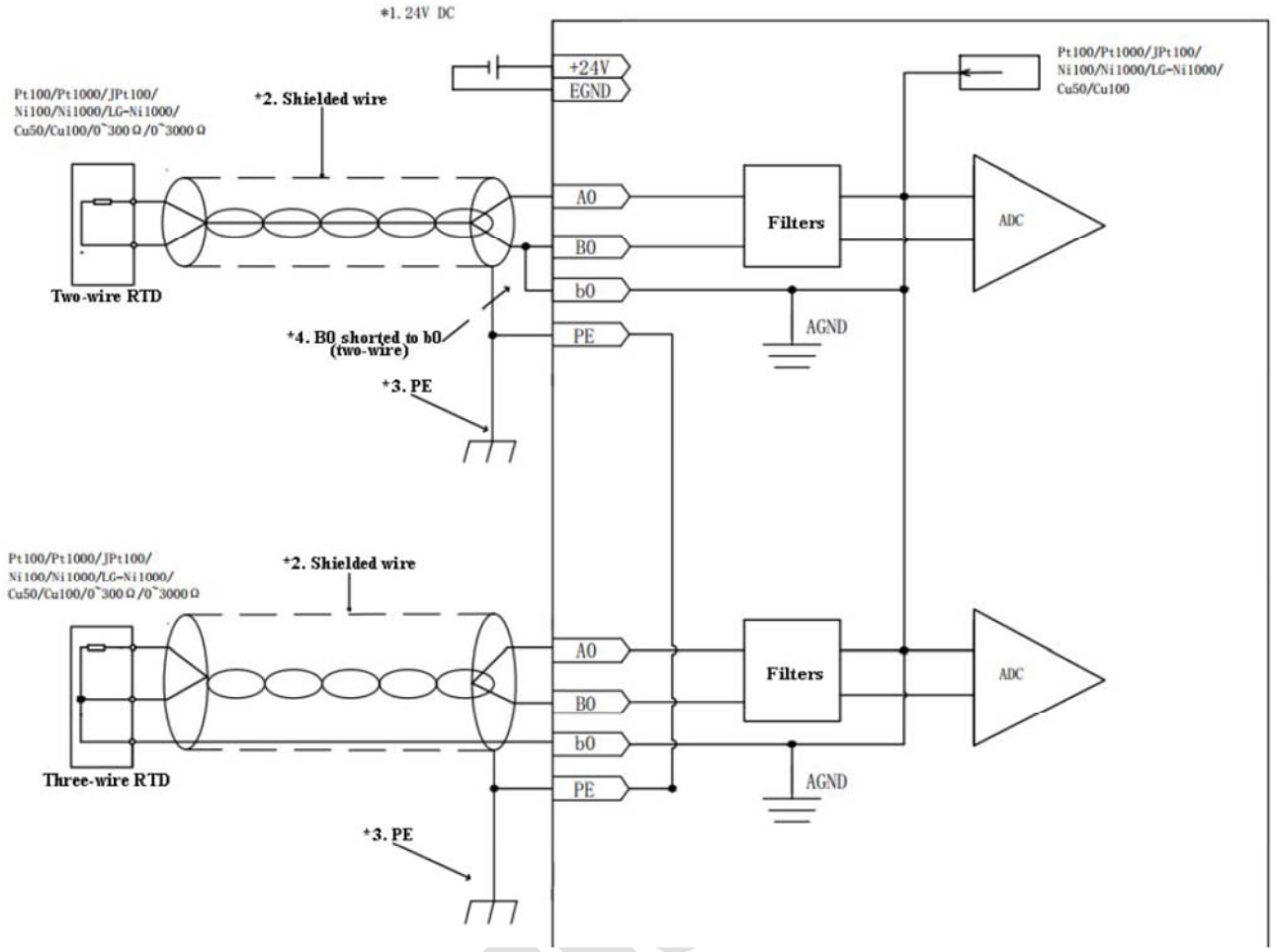
Схемы подключения входов на модуле расширения MX04TC



1. Модуль требует внешнего питания. Подключите 24 VDC к клеммам +24V и EGND
2. Для подключения датчика используйте экранированную витую пару
3. Экран витой пары нужно подключить к клемме PE

Модуль необходимо установить на хорошо заземленную металлическую ДИН-рейку. В верхней части защёлки имеется металлический вывод, который при защёлкивании модуля на ДИН-рейке входит в контакт с ней. Все 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 сигнального заземления можно подсоединить на одну из клемм PE (вместо соединения через ДИН-рейку).

Схемы подключения выходов на модуле расширения MX04RC



1. Модуль требует внешнего питания. Подключите 24 VDC к клеммам +24V и EGND
2. Для подключения датчика используйте экранированную витую пару
3. Экран витой пары нужно подключить к клемме PE

Модуль необходимо установить на хорошо заземленную металлическую ДИН-рейку. В верхней части защёлки имеется металлический вывод, который при защёлкивании модуля на ДИН-рейке входит в контакт с ней. Все 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 сигнального заземления можно подсоединить на одну из клемм PE (вместо соединения через ДИН-рейку).

Объекты EtherCAT температурных модулей

Объекты EtherCAT модулей MX04TC и MX04RC (SDO) (параметры модуля)

Индекс	Субиндекс	Содержание	Тип данных	Доступ	Описание
9000-91F0H	00H	Параметры модуля			На шине может быть до 32-х модулей. Нумерация 0-31. Индекс объекта EtherCAT будет нумероваться с шагом 1: 0x9000 – 0x91F0
	01H	Зарезервировано	Unsigned 32	RO	
	02H	Зарезервировано	Unsigned 32	RO	
	03H	Зарезервировано	Unsigned 32	RO	
	04H	Зарезервировано	Unsigned 32	RO	
	05H	Vendor ID	Unsigned 32	RO	Код производителя
	06H	Product Code	Unsigned 32	RO	Код продукта
	07H	Revision Number	Unsigned 32	RO	Номер версии встроенного ПО
	08H	Serial Number	Unsigned 32	RO	Серийный номер

Объекты EtherCAT модуля MX04TC (SDO) (настройки модуля)

Индекс	Субиндекс	Содержание	Тип данных	Доступ	Описание
8000-81F0H	00H	Выбор единиц измерения			На шине может быть до 32-х модулей. Нумерация 0-31. Индекс объекта EtherCAT будет нумероваться с шагом 1: 0x8000 – 0x81F0
	01H	Единицы измерения TC0	Unsigned 8	RW	Канал TC0: °C/°F 0:°C 1:°F ; Заводское значение: 0
	02H	Единицы измерения TC1	Unsigned 8	RW	см. канал TC0
	03H	Единицы измерения TC2	Unsigned 8	RW	см. канал TC0
	04H	Единицы измерения TC3	Unsigned 8	RW	см. канал TC0
8001-81F1H	00H	Разрешение работы каналов			Нумерация модулей 0-31 0x8001-0x81F1H
	01H	Канал TC0	Unsigned 8	RW	Канал 0: 0:OFF 1:ON ; Заводское значение: 1
	02H	Канал TC1	Unsigned 8	RW	см. канал TC0
	03H	Канал TC2	Unsigned 8	RW	см. канал TC0
	04H	Канал TC3	Unsigned 8	RW	см. канал TC0
8002-81F2H	00H	Выбор типа датчика			Нумерация модулей 0-31 0x8002-0x81F2H
	01H	Канал TC0	Unsigned 8	RW	Канал 0: 0:J; 1:K; 2:E; 3:N; 4:T; 5:R; 6:S; 7:B; 8:±100mV Заводское значение: 1
	02H	Канал TC1	Unsigned 8	RW	см. канал TC0
	03H	Канал TC2	Unsigned 8	RW	см. канал TC0
	04H	Канал TC3	Unsigned 8	RW	см. канал TC0
8003-81F3H	00H	Количество замеров выборки осреднения			Нумерация модулей 0-31 0x8003-0x81F3H
	01H	Канал TC0	Unsigned 8	RW	Канал 0: 1~100 Заводское значение: 5
	02H	Канал TC1	Unsigned 8	RW	см. канал TC0
	03H	Канал TC2	Unsigned 8	RW	см. канал TC0
	04H	Канал TC3	Unsigned 8	RW	см. канал TC0
8004-81F4H	00H	Определение превышения порога температуры			Нумерация модулей 0-31 0x8004-0x81F4H
	01H	Канал TC0	Unsigned 8	RW	Канал 0: 0: Порог по умолчанию

					1: Порог установленный
	02H	Канал TC1	Unsigned 8	RW	см. канал TC0
	03H	Канал TC2	Unsigned 8	RW	см. канал TC0
	04H	Канал TC3	Unsigned 8	RW	см. канал TC0
8005-81F5H	00H	Установка порогов температур для аварии по выходу за диапазон			Нумерация модулей 0-31 0x8005-0x81F5H
	01H	Канал TC0 нижний предел	Signed 16	RW	Канал 0: Порог по умолчанию 1500 (-150°C)
	02H	Канал TC0 верхний предел	Signed 16	RW	Канал 0: Порог по умолчанию 18000 (1800°C)
	03H	Канал TC1 нижний предел	Signed 16	RW	Канал 0: Порог по умолчанию 1500 (-150°C)
	04H	Канал TC1 верхний предел	Signed 16	RW	Канал 0: Порог по умолчанию 18000 (1800°C)
	05H	Канал TC2 нижний предел	Signed 16	RW	Канал 0: Порог по умолчанию 1500 (-150°C)
	06H	Канал TC2 верхний предел	Signed 16	RW	Канал 0: Порог по умолчанию 18000 (1800°C)
	07H	Канал TC3 нижний предел	Signed 16	RW	Канал 0: Порог по умолчанию 1500 (-150°C)
	08H	Канал TC3 верхний предел	Signed 16	RW	Канал 0: Порог по умолчанию 18000 (1800°C)
8006-81F6H	00H	Проверка обрыва датчика			Нумерация модулей 0-31 0x8006-0x81F6H
	01H	Канал TC0	Signed 8	RW	Канал 0: 0: Выключена 1: Включена Заводское значение: 1
	02H	Канал TC1	Signed 8	RW	см. канал TC0
	03H	Канал TC2	Signed 8	RW	см. канал TC0
	04H	Канал TC3	Signed 8	RW	см. канал TC0
	05H	Отображение значения при обрыве датчика	Signed 8	RW	0: 32767 1: –32767 Заводское значение: 0
8007-81F7H	00H	Сохранение, сброс, цикл			Нумерация модулей 0-31 0x8007-0x81F7H
	01H	Цикл опроса входов	Signed 8	RW	0:250ms 1:500ms 2:1000ms Заводское значение: 1
	02H	Сохранение параметров модуля	Signed 8	RW	Запишите 1
	03H	Сброс на заводские установки	Signed 8	RW	Запишите 1
8008-81F8H	00H	Установка смещения нуля			Нумерация модулей 0-31 0x8008-0x81F8H
	01H	Канал TC0	Unsigned 8	RW	Заводское значение: 0
	02H	Канал TC1	Unsigned 8	RW	Заводское значение: 0
	03H	Канал TC2	Unsigned 8	RW	Заводское значение: 0
	04H	Канал TC3	Unsigned 8	RW	Заводское значение: 0
8009-81F9H	00H	Включение компенсации холодного спая			Нумерация модулей 0-31 0x8008-0x81F8H
	01H	Канал TC0	Unsigned 8	RW	Канал 0: 0: Выключена 1: Включена Заводское значение: 1
	02H	Канал TC1	Unsigned 8	RW	см. канал TC0
	03H	Канал TC2	Unsigned 8	RW	см. канал TC0
	04H	Канал TC3	Unsigned 8	RW	см. канал TC0

800A-81FAH	00H	Тип компенсации холодного спая			Нумерация модулей 0-31 0x800A-0x81FAH
	01H	Канал TC0	Unsigned 8	RW	Канал 0: 0: Компенсация на чипе 1: Значение в регистре Заводское значение: 0
	02H	Канал TC1	Unsigned 8	RW	см. канал TC0
	03H	Канал TC2	Unsigned 8	RW	см. канал TC0
	04H	Канал TC3	Unsigned 8	RW	см. канал TC0
800B-81FBH	00H	Значение в регистре для компенсации холодного спая			Нумерация модулей 0-31 0x800B-0x81FBH
	01H	Канал TC0	Signed 32	RW	Канал 0: Значение для компенсации температуры холодного спая в текущих единицах
	02H	Канал TC1	Signed 32	RW	см. канал TC0
	03H	Канал TC2	Signed 32	RW	см. канал TC0
	04H	Канал TC3	Signed 32	RW	см. канал TC0
800E-81FEH	00H	Коэффициенты ПИД регулятора			Нумерация модулей 0-31 0x800E-0x81FEH
	01H	Канал TC0 Kp	Signed 32	RW	Пропорциональный коэф.
	02H	Канал TC0 Ti	Signed 32	RW	Интегральный коэф.
	03H	Канал TC0 Td	Signed 32	RW	Дифференциальный коэф.
	04H	Канал TC0 Cycle	Signed 32	RW	Цикл ПИД регулятора (сек)
	05H	Канал TC1 Kp	Signed 32	RW	Пропорциональный коэф.
	06H	Канал TC1 Ti	Signed 32	RW	Интегральный коэф.
	07H	Канал TC1 Td	Signed 32	RW	Дифференциальный коэф.
	08H	Канал TC1 Cycle	Signed 32	RW	Цикл ПИД регулятора (сек)
	09H	Канал TC2 Kp	Signed 32	RW	Пропорциональный коэф.
	0AH	Канал TC2 Ti	Signed 32	RW	Интегральный коэф.
	0BH	Канал TC2 Td	Signed 32	RW	Дифференциальный коэф.
	0CH	Канал TC2 Cycle	Signed 32	RW	Цикл ПИД регулятора (сек)
	0DH	Канал TC3 Kp	Signed 32	RW	Пропорциональный коэф.
	0EH	Канал TC3 Ti	Signed 32	RW	Интегральный коэф.
	0FH	Канал TC3 Td	Signed 32	RW	Дифференциальный коэф.
	10H	Канал TC3 Cycle	Signed 32	RW	Цикл ПИД регулятора (сек)
800F-81FFH	00H	Уставка и управление ПИД регулятором			Нумерация модулей 0-31 0x800F-0x81FFH
	01H	Канал TC0 Уставка	Signed 16	RW	Задание температуры
	02H	Канал TC1 Уставка	Signed 16	RW	Задание температуры
	03H	Канал TC2 Уставка	Signed 16	RW	Задание температуры
	04H	Канал TC3 Уставка	Signed 16	RW	Задание температуры
	05H	Канал TC0 Пуск/Стоп ПИД	Unsigned 8	RW	0: Стоп 1:Работа Заводское значение: 0
	06H	Канал TC0 Автонастройка	Unsigned 8	RW	0: Стоп 1:Начать автонастройку Заводское значение: 0
	07H	Канал TC0 Полярность ПИД	Unsigned 8	RW	0: Униполярный 1: Биполярный Заводское значение: 0
	08H	Канал TC1 Пуск/Стоп ПИД	Unsigned 8	RW	0: Стоп 1:Работа Заводское значение: 0
	09H	Канал TC1 Автонастройка	Unsigned 8	RW	0: Стоп 1:Начать автонастройку Заводское значение: 0
	0AH	Канал TC1 Полярность ПИД	Unsigned 8	RW	0: Униполярный 1: Биполярный Заводское значение: 0

	0BH	Канал TC2 Пуск/Стоп ПИД	Unsigned 8	RW	0: Стоп 1:Работа Заводское значение: 0
	0CH	Канал TC2 Автонастройка	Unsigned 8	RW	0: Стоп 1:Начать автонастройку Заводское значение: 0
	0DH	Канал TC2 Полярность ПИД	Unsigned 8	RW	0: Униполярный 1: Биполярный Заводское значение: 0
	0EH	Канал TC3 Пуск/Стоп ПИД	Unsigned 8	RW	0: Стоп 1:Работа Заводское значение: 0
	0FH	Канал TC3 Автонастройка	Unsigned 8	RW	0: Стоп 1:Начать автонастройку Заводское значение: 0
	10H	Канал TC3 Полярность ПИД	Unsigned 8	RW	0: Униполярный 1: Биполярный Заводское значение: 0

Объекты EtherCAT модуля MX04TC (TxPDO) (данные процесса)

Индекс	Субиндекс	Содержание	Тип данных	Доступ	Описание
6000~61F0H	00H	Измеренные значения на каналах			На шине может быть до 32-х модулей. Нумерация 0-31. Индекс объекта EtherCAT будет нумероваться с шагом 1: 0x6000 – 0x61F0
	01H	Канал TC0	Signed 16	RO	Текущая температура (/10)
	02H	Канал TC1	Signed 16	RO	Текущая температура (/10)
	03H	Канал TC2	Signed 16	RO	Текущая температура (/10)
	04H	Канал TC3	Signed 16	RO	Текущая температура (/10)
A000~A1F0H	00H	Статус каналов			Нумерация модулей 0-31 0xA000-0xA1F0H
	01H	Канал TC0	Unsigned 8	RO	Канал 0 Состояние 0x00: Нормальное 0x01: Режим автонастройки, ожидание первого стандартного значения 0x02: Ожидание второго стандартного значения 0x03: Вычисление коэффициентов 0x04: Вычисление коэффициентов закончено 0x10:Превышение температуры 0x20: Обрыв датчика 0x40: Ошибка записи предела 0x80: Ошибка компенсации холодного спая или смещения температуры 0xFF: Канал отключен Если возникает одновременно превышение температуры и обрыв датчика, то код ошибки будет 0x10+0x20=0x30
	02H	Канал TC1	Unsigned 8	RO	см. канал TC0
	03H	Канал TC2	Unsigned 8	RO	см. канал TC0
	04H	Канал TC3	Unsigned 8	RO	см. канал TC0

A001~A1F1H	00H	Состояние ПИД регулятора			Нумерация модулей 0-31 0xA001-0xA1F1H
	01H	Канал TC0	Unsigned 8	RO	Канал 0 Состояние ПИД рег. Bit0: Индикация работы 0: Стоп 1: Работа (1 когда ПИД в работе, 0 в режиме автонастройки, 1 когда автонастройка окончена) Bit1:Режим выхода 0: Нагрев 1: Охлаждение Bit2: Автонастройка 0: Выключена 1: Процесс автонастройки (1 когда запущен процесс автонастройки, 0 когда закончена автонастройка) Bit3: Ошибка автонастройки 0: Нет ошибки 1: Ошибка автонастройки Bit4: heat_out 0:NO_Out 1:Tune_Out Bit5:cool_out 0:NO_Out 1:Tune_Out
	02H	Канал TC1	Unsigned 8	RO	см. канал TC0
	03H	Канал TC2	Unsigned 8	RO	см. канал TC0
	04H	Канал TC3	Unsigned 8	RO	см. канал TC0

Объекты EtherCAT модуля MX04RC (SDO) (настройки модуля)

Индекс	Субиндекс	Содержание	Тип данных	Доступ	Описание
8000-81F0H	00H	Выбор единиц измерения			На шине может быть до 32-х модулей. Нумерация 0-31. Индекс объекта EtherCAT будет нумероваться с шагом 1: 0x8000 – 0x81F0
	01H	Единицы измерения CH0	Unsigned 8	RW	Канал CH0: °C/°F 0:°C 1:°F ; Заводское значение: 0
	02H	Единицы измерения CH1	Unsigned 8	RW	см. канал CH0
	03H	Единицы измерения CH2	Unsigned 8	RW	см. канал CH0
	04H	Единицы измерения CH3	Unsigned 8	RW	см. канал CH0
8001-81F1H	00H	Разрешение работы каналов			Нумерация модулей 0-31 0x8001-0x81F1H
	01H	Канал CH0	Unsigned 8	RW	Канал 0: 0:OFF 1:ON ; Заводское значение: 1
	02H	Канал CH1	Unsigned 8	RW	см. канал CH0
	03H	Канал CH2	Unsigned 8	RW	см. канал CH0
	04H	Канал CH3	Unsigned 8	RW	см. канал CH0
8002-81F2H	00H	Выбор типа датчика			Нумерация модулей 0-31 0x8002-0x81F2H
	01H	Канал CH0	Unsigned 8	RW	Канал 0 0:Pt100 1: Pt1000 2:JPt100 3:Ni100 4: Ni1000

					5:LG- Ni1000 6:Cu50 7: Cu100 8:0~300Ω 9:0~3000Ω Заводское значение: 0
	02H	Канал CH1	Unsigned 8	RW	см. канал CH0
	03H	Канал CH2	Unsigned 8	RW	см. канал CH0
	04H	Канал CH3	Unsigned 8	RW	см. канал CH0
8003-81F3H	00H	Количество замеров выборки осреднения			Нумерация модулей 0-31 0x8003-0x81F3H
	01H	Канал CH0	Unsigned 8	RW	Канал 0: 1~100 Заводское значение: 5
	02H	Канал CH1	Unsigned 8	RW	см. канал CH0
	03H	Канал CH2	Unsigned 8	RW	см. канал CH0
	04H	Канал CH3	Unsigned 8	RW	см. канал CH0
8004-81F4H	00H	Определение превышения порога температуры			Нумерация модулей 0-31 0x8004-0x81F4H
	01H	Канал CH0	Unsigned 8	RW	Канал 0: 0: Порог по умолчанию 1: Порог установленный
	02H	Канал CH1	Unsigned 8	RW	см. канал CH0
	03H	Канал CH2	Unsigned 8	RW	см. канал CH0
	04H	Канал CH3	Unsigned 8	RW	см. канал CH0
8005-81F5H	00H	Установка порогов температур для аварии по выходу за диапазон			Нумерация модулей 0-31 0x8005-0x81F5H
	01H	Канал CH0 нижний предел	Signed 16	RW	Канал 0: Порог по умолчанию 1500 (-150°C)
	02H	Канал CH0 верхний предел	Signed 16	RW	Канал 0: Порог по умолчанию 18000 (1800°C)
	03H	Канал CH1 нижний предел	Signed 16	RW	Канал 0: Порог по умолчанию 1500 (-150°C)
	04H	Канал CH1 верхний предел	Signed 16	RW	Канал 0: Порог по умолчанию 18000 (1800°C)
	05H	Канал CH2 нижний предел	Signed 16	RW	Канал 0: Порог по умолчанию 1500 (-150°C)
	06H	Канал CH2 верхний предел	Signed 16	RW	Канал 0: Порог по умолчанию 18000 (1800°C)
	07H	Канал CH3 нижний предел	Signed 16	RW	Канал 0: Порог по умолчанию 1500 (-150°C)
	08H	Канал CH3 верхний предел	Signed 16	RW	Канал 0: Порог по умолчанию 18000 (1800°C)
8006-81F6H	00H	Проверка обрыва датчика			Нумерация модулей 0-31 0x8006-0x81F6H
	01H	Канал CH0	Signed 8	RW	Канал 0: 0: Выключена 1: Включена Заводское значение: 1
	02H	Канал CH1	Signed 8	RW	см. канал CH0
	03H	Канал CH2	Signed 8	RW	см. канал CH0
	04H	Канал CH3	Signed 8	RW	см. канал CH0
	05H	Отображение значения при обрыве датчика	Signed 8	RW	0: 32767 1: –32767 Заводское значение: 0
8007-81F7H	00H	Сохранение, сброс, цикл			Нумерация модулей 0-31 0x8007-0x81F7H
	01H	Цикл опроса входов	Signed 8	RW	0:250ms 1:500ms 2:1000ms Заводское значение: 1

	02H	Сохранение параметров модуля	Signed 8	RW	Запишите 1
	03H	Сброс на заводские установки	Signed 8	RW	Запишите 1
8008-81F8H	00H	Установка смещения нуля			Нумерация модулей 0-31 0x8008-0x81F8H
	01H	Канал CH0	Unsigned 8	RW	Заводское значение: 0
	02H	Канал CH1	Unsigned 8	RW	Заводское значение: 0
	03H	Канал CH2	Unsigned 8	RW	Заводское значение: 0
	04H	Канал CH3	Unsigned 8	RW	Заводское значение: 0
800E-81FEH	00H	Коэффициенты ПИД регулятора			Нумерация модулей 0-31 0x800E-0x81FEH
	01H	Канал CH0 Kp	Signed 32	RW	Пропорциональный коэф.
	02H	Канал CH0 Ti	Signed 32	RW	Интегральный коэф.
	03H	Канал CH0 Td	Signed 32	RW	Дифференциальный коэф.
	04H	Канал CH0 Cycle	Signed 32	RW	Цикл ПИД регулятора (сек)
	05H	Канал CH1 Kp	Signed 32	RW	Пропорциональный коэф.
	06H	Канал CH1 Ti	Signed 32	RW	Интегральный коэф.
	07H	Канал CH1 Td	Signed 32	RW	Дифференциальный коэф.
	08H	Канал CH1 Cycle	Signed 32	RW	Цикл ПИД регулятора (сек)
	09H	Канал CH2 Kp	Signed 32	RW	Пропорциональный коэф.
	0AH	Канал CH2 Ti	Signed 32	RW	Интегральный коэф.
	0BH	Канал CH2 Td	Signed 32	RW	Дифференциальный коэф.
	0CH	Канал CH2 Cycle	Signed 32	RW	Цикл ПИД регулятора (сек)
	0DH	Канал CH3 Kp	Signed 32	RW	Пропорциональный коэф.
	0EH	Канал CH3 Ti	Signed 32	RW	Интегральный коэф.
	0FH	Канал CH3 Td	Signed 32	RW	Дифференциальный коэф.
	10H	Канал CH3 Cycle	Signed 32	RW	Цикл ПИД регулятора (сек)
800F-81FFH	00H	Уставка и управление ПИД регулятором			Нумерация модулей 0-31 0x800F-0x81FFH
	01H	Канал CH0 Уставка	Signed 16	RW	Задание температуры
	02H	Канал CH1 Уставка	Signed 16	RW	Задание температуры
	03H	Канал CH2 Уставка	Signed 16	RW	Задание температуры
	04H	Канал CH3 Уставка	Signed 16	RW	Задание температуры
	05H	Канал CH0 Пуск/Стоп ПИД	Unsigned 8	RW	0: Стоп 1:Работа Заводское значение: 0
	06H	Канал CH0 Автонастройка	Unsigned 8	RW	0: Стоп 1:Начать автонастройку Заводское значение: 0
	07H	Канал CH0 Полярность ПИД	Unsigned 8	RW	0: Униполярный 1: Биполярный Заводское значение: 0
	08H	Канал CH1 Пуск/Стоп ПИД	Unsigned 8	RW	0: Стоп 1:Работа Заводское значение: 0
	09H	Канал CH1 Автонастройка	Unsigned 8	RW	0: Стоп 1:Начать автонастройку Заводское значение: 0
	0AH	Канал CH1 Полярность ПИД	Unsigned 8	RW	0: Униполярный 1: Биполярный Заводское значение: 0
	0BH	Канал CH2 Пуск/Стоп ПИД	Unsigned 8	RW	0: Стоп 1:Работа Заводское значение: 0
	0CH	Канал CH2 Автонастройка	Unsigned 8	RW	0: Стоп 1:Начать автонастройку Заводское значение: 0
	0DH	Канал CH2 Полярность ПИД	Unsigned 8	RW	0: Униполярный 1: Биполярный Заводское значение: 0

	0EH	Канал CH3 Пуск/Стоп ПИД	Unsigned 8	RW	0: Стоп 1:Работа Заводское значение: 0
	0FH	Канал CH3 Автонастройка	Unsigned 8	RW	0: Стоп 1:Начать автонастройку Заводское значение: 0
	10H	Канал CH3 Полярность ПИД	Unsigned 8	RW	0: Униполярный 1: Биполярный Заводское значение: 0

Объекты EtherCAT модуля MX04RC (TxPDO) (данные процесса)

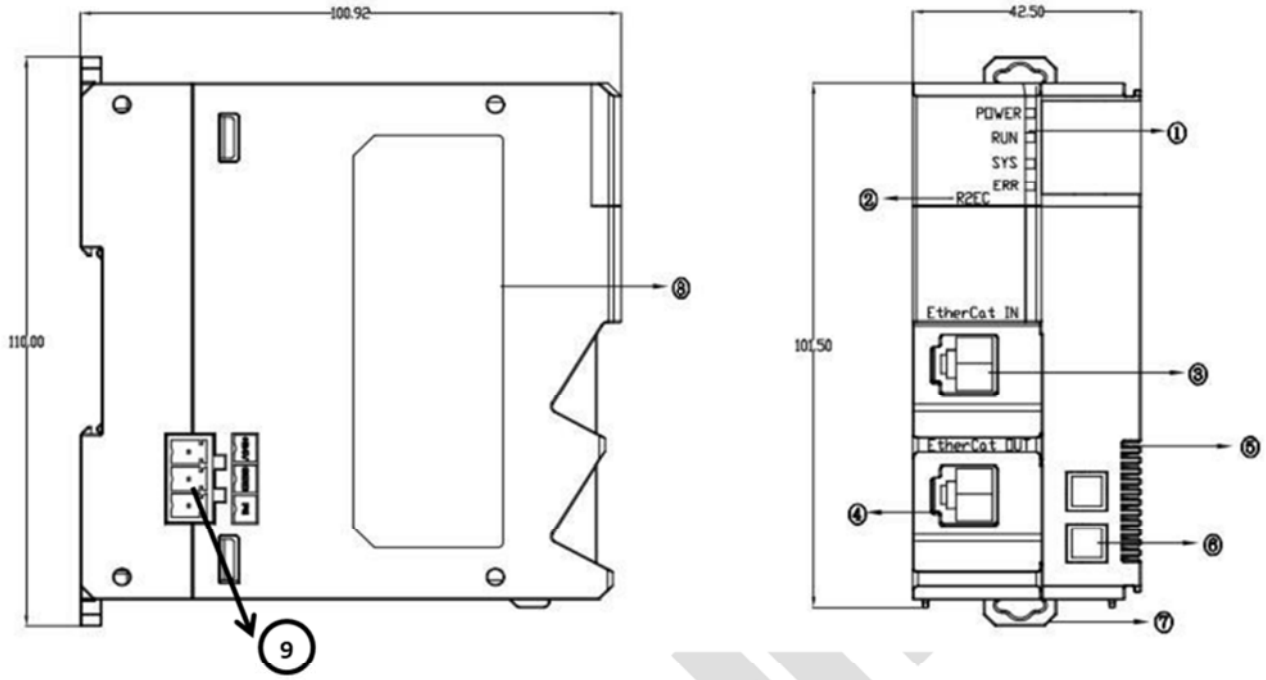
Индекс	Субиндекс	Содержание	Тип данных	Доступ	Описание
6000~61F0H	00H	Измеренные значения на каналах			На шине может быть до 32-х модулей. Нумерация 0-31. Индекс объекта EtherCAT будет нумероваться с шагом 1: 0x6000 – 0x61F0
	01H	Канал CH0	Signed 16	RO	Текущая температура (/10)
	02H	Канал CH1	Signed 16	RO	Текущая температура (/10)
	03H	Канал CH2	Signed 16	RO	Текущая температура (/10)
	04H	Канал CH3	Signed 16	RO	Текущая температура (/10)
A000~A1F0H	00H	Статус каналов			Нумерация модулей 0-31 0xA000-0xA1F0H
	01H	Канал CH0	Unsigned 8	RO	Канал 0 Состояние 0x00: Нормальное 0x01: Режим автонастройки, ожидание первого стандартного значения 0x02: Ожиданис второго стандартного значения 0x03: Вычисление коэффициентов 0x04: Вычисление коэффициентов закончено 0x10:Превышение температуры 0x20: Обрыв датчика 0x40: Ошибка записи предела 0x80: Ошибка смещения нуля температуры 0xFF: Канал отключен Если возникает одновременно превышение температуры и обрыв датчика, то код ошибки будет 0x10+0x20=0x30
	02H	Канал CH1	Unsigned 8	RO	см. канал CH0
	03H	Канал CH2	Unsigned 8	RO	см. канал CH0
	04H	Канал CH3	Unsigned 8	RO	см. канал CH0
A001~A1F1H	00H	Состояние ПИД регулятора			Нумерация модулей 0-31 0xA001-0xA1F1H
	01H	Канал CH0	Unsigned 8	RO	Канал 0 Состояние ПИД рег. Bit0: Индикация работы 0: Стоп 1: Работа (1 когда ПИД в работе, 0 в режиме автонастройки, 1 когда автонастройка окончена) Bit1:Режим выхода 0: Нагрев

					1: Охлаждение Bit2: Автонастройка 0: Выключена 1: Процесс автонастройки (1 когда запущен процесс автонастройки, 0 когда закончена автонастройка) Bit3: Ошибка автонастройки 0: Нет ошибки 1: Ошибка автонастройки Bit4: heat_out 0:NO_Out 1:Tune_Out Bit5:cool_out 0:NO_Out 1:Tune_Out
	02H	Канал CH1	Unsigned 8	RO	см. канал CH0
	03H	Канал CH2	Unsigned 8	RO	см. канал CH0
	04H	Канал CH3	Unsigned 8	RO	см. канал CH0

Станция удалённого ввода-вывода (каплер) R2EC для сети EtherCAT

Станция **R2EC** предназначена для сбора физических сигналов с модулей дискретного, аналогового и температурного ввода-вывода и передачи данных по сети EtherCAT любому стандартному Мастеру сети EtherCAT. Используются те же модули, что и для контроллеров серии MX300.

Внешний вид и размеры станции R2EC



Габаритные размеры: ВхШхГ: 101.50 x 42.50 x 100.92 мм

Питание 24 VDC

1	Светодиоды: RUN: работа (зеленый), ERR: ошибка (красный)
2	Название модели
3	Разъём RJ45 EtherCAT IN
4	Разъём RJ45 EtherCAT OUT
5	Разъем подключения модуля расширения
6	Позиционные переключатели
7	Крепление на ДИН-рейку
8	Шильдик
9	Разъём питания 24 VDC

Объекты EtherCAT станции R2EC (SDO) (настройки)

Индекс	Субиндекс	Содержание	Тип данных	Доступ	Описание
1000H	00H	Тип устройства	Unsigned 32		Тип и профиль устройства: 0x00001389
1008H	00H	Имя устройства	String 8	RO	Наименование продукта: R2EC
1009H	00H	Версия аппаратной части	String 8	RO	Обозначение типа: $02
100AH	00H	Версия встроенного ПО	String 8	RO	Обозначение типа: $02
1018H		Идентификационная информация			Описание устройства
	00H	Largest sub-index	Unsigned 8	RO	Максимальный субиндекс – 04H
	01H	Vendor ID	Unsigned 32	RO	Идентификатор производителя
	02H	Product code	Unsigned 32	RO	Код продукта
	03H	Revision	Unsigned 32	RO	Версия файла с описанием
	04H	Serial number	Unsigned 32	RO	Серийный номер
F030H	00H	Количество и ID модулей на станции	Unsigned 8	R/W	Количество сконфигурированных модулей на станции
	01H	Модуль № 1		RO	Идентификатор типа модуля
	…	…	…	RO	Идентификатор типа модуля
	32H	Модуль № 32		RO	Идентификатор типа модуля
A000~A1F0H	00H	Состояние модулей ввода-вывода на станции		RO	Нумерация модулей 0-31 0xA001-0xA1F1H
	01H	Код ошибки	Unsigned 8	RO	0x0001: модуль отсутствует (отключен)

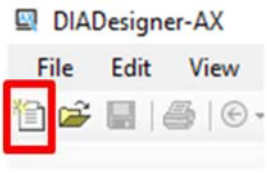
Объекты EtherCAT станции R2EC (TxPDO) (данные процесса)

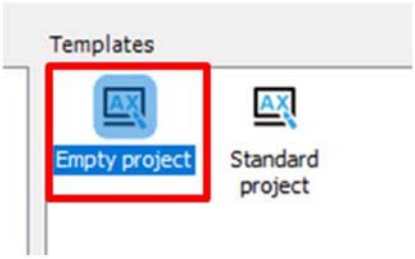
Индекс	Субиндекс	Содержание	Тип данных	Доступ	Описание
F100H	00H	Статус станции			Состояние устройства
	01H	Состояние модулей на станции	Unsigned 32	RO	Каждый бит регистра представляет состояние подключенного к станции модуля от 1 до 32 (bit0-bit31) 0: Модуль работает нормально 1: Модуль в состоянии аварии

Запуск среды программирования и создание проекта

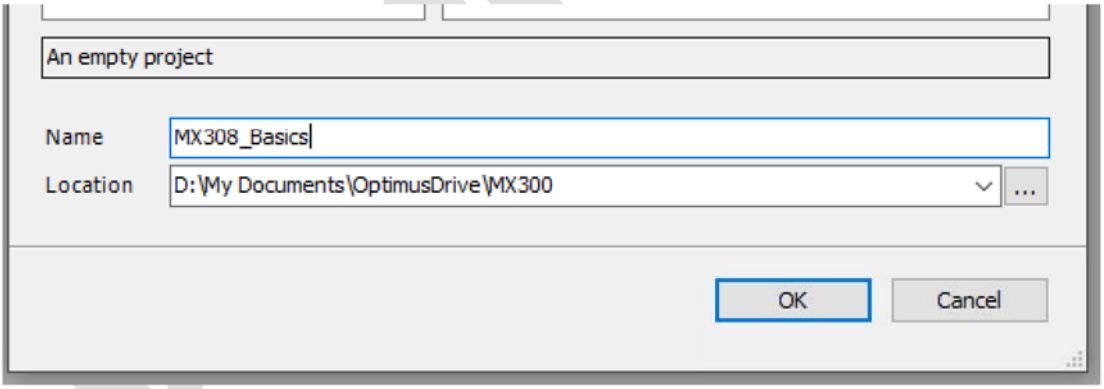
Установите на ПК (ноутбук) среду программирования DesignerAX следуя командам Мастера установки. После установки на рабочем столе появится иконка:

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»**.

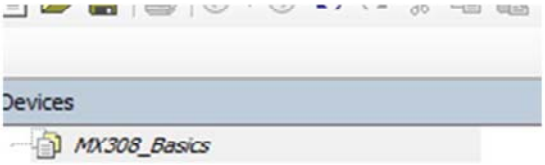




Дайте название проекту:



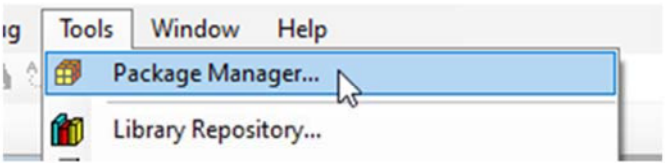
В открывшемся рабочем окне в верхнем левом углу древа проекта будет единственный пункт с названием проекта.



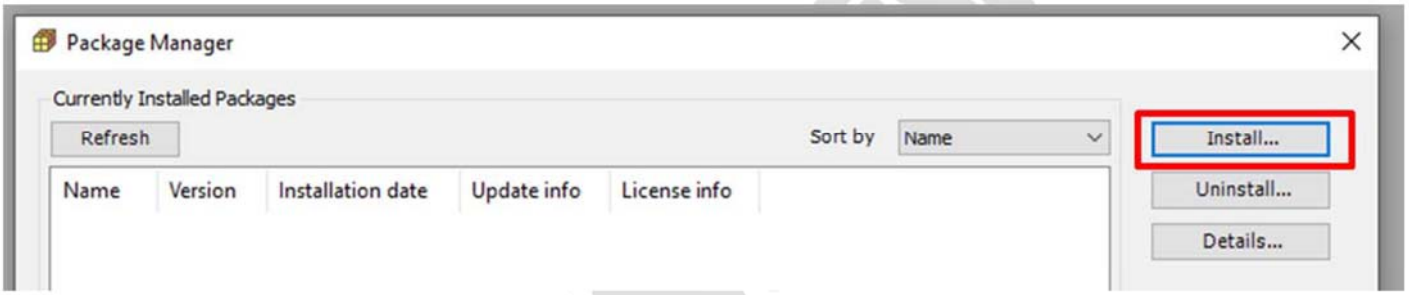
Установка описания устройства для контроллера MX300

Для начала работы с контроллером семейства MX300 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. Необходимо использовать актуальный пэкидж, который можно скачать с сайта **https://optimusdrive.ru**.

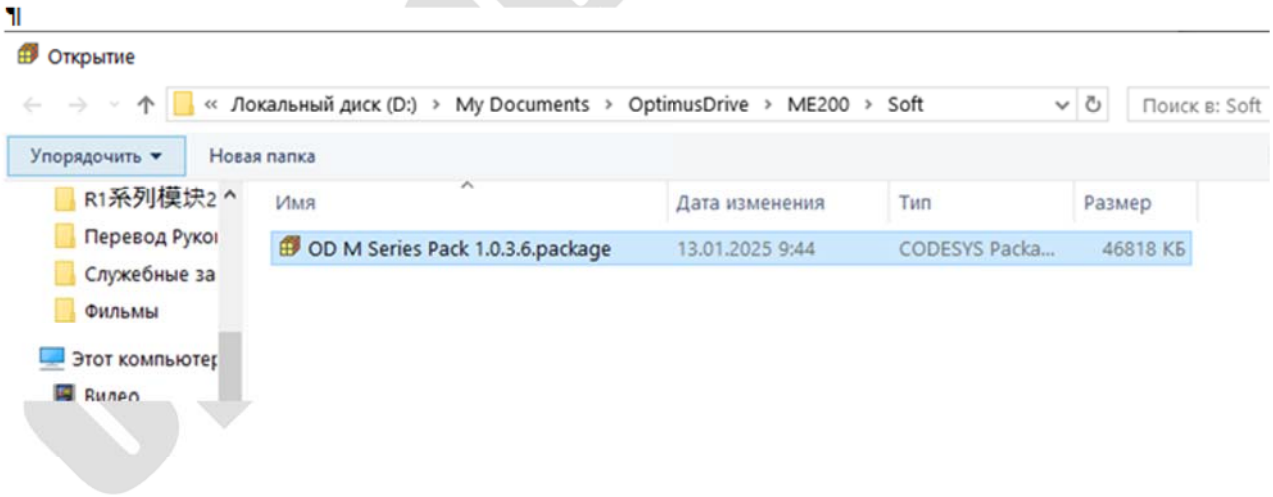
В созданном проекте войдите в меню установки пакетов устройств:



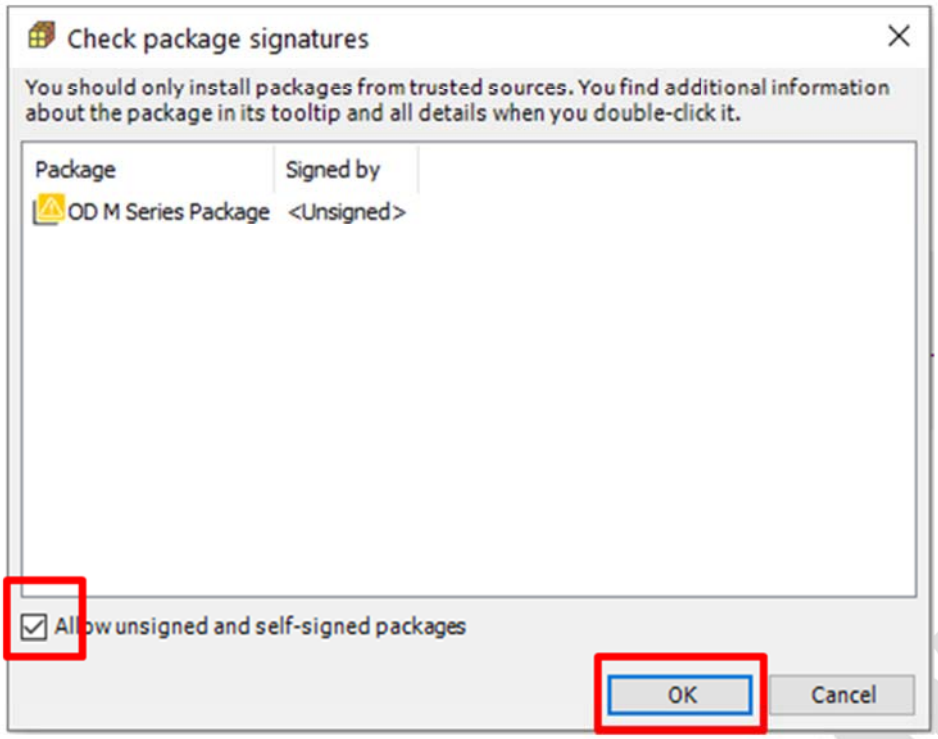
В открывшемся Мастере установки пакетов (Package Manager) нажмите кнопку **Install**:



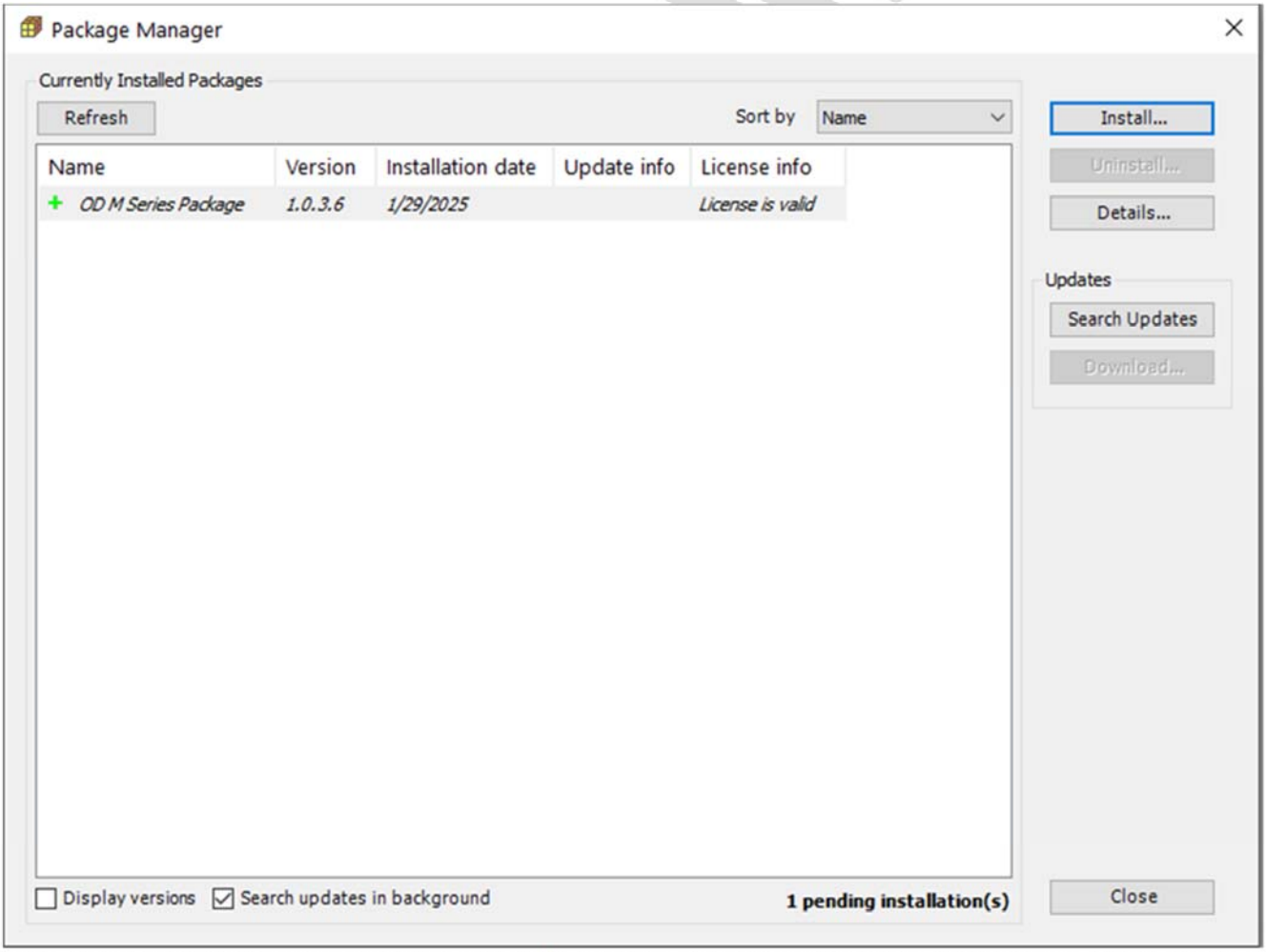
Выберите в проводнике нужный файл:



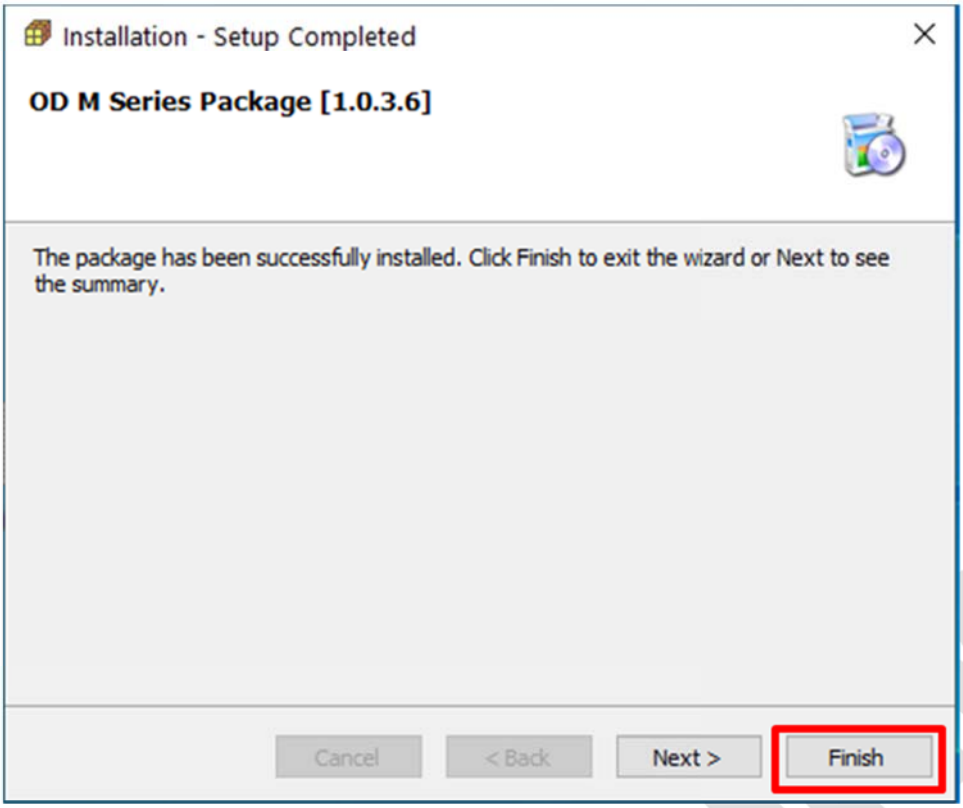
Поставьте флажок с разрешением работы без подписи файла:



Появится сообщение о готовности к установке пэкиджа:

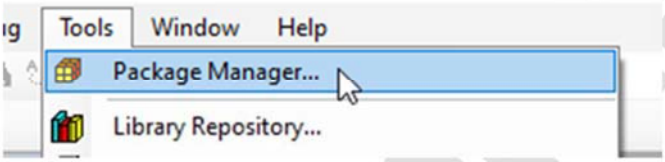


Закройте среду программирования DesignerAX, начнётся установка пэкиджа. Следуйте сообщениям на экране. По окончанию установки нажмите **Finish**:

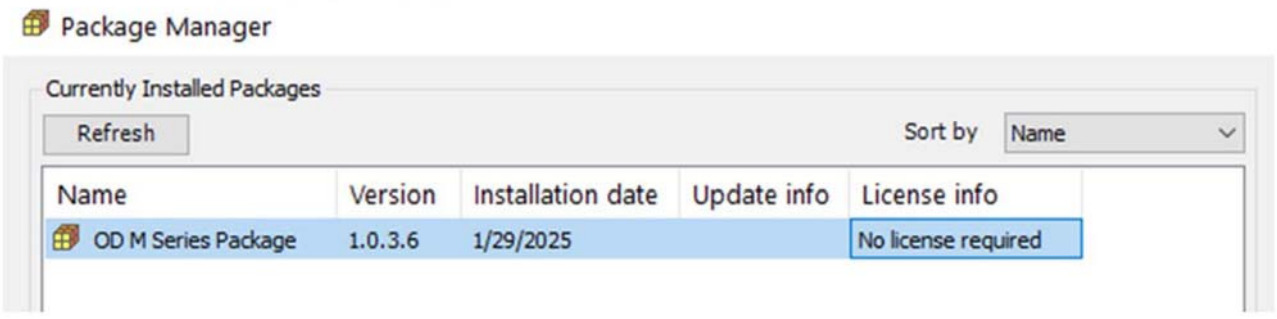


Запустите заново среду программирования DesignerAX.

Войдите в меню установки пакетов устройств:



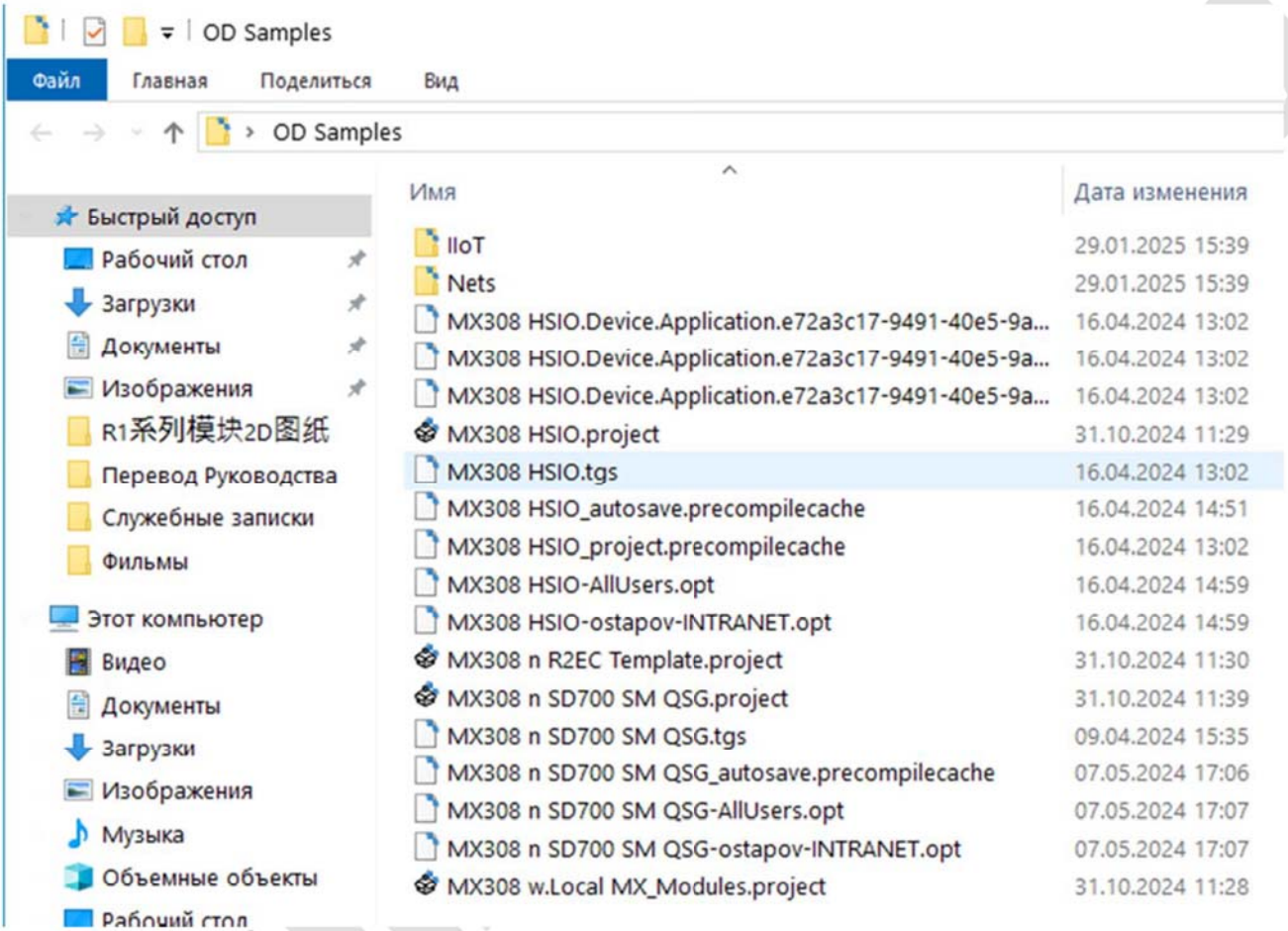
После установки появится запись в списке установленных пакетов:



Закройте менеджер пакетов.

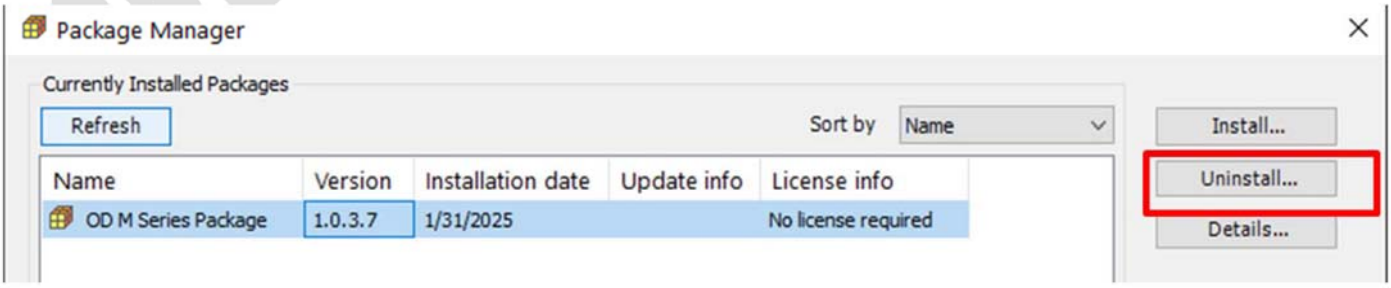
Закройте менеджер пакетов.
На этом установка пакета закончена и можно переходить к добавлению устройств в проект.

На рабочем столе появится папка с примерами:

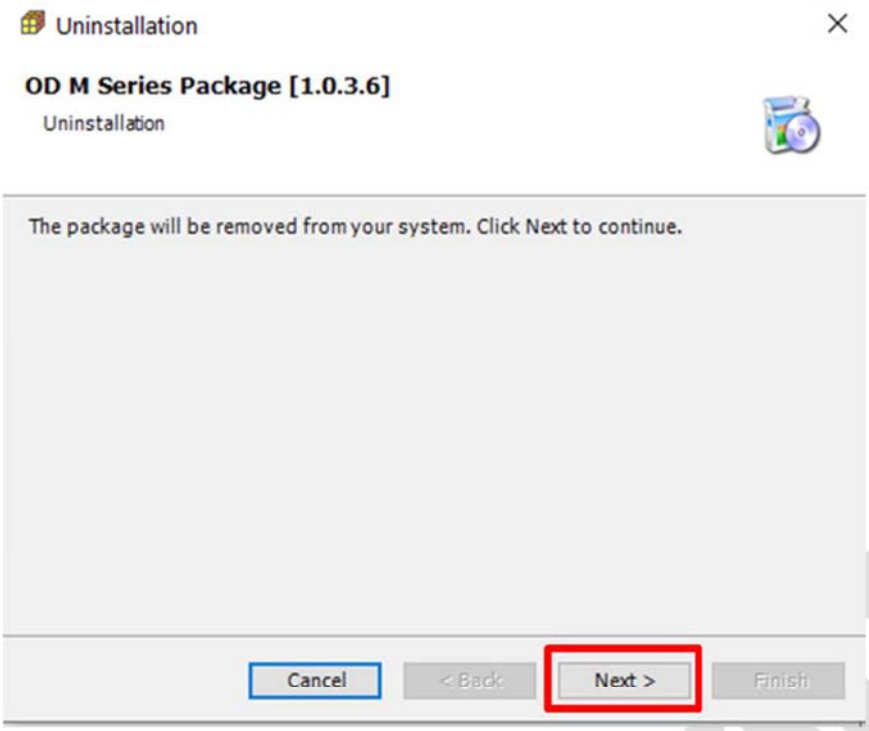


На этом установка пакета закончена и можно переходить к добавлению устройств в проект.

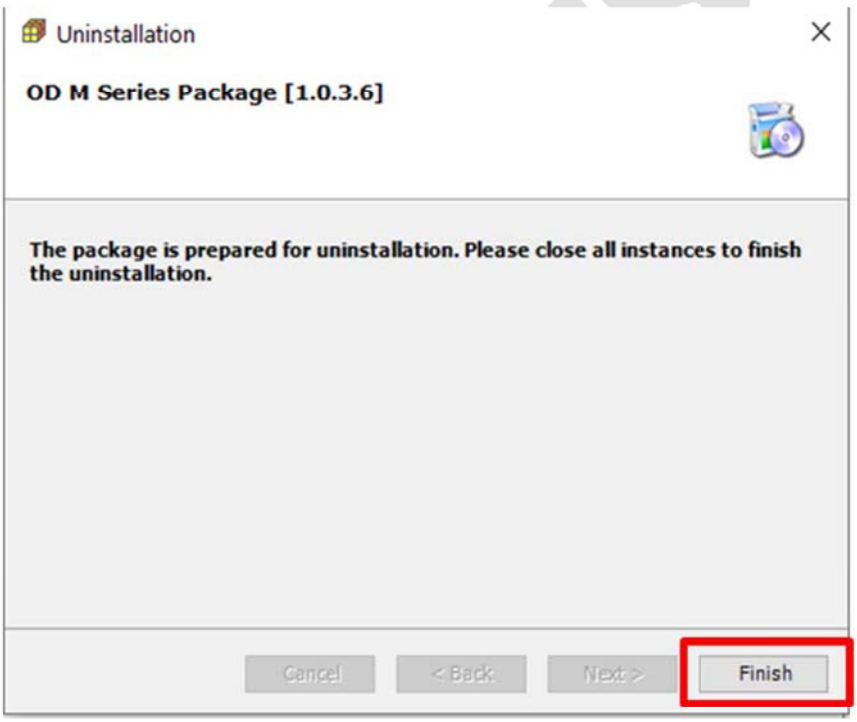
Для удаления пэкиджа необходимо нажать кнопку **Uninstall**:



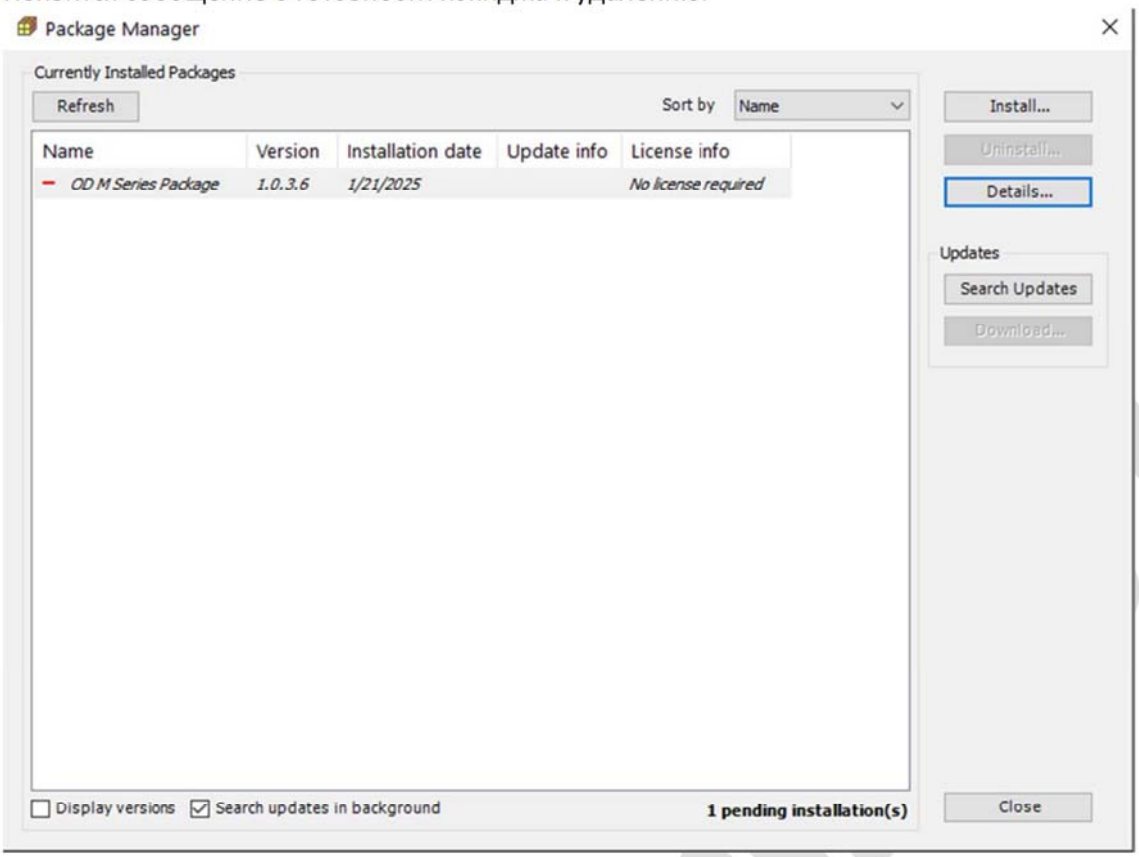
Далее **Next**:



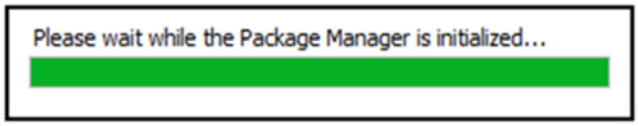
Далее **Finish**:



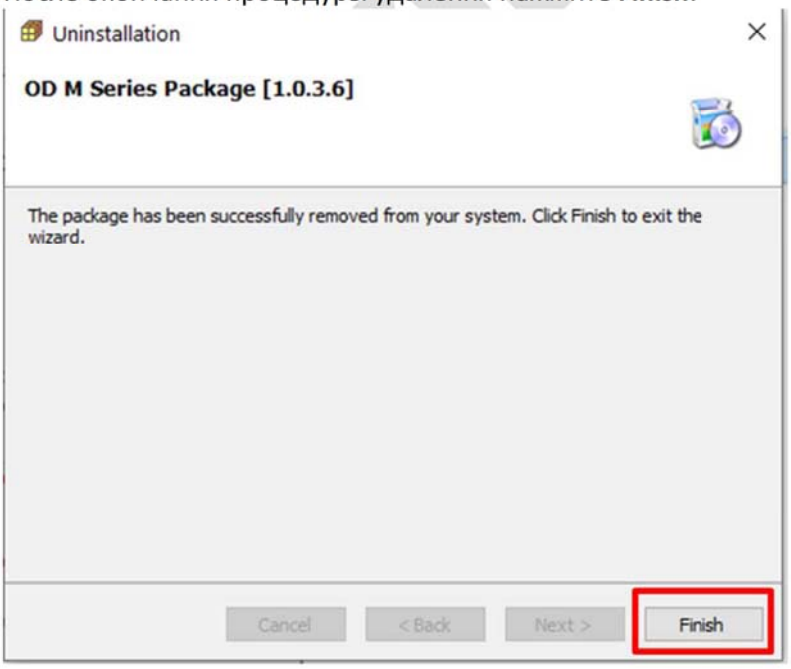
Появится сообщение о готовности пэкиджа к удалению:



После этого необходимо закрыть среду программирования. Начнётся удаление пэкиджа.



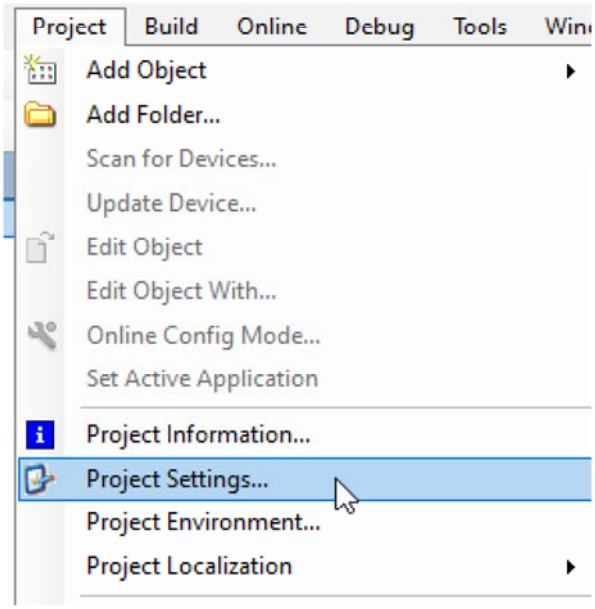
После окончания процедуры удаления нажмите **Finish**:



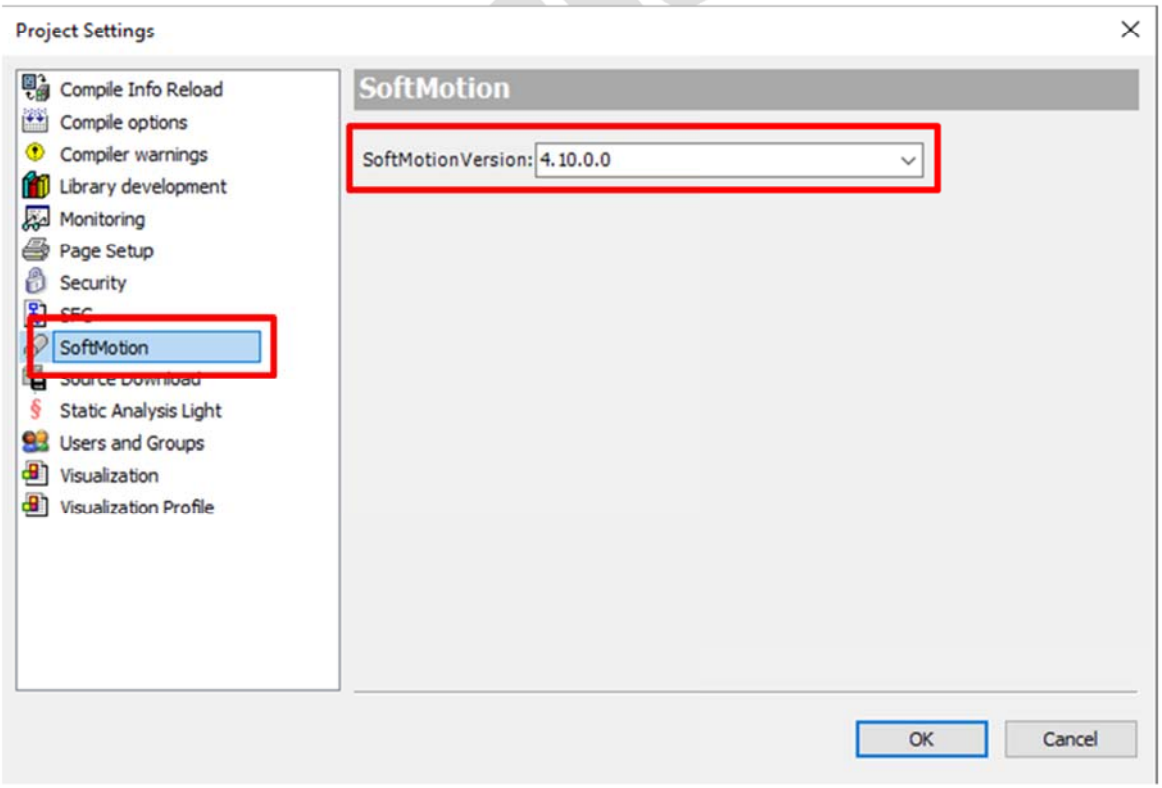
Определение версии библиотеки 3S SoftMotion (SM3)

Для корректной работы функций управления движением необходимо выбрать правильную версию библиотеки 3S SoftMotion. Для контроллеров MX300 используется версия 4.16.0.0. Но при необходимости можно выбрать и предыдущую версию 4.10.0.0. Для этого выполните следующие действия:

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



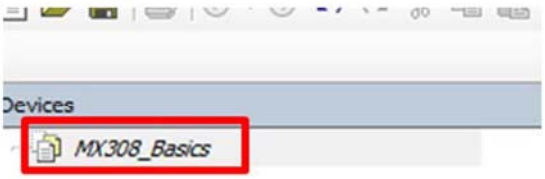
В открывшемся окне выберите пункт SoftMotion и версию 4.10.00:



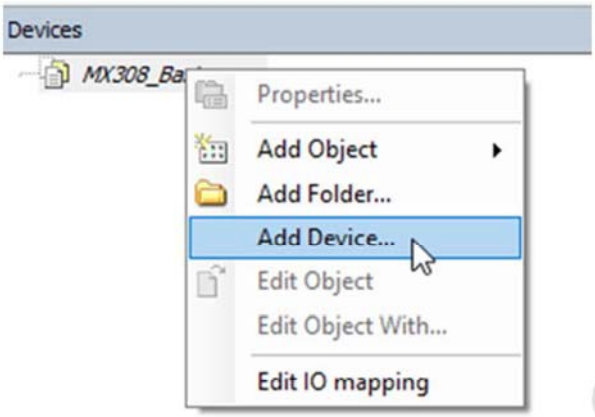
Добавление контроллера в проект

Для добавления контроллера в проект необходимо выполнить следующие действия.

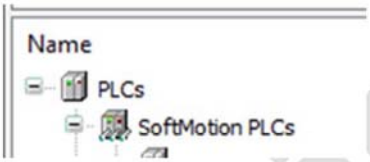
Щёлкните правой кнопкой мышки на названии проекта в левом верхнем углу проекта:



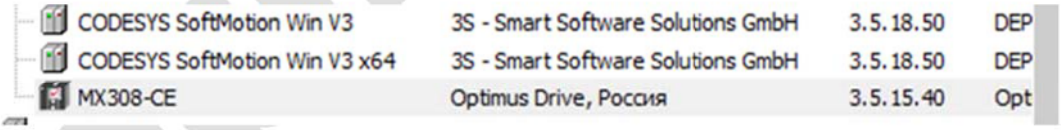
В появившемся меня выберите пункт **Add Device**:



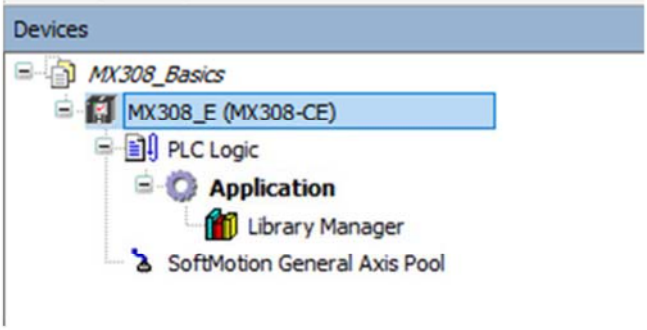
В открывшемся окне выберите раздел **PLC – SoftMotion PLC**:



Далее прокрутите вниз до пункта с контроллером MX300

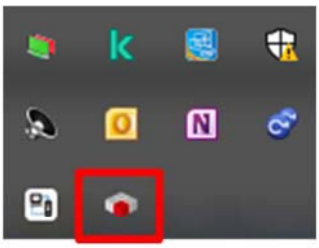


выберите его и нажмите кнопку Add Device и закройте окно. В древе проекта появится устройство – контроллер MX300:

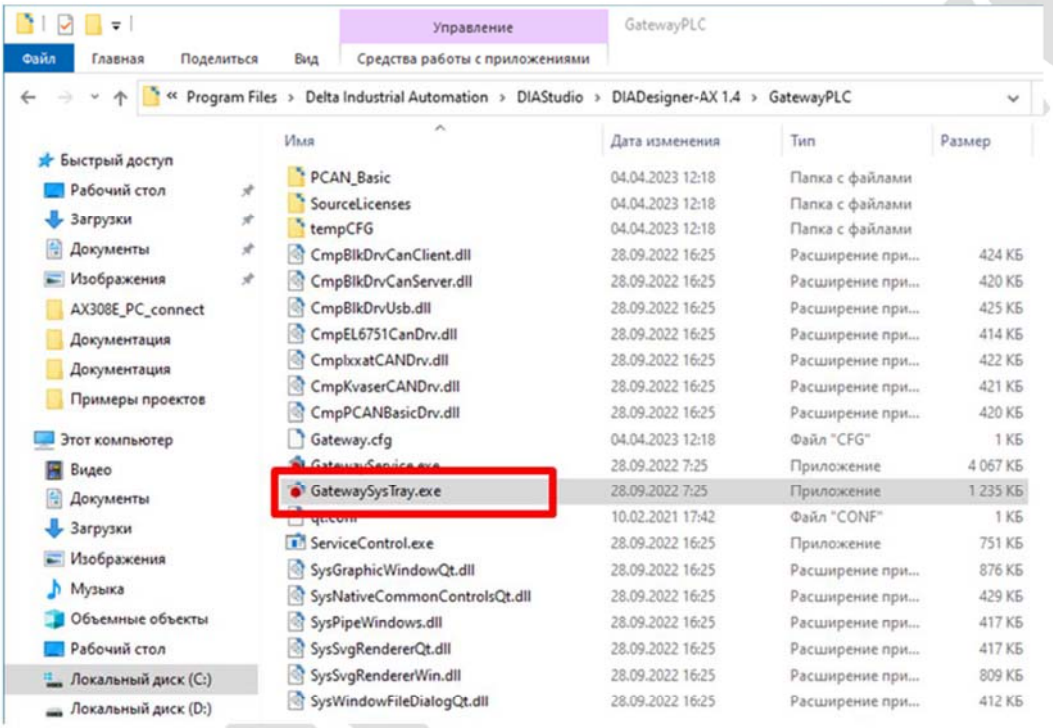


Организация связи контроллеров типа MX300 и среды программирования. Загрузка программы. Онлайн режим

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:

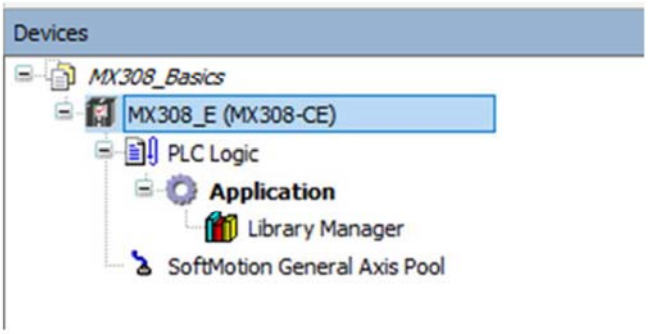


После этого должна появиться иконка.

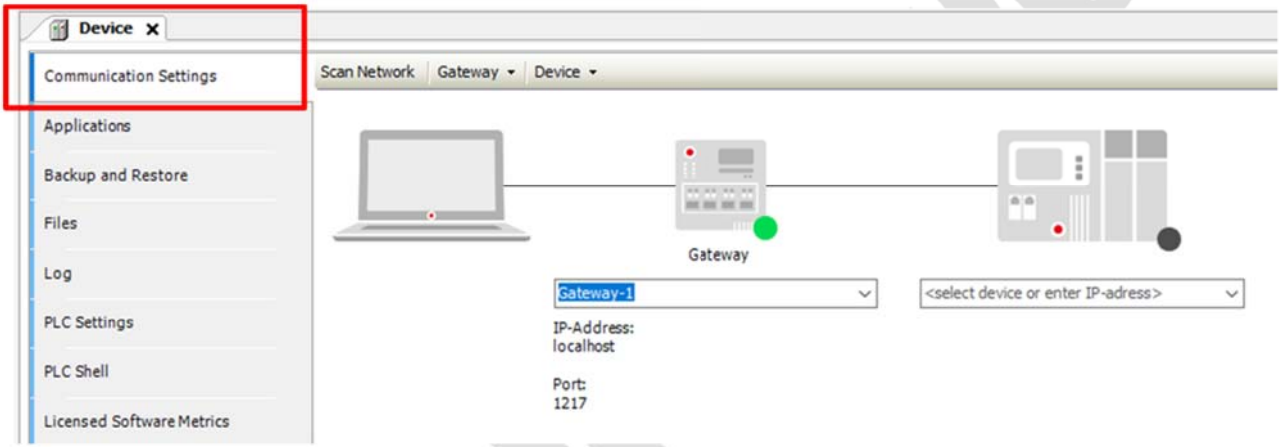
Если щёлкнуть на иконке шлюза левой кнопки мышки, то откроется меню, позволяющее включить/выключить шлюз и закрыть приложение:

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

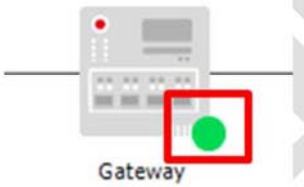
В древе проекта щёлкните дважды на названии контроллера:

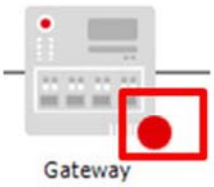


Откроется вкладка **Device** пункт **Communication Settings**:



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

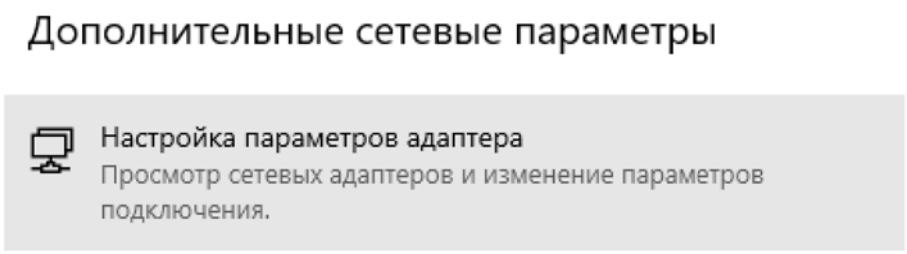




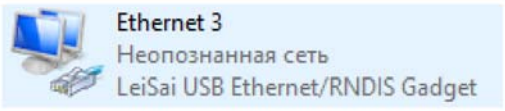
Соединение через порт USB-C

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:

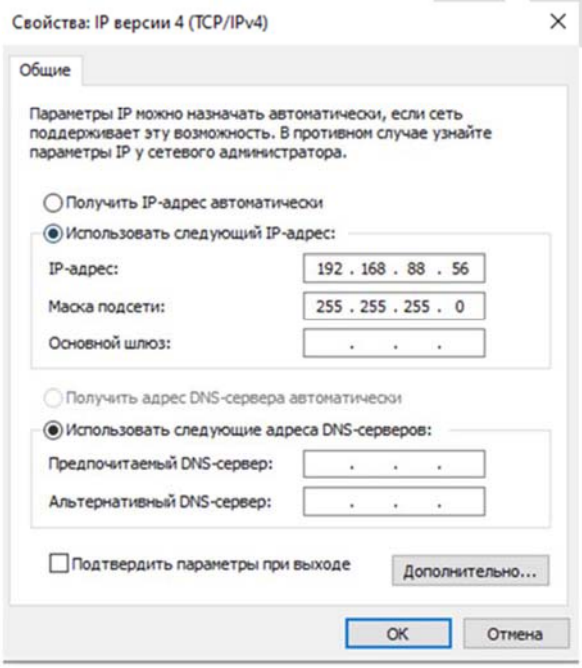


При подключении контроллера к ПК и правильно установленном драйвере должна появиться Неопознанная сеть данного вида:

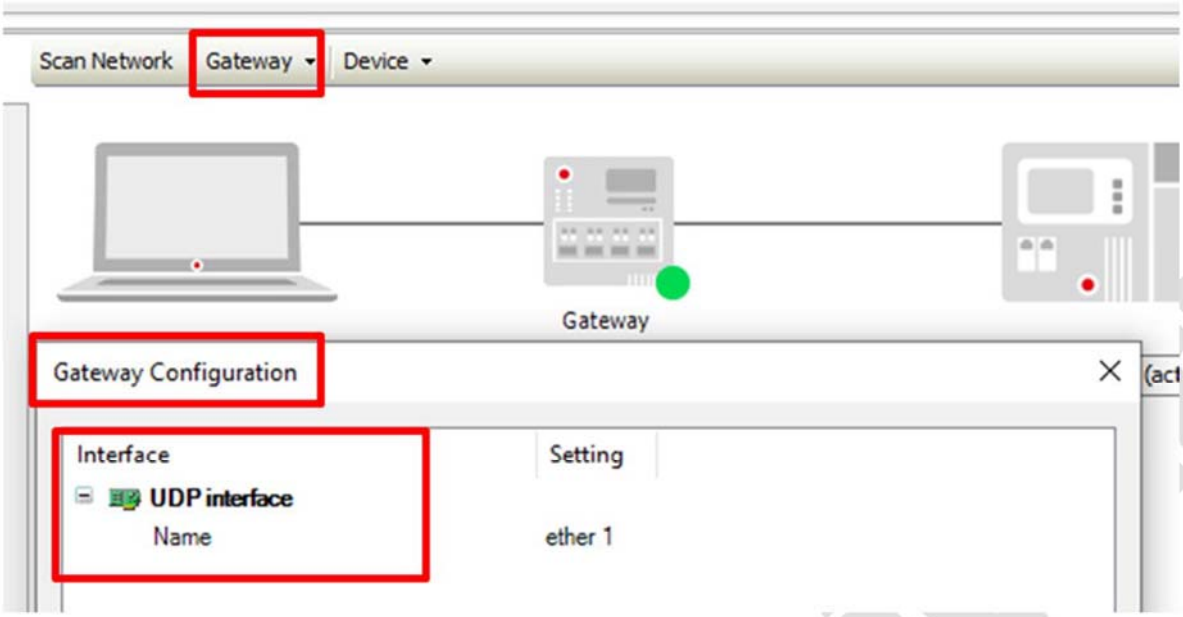


При соединении по USB сетевой контроллера будет фиксированный: **192.168.88.88**

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

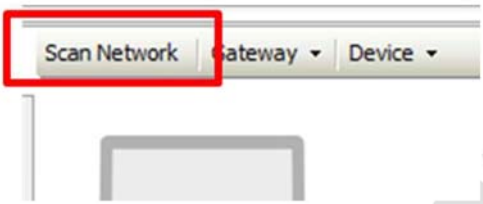


В общем случае для соединения достаточно драйвера по умолчанию:

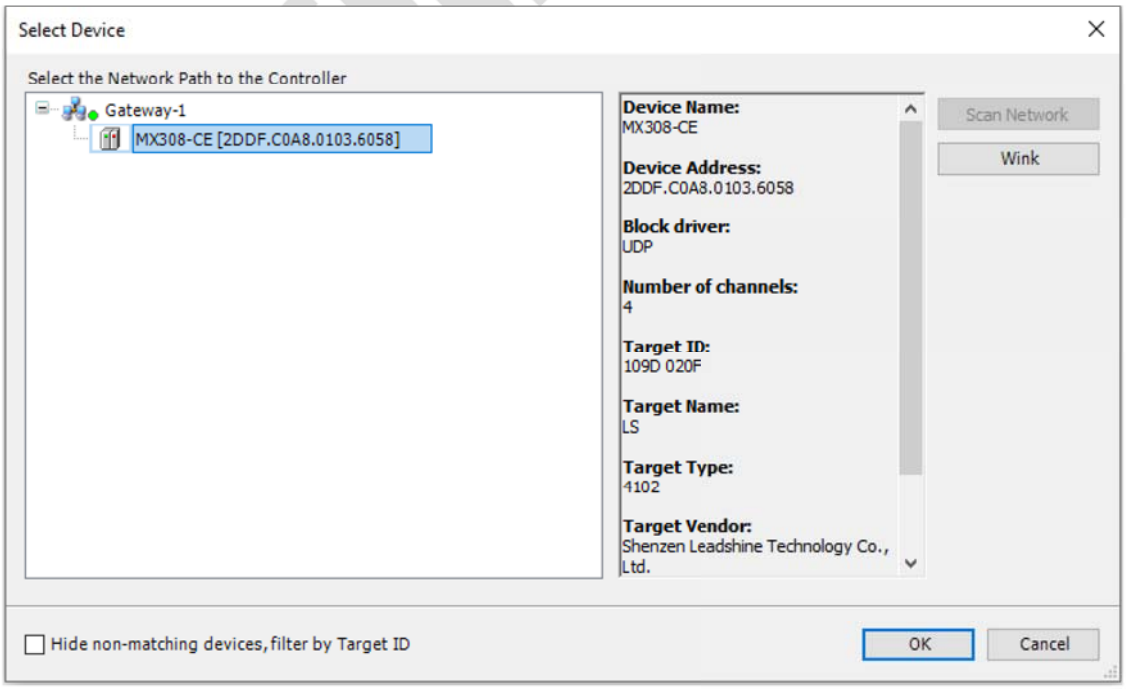


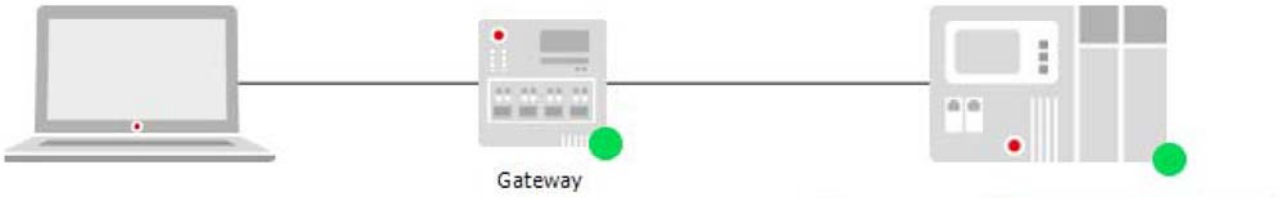
После этого можно приступать к установлению соединения среды программирования и контроллера.

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



Появится окно поиска и в итоге должен быть обнаружен контроллер:





Gateway-1

IP-Address:
localhost

Port:
1217

MX308-CE (active)

Device Name:
MX308-CE

Device Address:
2DDF.C0A8.0103.6058

Target ID:
109D 020F

Target Type:
4102

Target Vendor:
Shenzen Leadshine Technology Co., Ltd.

Target Version:
3.5.15.40

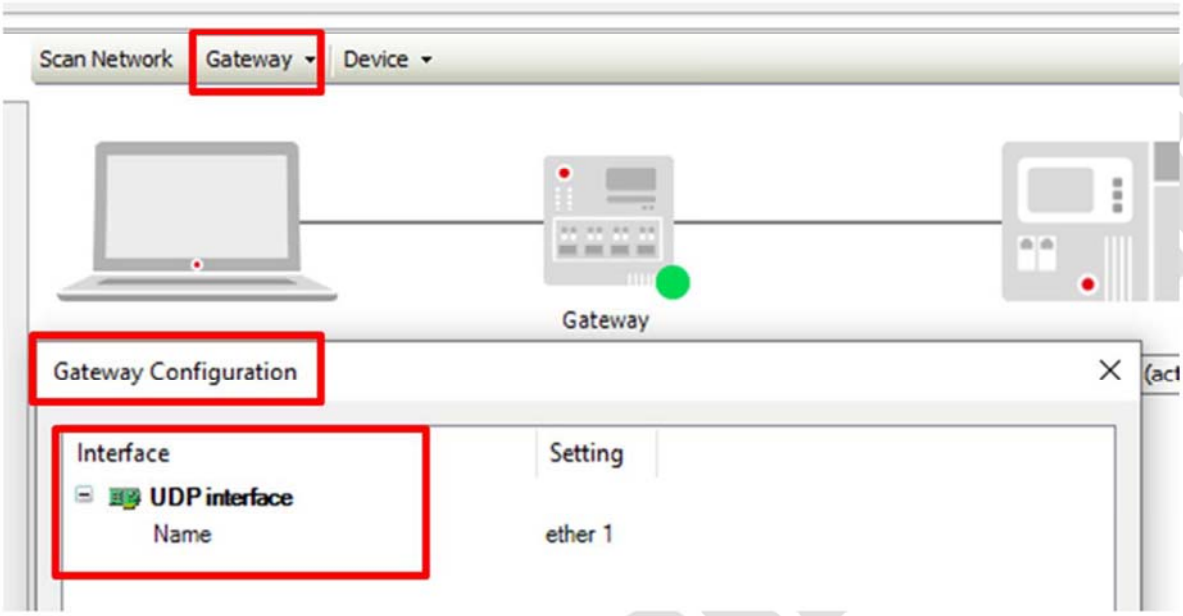
Внимание!
Если соединение не устанавливается, то можно сбросить в контроллере настройки связи на заводские. Для этого необходимо быстро включить-выключить переключатель RUN-STOP на передней панели контроллера (под крышкой). На пятый раз должны включиться примерно на 1 секунду все светодиоды рядом с переключателем. После данной процедуры контроллер будет стоять на заводских настройках до следующего отключения питания.

Соединение через порт Ethernet

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

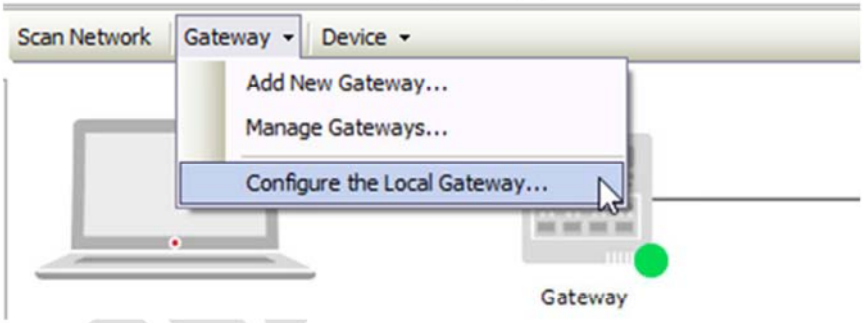
У контроллера IP адрес по умолчанию - **192.168.1.3**

В общем случае для соединения достаточно драйвера по умолчанию:

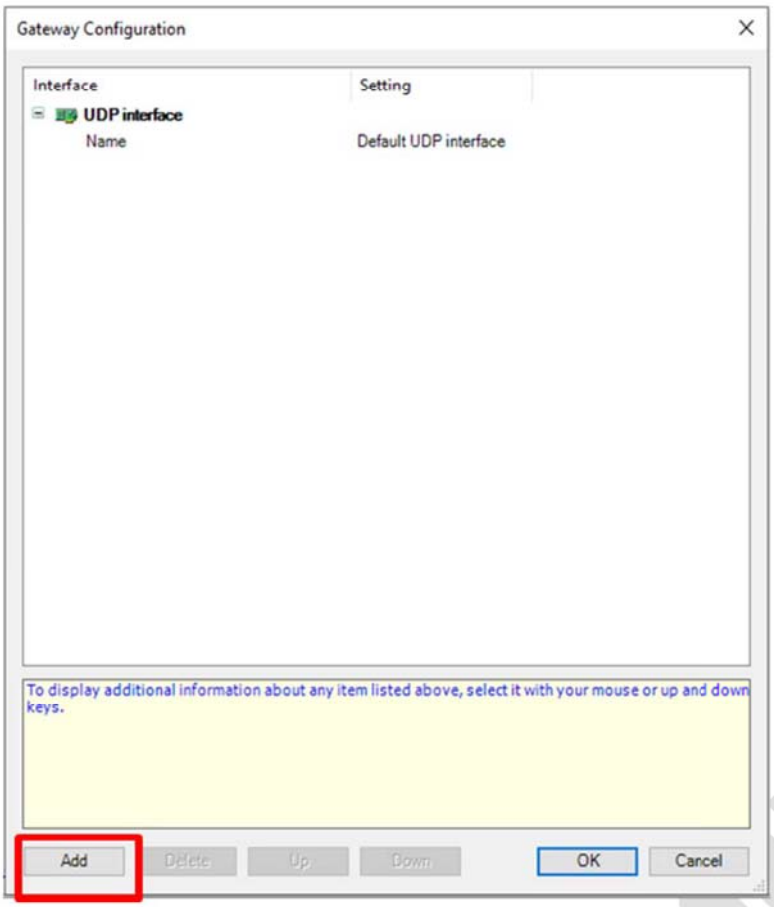


В случае соединения через несколько коммутаторов возможно потребуется создать явное TCP соединение.

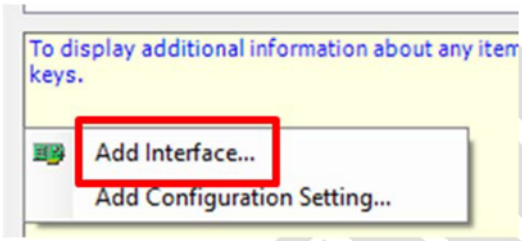
Для этого выберите пункт **Gateway – Configure the Local Gateway**:



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

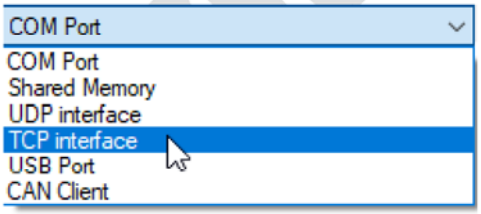


В левом нижнем углу нажмите кнопку **Add**, появится окно создания интерфейса:



Выберите пункт **Add Interface**.

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



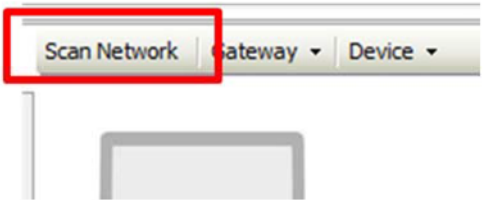
Щёлкните правой кнопкой мышки на надписи **TCP interface** и выберите пункт **Add Configuration Setting**:

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.

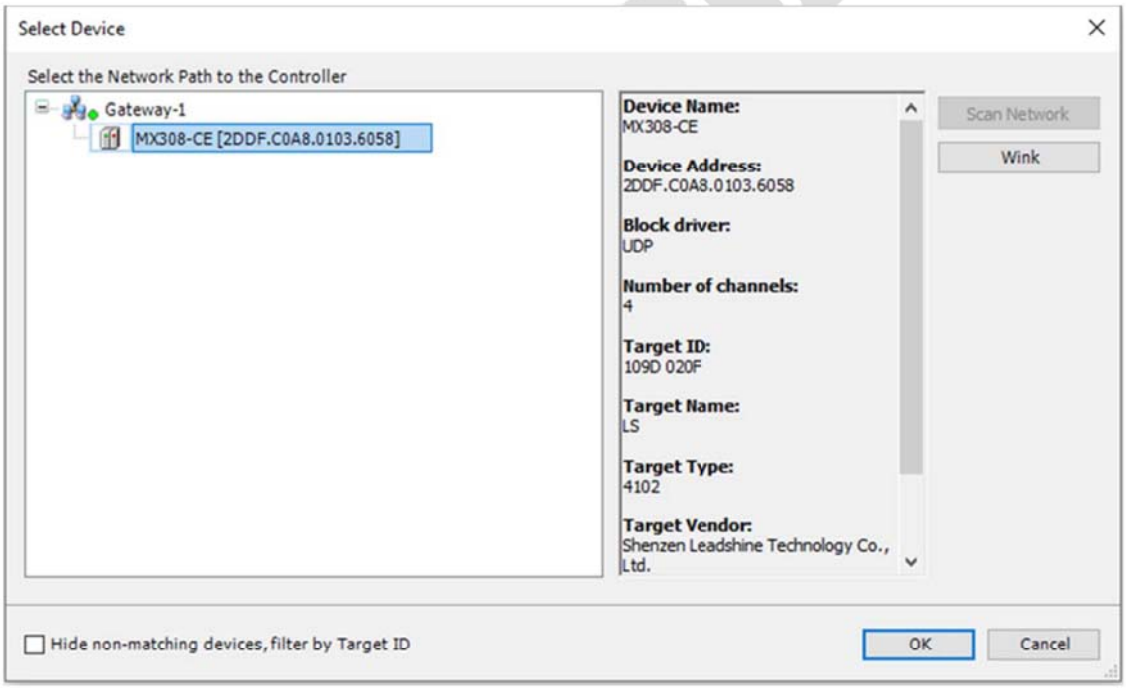
У контроллера IP адрес по умолчанию - **192.168.1.3**

Interface	Setting
TCP interface	
Port	11743
IP address	192.168.1.3
Inactivity timeout	60
Enable client	TRUE
Enable server	TRUE
Local access only	FALSE

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



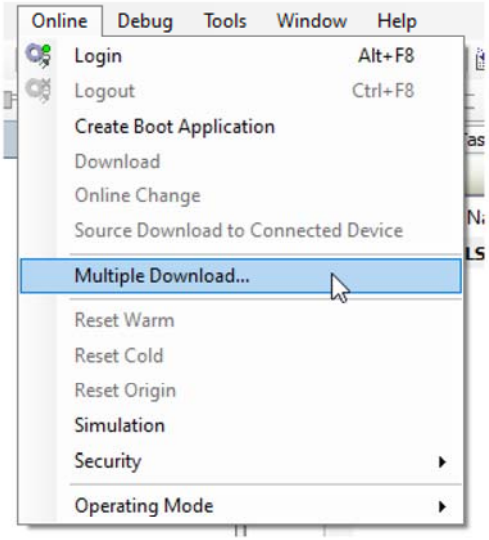
Появиться окно с поиском и найденным контроллером:



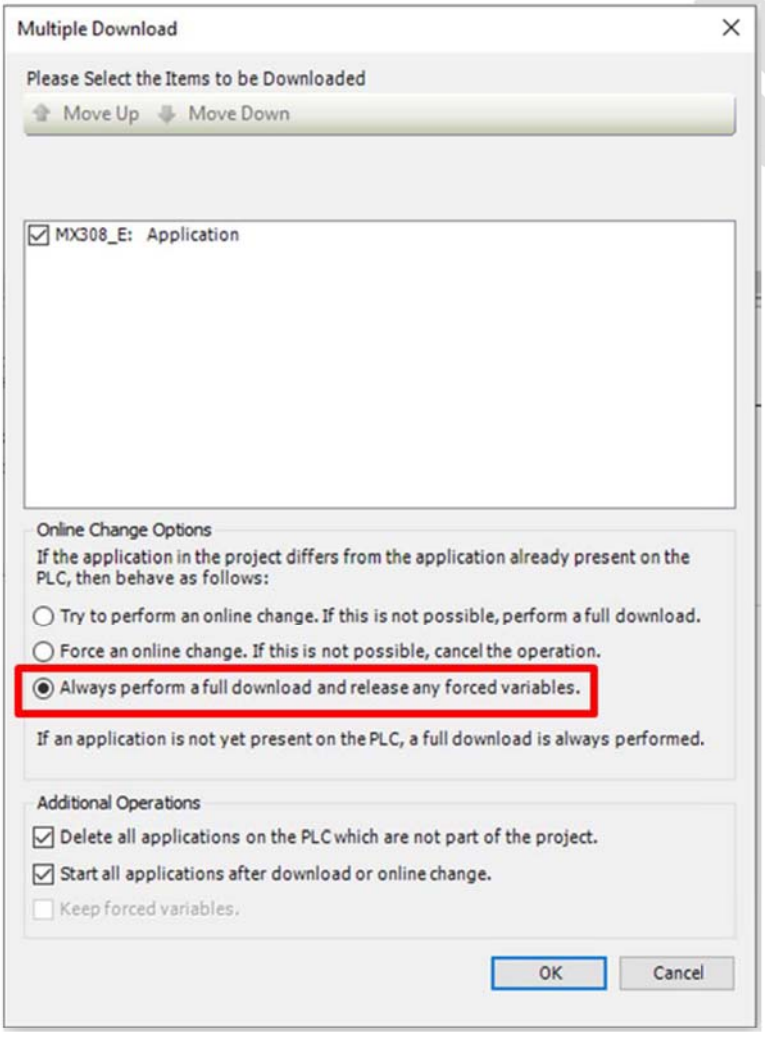
Внимание!
Если соединение не устанавливается, то можно сбросить в контроллере настройки связи на заводские. Для этого необходимо быстро включить-выключить переключатель RUN-STOP на передней панели контроллера (под крышкой). На пятый раз должны включиться примерно на 1 секунду все светодиоды рядом с переключателем. После данной процедуры контроллер будет стоять на заводских настройках до следующего отключения питания.

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:

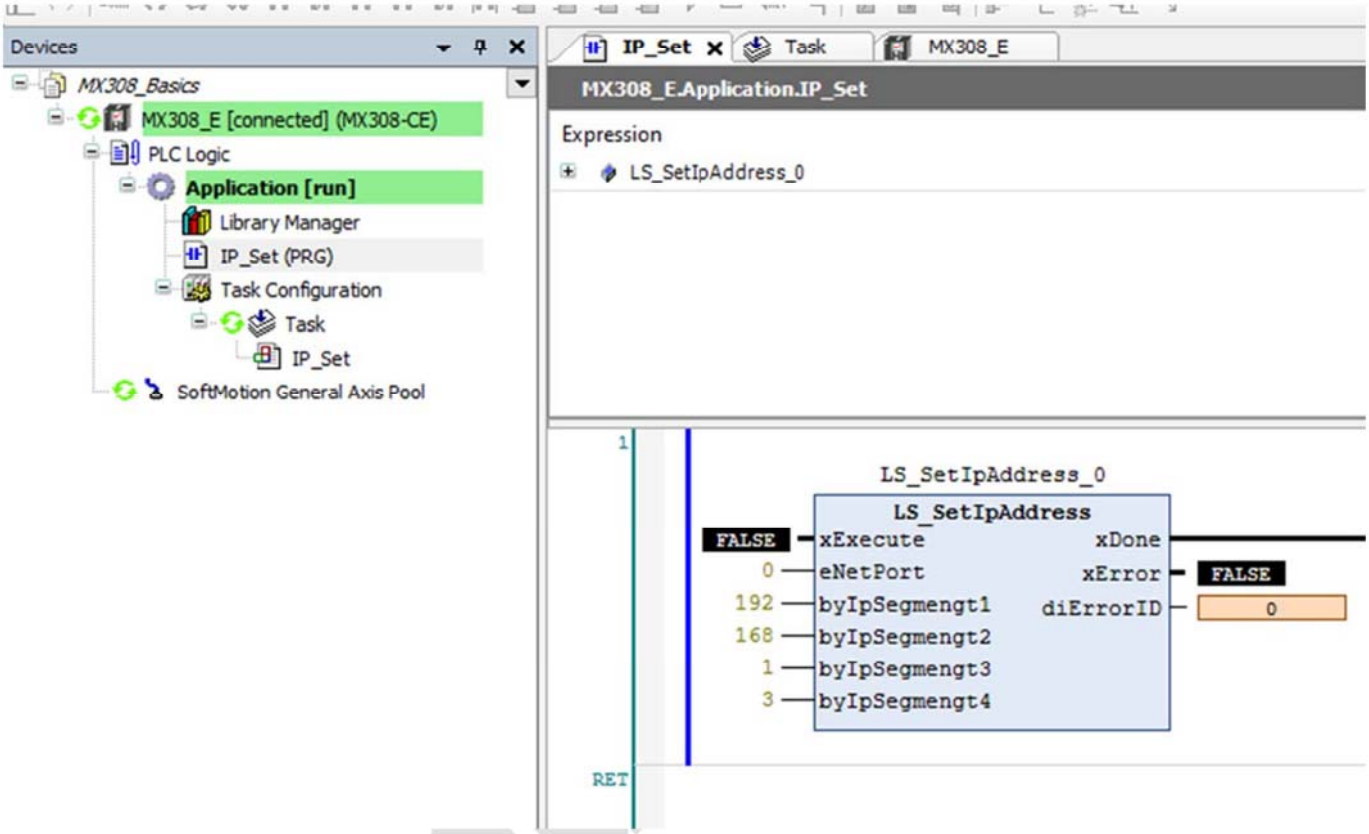


В открывшемся окне выберите вариант полной загрузки и нажмите **ОК**:

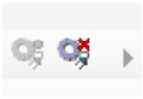


Для входа в режим онлайн нажмите кнопку:

Программа подсветится состоянием объектов:

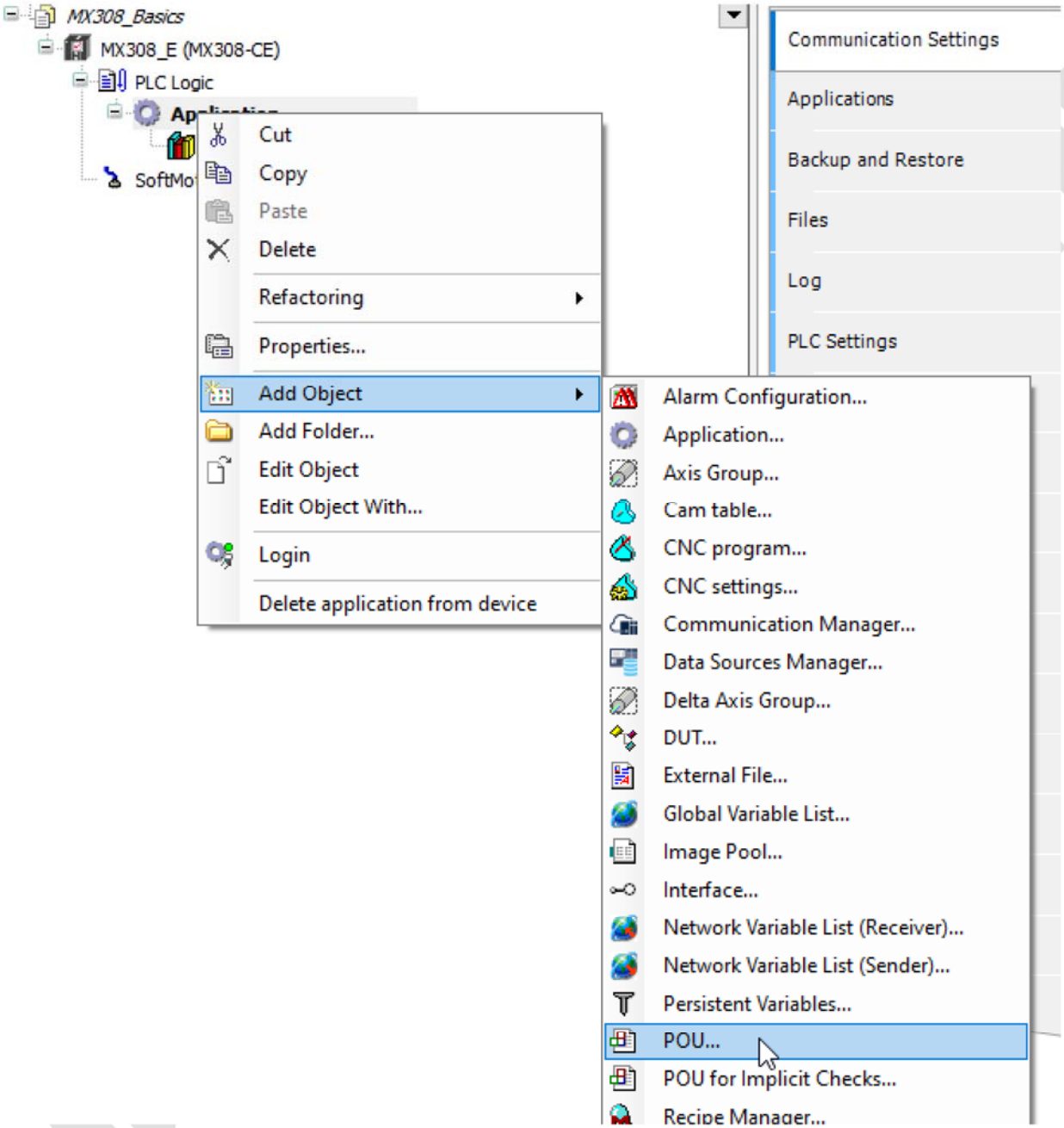


Для выхода из режима онлайн нажмите кнопку с красным крестиком:

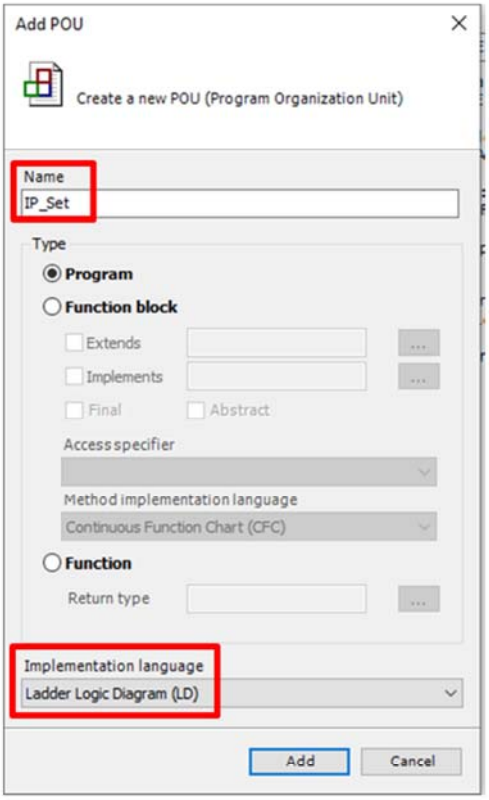


Изменение IP адреса контроллера из программы контроллера

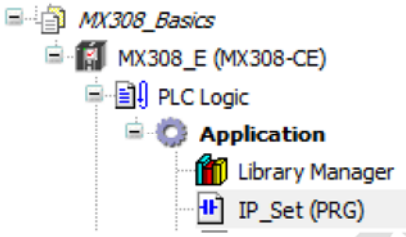
Создайте в проекте контроллера программную единицу (POU). Для этого щёлкните в древе проекта на пункте **Application** правой кнопкой мышки и в открывшемся меню выберите пункт **Add Object – POU:**



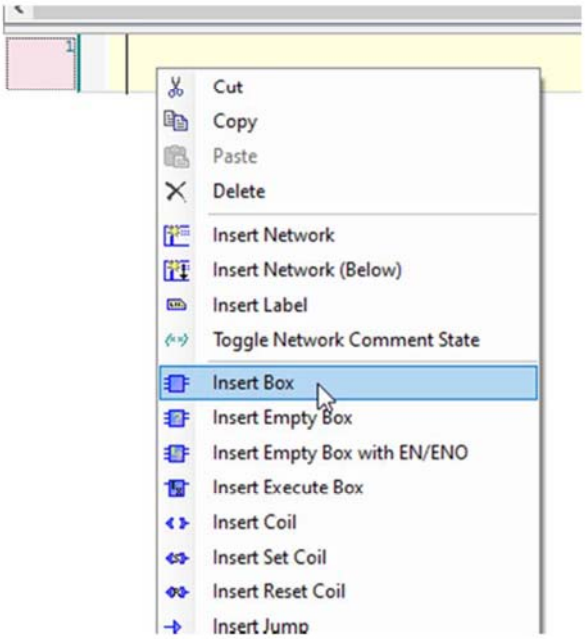
В открывшемся окне задайте название и язык программирования:



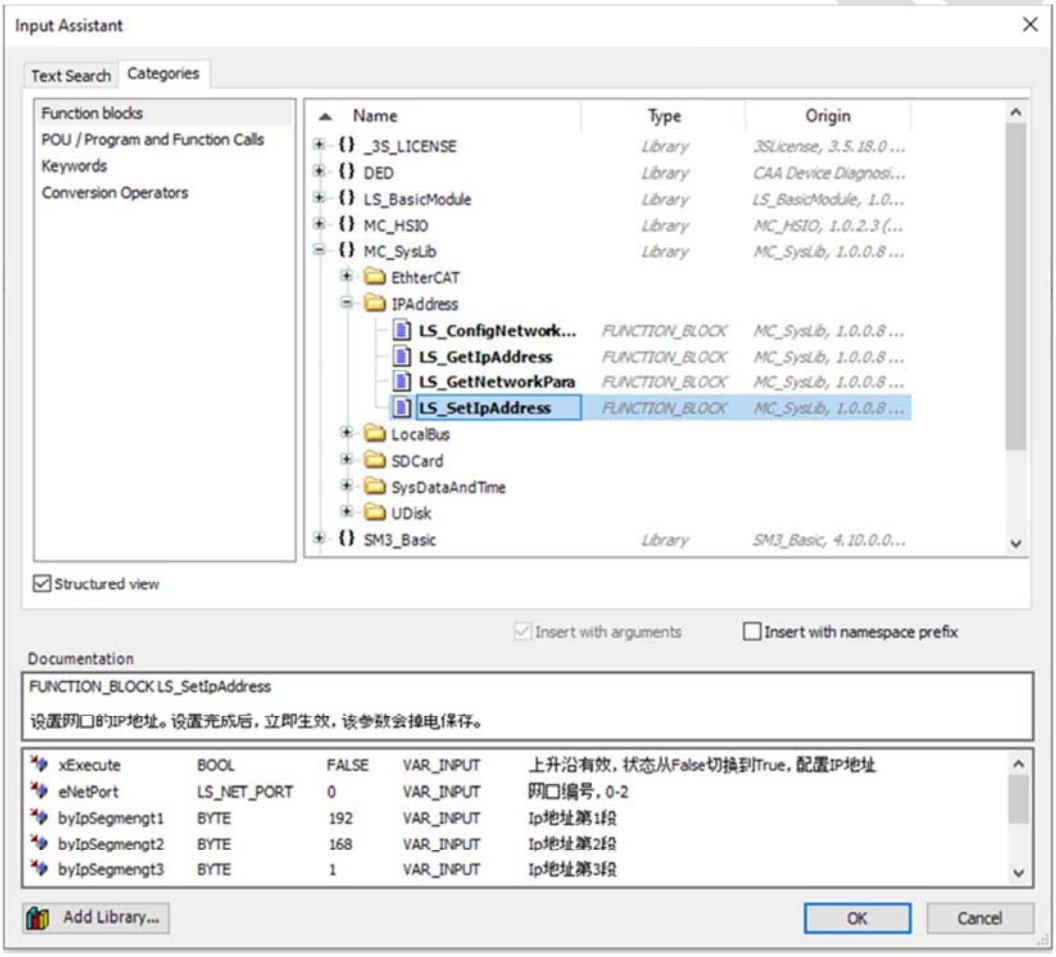
Нажмите кнопку **Add**. В древе проекта появится POU IP_Set



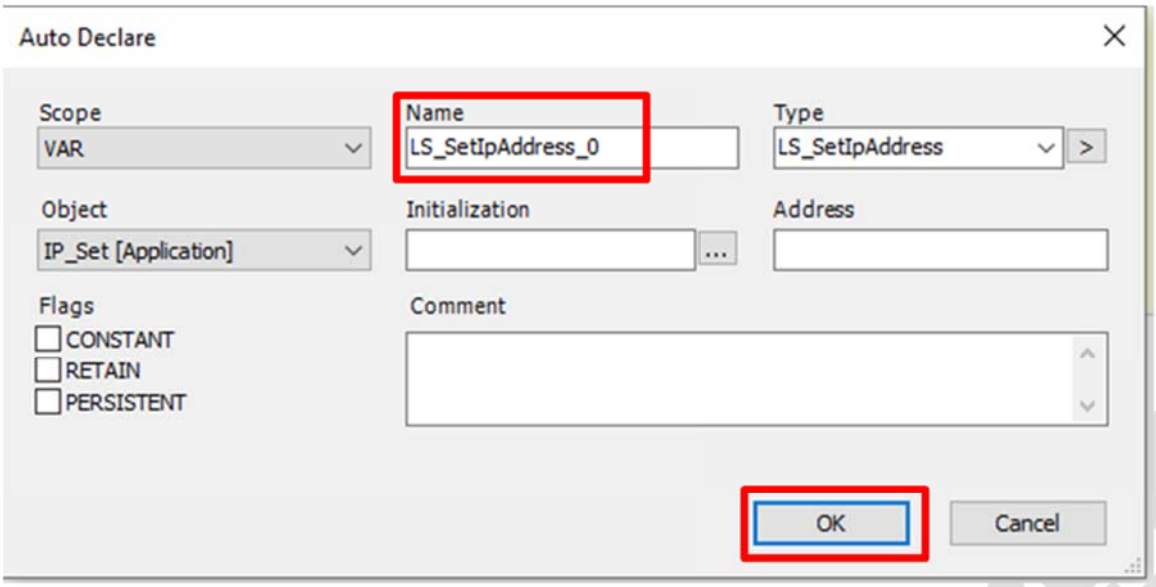
В окне созданного POU в поле ввода кода нажмите правой кнопкой мышки и выберите пункт **Insert Box**:



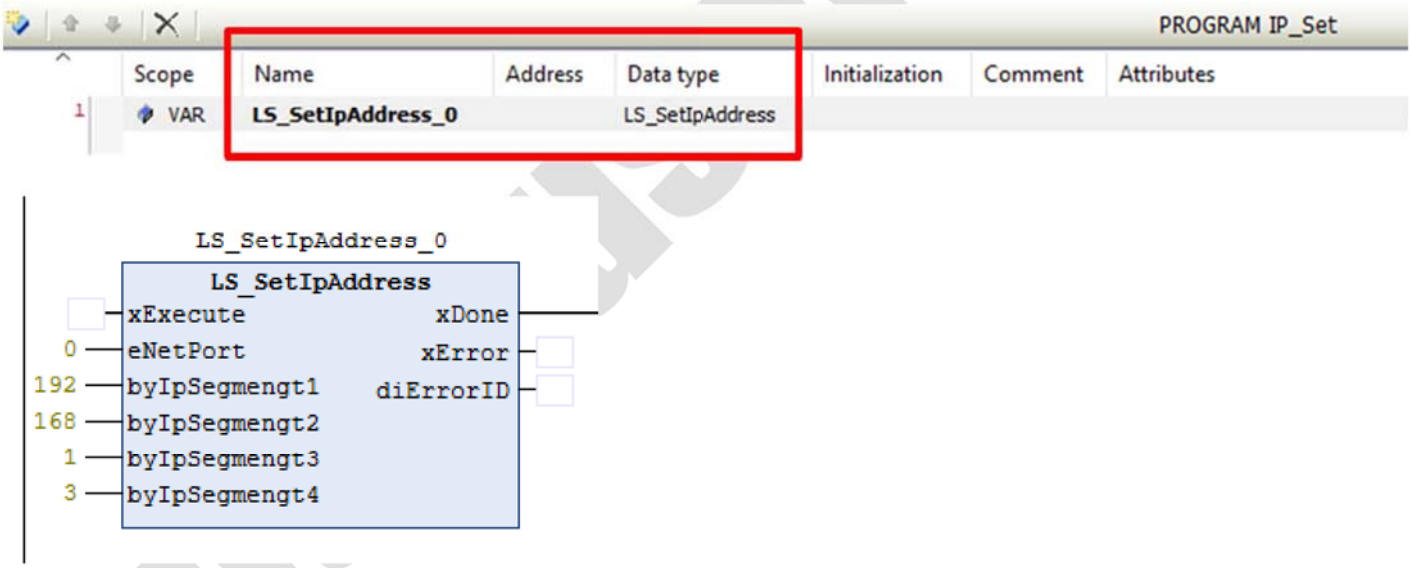
Откроется Мастер выбора команд. Нужно выбрать команду **LS_SetIpAddress** (библиотека **MC_SysLib**)



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

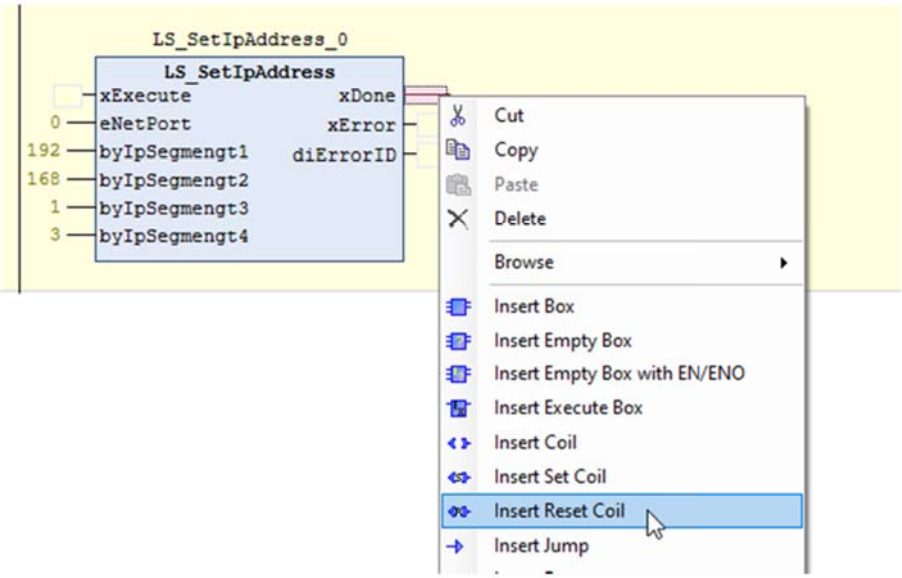


Появится переменная в таблице, а команда на строке кода:



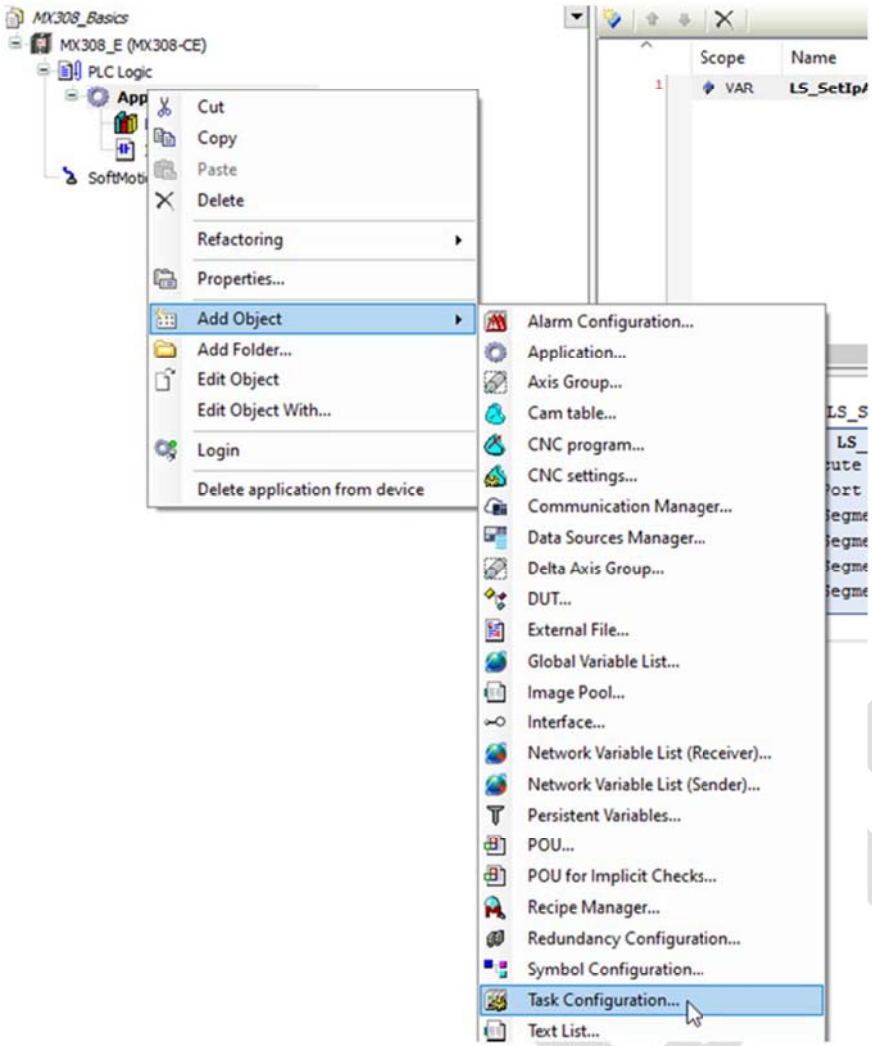
Введите нужный IP адрес (в примере вверху задан **192.168.1.3**)

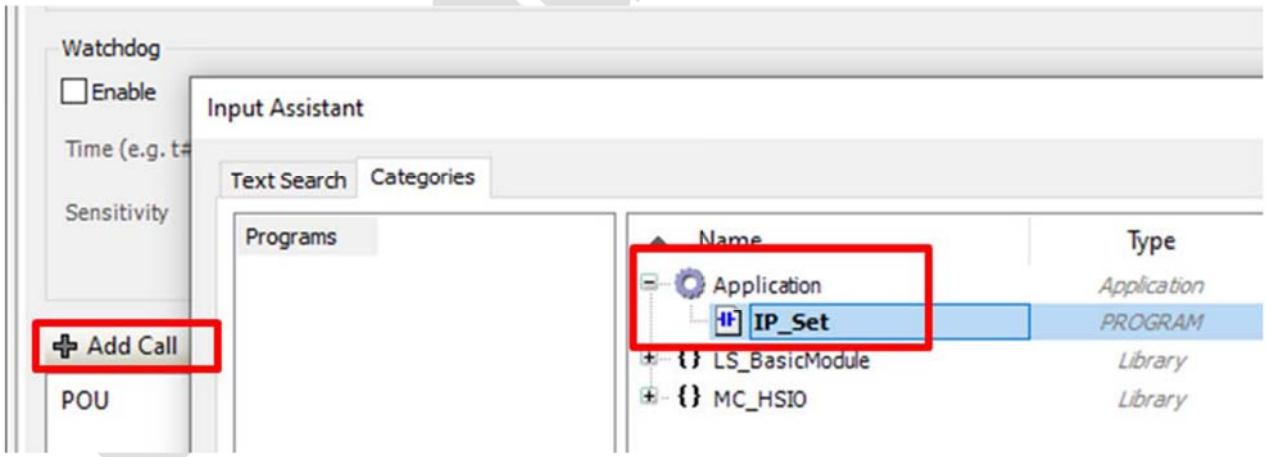
На ножку xDone установите катушка типа RST со следующей переменной **LS_SetIpAddress_0.xExecute**:



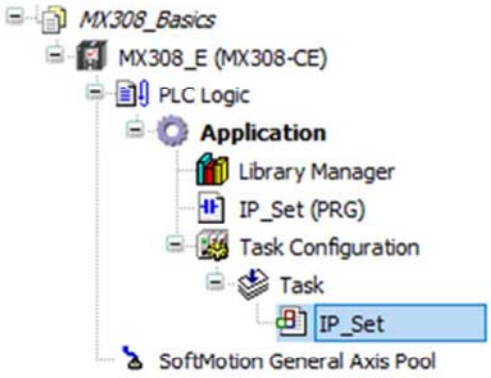
В итоге получится код следующего вида:

Далее необходимо добавить в проект Менеджер задач и добавить в него POU IP_Set:





В древе проекта появится соответствующий пункт:



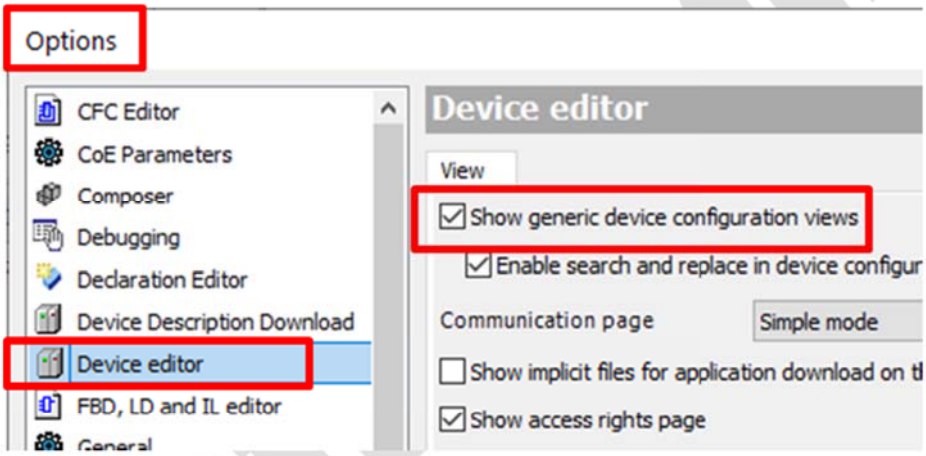
Внимание! Для активации нового IP адреса необходимо загрузить программу в контроллер и кратковременно перевести его в RUN.

См. Пример OD Samples – Nets - IpAddrChg_SD.project

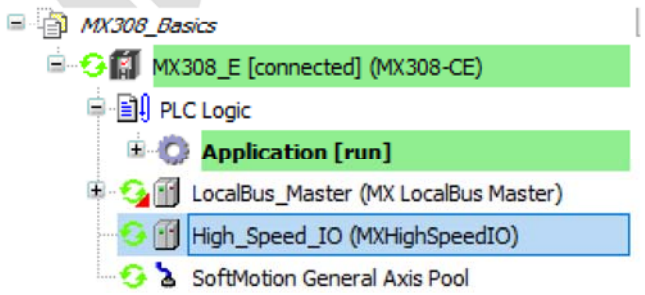
Изменение IP адреса контроллера из среды программирования

Для изменения IP адреса контроллера из среды программирования в проект необходимо добавить адаптер встроенных входов-выходов **MXHighSpeedIO** (см. соответствующие Главы настоящего Руководства).

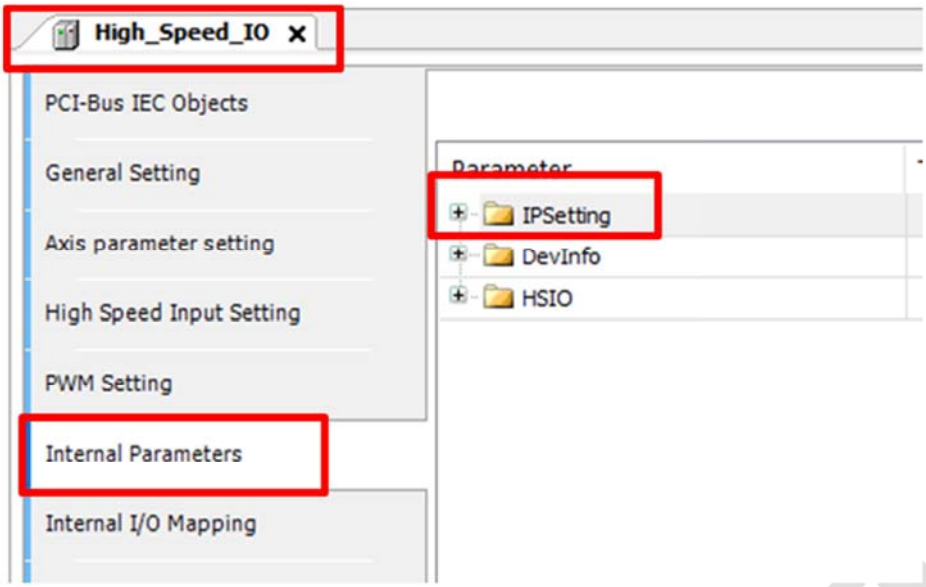
Также, а настройках среды программирования необходимо включить полное отображение настроек устройств. Для этого зайдите в меню: Tools – Options – Device editor и поставьте флажок на пункте **Show generic device configuration views**:



Далее необходимо соединиться с контроллером, загрузить проект и войти в онлайн с ним (см. соответствующие Главы настоящего Руководства). В нашем примере имя экземпляра **MXHighSpeedIO** переименовано в **High_Speed_IO**. Если указанные этапы будут выполнены правильно, то при подключении к контроллеру в онлайн режиме все объекты будут подсвечены зелёным цветом:



Щёлкните в древе проекте дважды левой кнопкой мышки на пункте **High_Speed_IO (MXHighSpeedIO)**, отроется вкладка с настройками адаптера. Выберите пункт **Internal Parameters**:



Разверните список **IPSetting**:

Появятся поля настройки адреса, маски и шлюза.

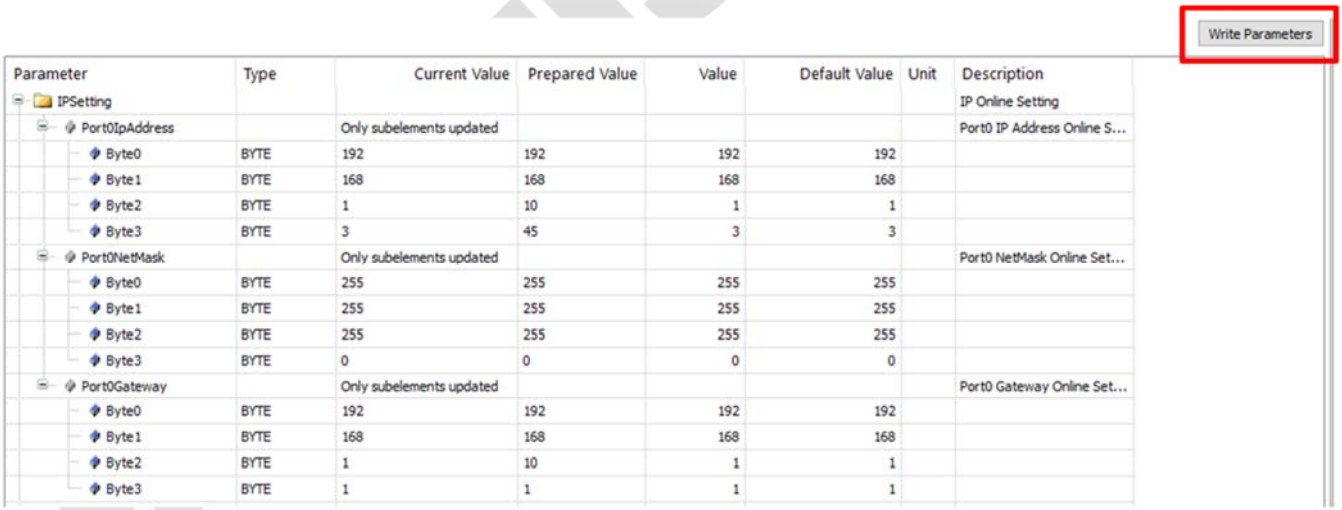
В колонке **Current Value** отображаются текущий адрес, маска и шлюз.

Parameter	Type	Current Value	Prepared Value	Value
IPSetting				
Port0IpAddress		Only subelements updated		
Byte0	BYTE	192		192
Byte1	BYTE	168		168
Byte2	BYTE	1		1
Byte3	BYTE	3		3
Port0NetMask		Only subelements updated		
Byte0	BYTE	255		255
Byte1	BYTE	255		255
Byte2	BYTE	255		255
Byte3	BYTE	0		0
Port0Gateway		Only subelements updated		
Byte0	BYTE	192		192
Byte1	BYTE	168		168
Byte2	BYTE	1		1
Byte3	BYTE	1		1

Для их изменения используйте колонку **Prepared Value**:

Parameter	Type	Current Value	Prepared Value
IPSetting			
Port0IpAddress		Only subelements updated	
Byte0	BYTE	192	192
Byte1	BYTE	168	168
Byte2	BYTE	1	10
Byte3	BYTE	3	45
Port0NetMask		Only subelements updated	
Byte0	BYTE	255	255
Byte1	BYTE	255	255
Byte2	BYTE	255	255
Byte3	BYTE	0	0
Port0Gateway		Only subelements updated	
Byte0	BYTE	192	192
Byte1	BYTE	168	168
Byte2	BYTE	1	10
Byte3	BYTE	1	1
DevInfo			
HSIO			

Далее необходимо нажать кнопку **Write Parameters** в правом верхнем углу вкладки:



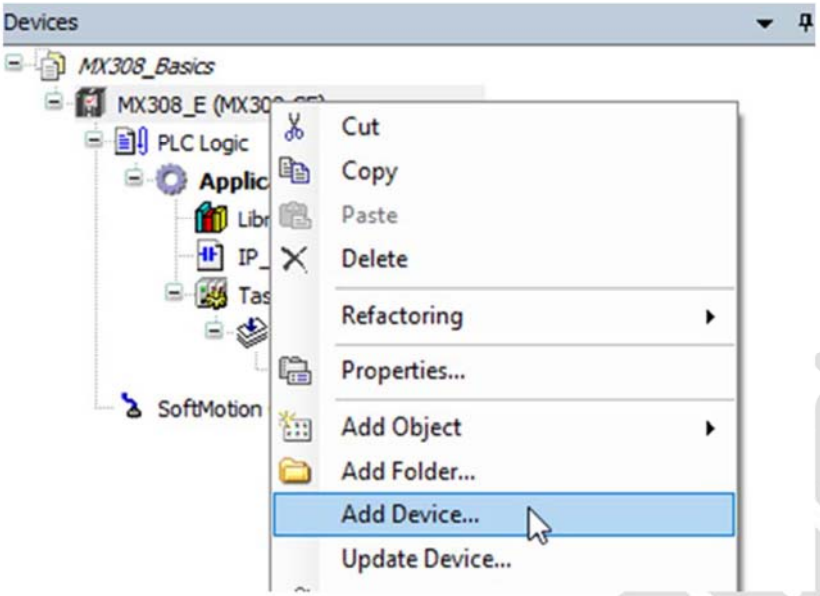
Parameter	Type	Current Value	Prepared Value	Value	Default Value	Unit	Description
IPSetting							IP Online Setting
Port0IpAddress		Only subelements updated					Port0 IP Address Online S...
Byte0	BYTE	192	192	192	192		
Byte1	BYTE	168	168	168	168		
Byte2	BYTE	1	10	1	1		
Byte3	BYTE	3	45	3	3		
Port0NetMask		Only subelements updated					Port0 NetMask Online Set...
Byte0	BYTE	255	255	255	255		
Byte1	BYTE	255	255	255	255		
Byte2	BYTE	255	255	255	255		
Byte3	BYTE	0	0	0	0		
Port0Gateway		Only subelements updated					Port0 Gateway Online Set...
Byte0	BYTE	192	192	192	192		
Byte1	BYTE	168	168	168	168		
Byte2	BYTE	1	10	1	1		
Byte3	BYTE	1	1	1	1		

Далее необходимо снять-подать питание на контроллер!

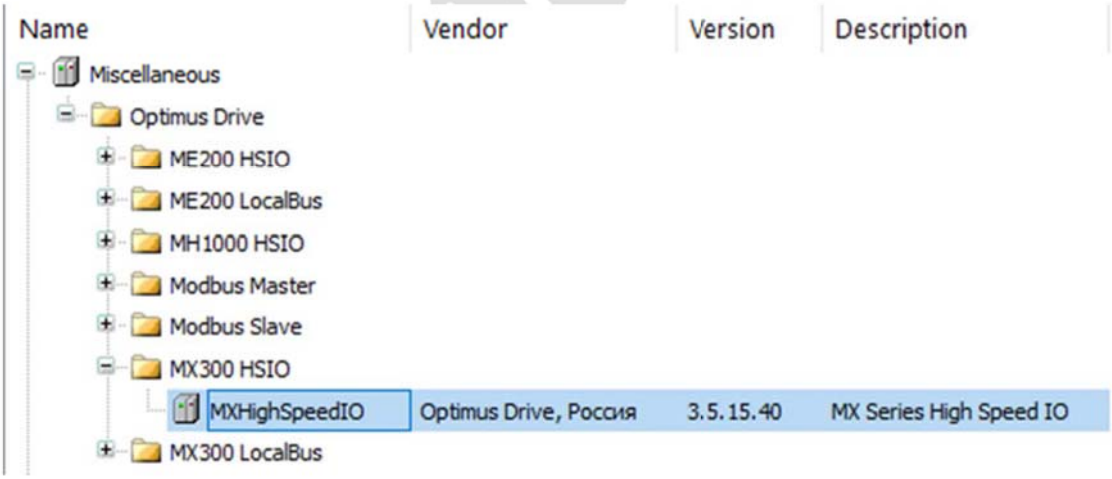
Новые адрес, маска и шлюз запишутся в контроллер. После смены IP адреса потребуется настройка сетевой карты компьютера и повторное соединение с контроллером по новому адресу.

Использование встроенных входов-выходов контроллера в обычном режиме

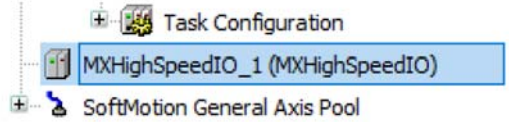
Для использования входов-выходов на ЦПУ в обычном режиме (не в импульсном) достаточно добавить в проект объект типа **MXHighSpeedIO** (идёт в составе пэкиджа, см. выше) для чего щёлкните правой кнопкой мышки на контроллере в древе проекта и в отрывшемся меню выберите пункт **Add Device**:



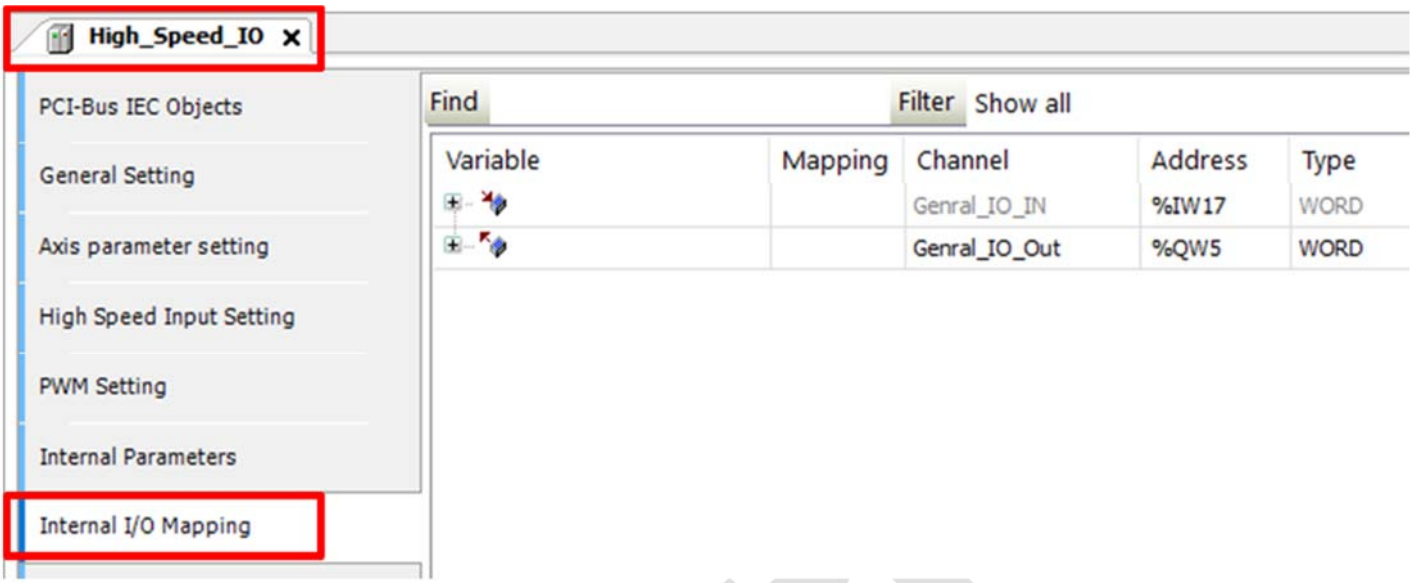
В появившемся окне выберите папку **Miscellaneous – Optimus Drive – MX300 HSIO** и пункт **MXHighSpeedIO** и нажмите кнопку **Add Device**:



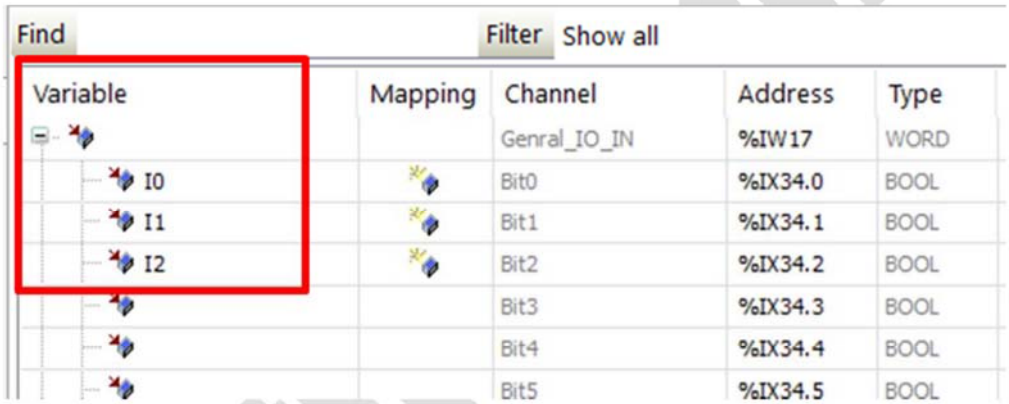
В древе проекта появится пункт **MXHighSpeedIO_1 (MXHighSpeedIO)**:



Щёлкните на пункте **MXHighSpeedIO_1** дважды левой кнопкой мышки, отроется вкладка параметров входов-выходов на ЦПУ. Выберите в ней пункт **Internal I/O Mapping**, в которой будут видны физические адреса входов и выходов. Здесь же в колонке **Variable** можно задать тэги для них. Для этого разверните список и щёлкните правой кнопкой мышки на самом верхнем входе. В открывшемся окне выберите пункт Create I/O variables и, например, Name by Prefix.



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



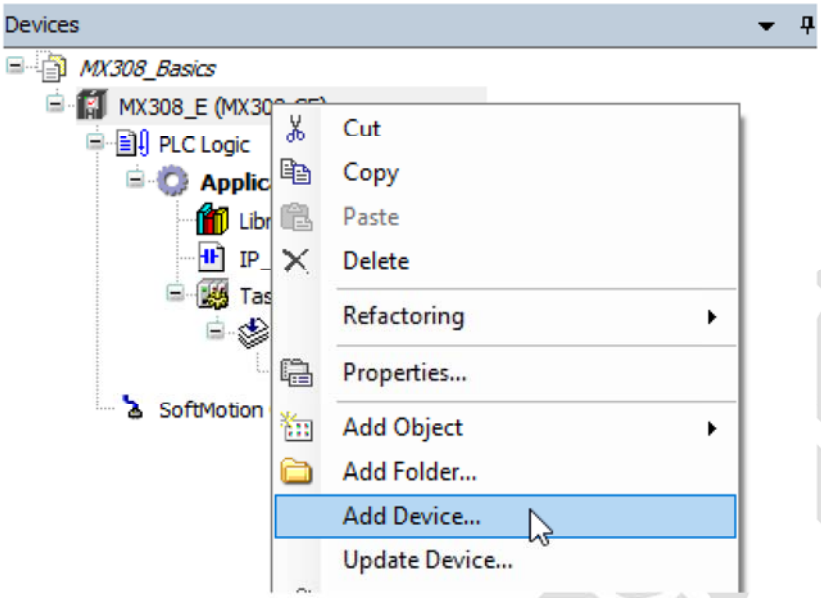
Теги для выходов заполняются аналогичным образом.
В программе состояние входов можно считывать как по тегам, так и по физическим адресам.

Добавление в проект модулей расширения

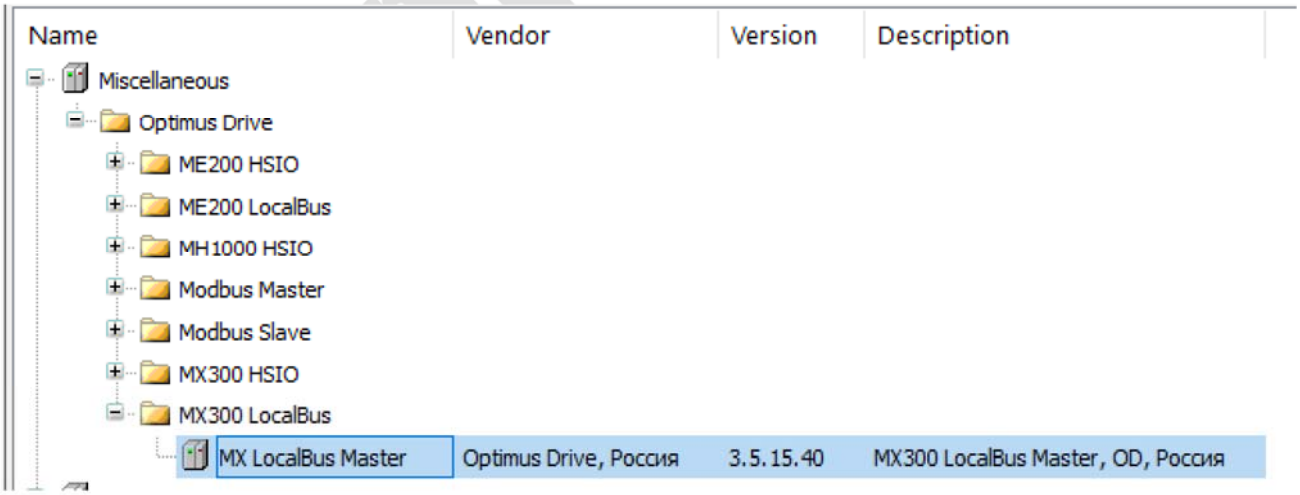
Контроллеры серии MX300 могут расширяться модулями дискретного и аналогового ввода-вывода. Всего к контроллеру по внутренней шине могут быть подключены до 32-х модулей расширения.

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

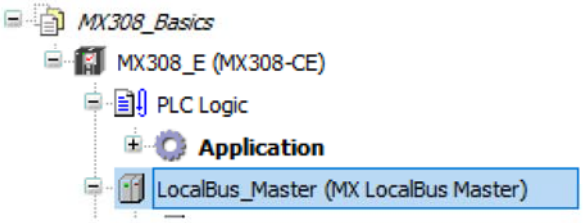
Для добавления модуля расширения в проект щёлкните правой кнопкой мышки на контроллере в древе проекта и в отрывшемся меню выберите пункт **Add Device**:



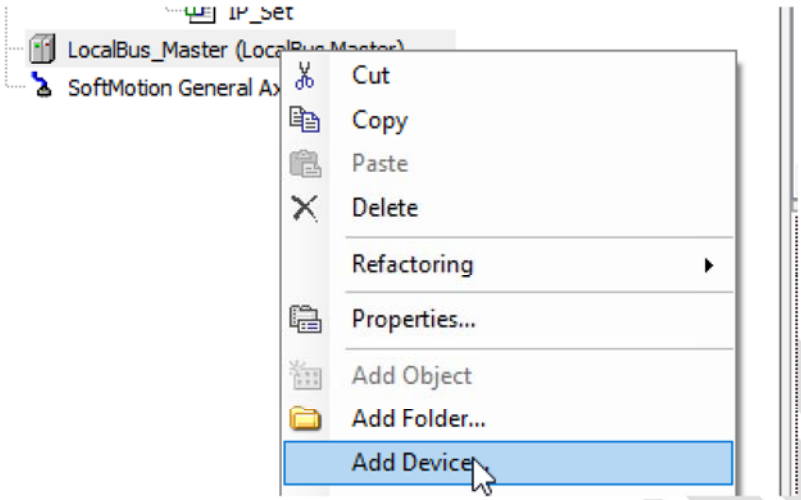
В появившемся окне выберите папку **Miscellaneous – Optimus Drive – MX LocalBus** и пункт **MX LocalBus Master** и нажмите кнопку **Add Device**:



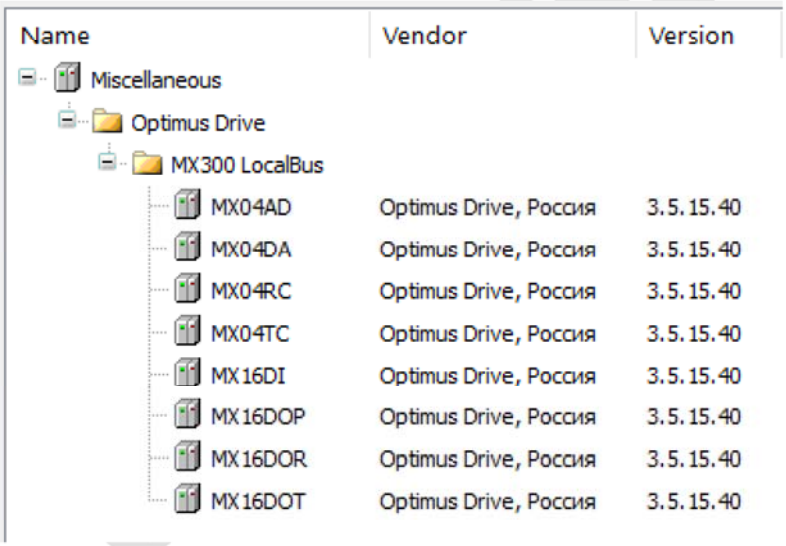
В древе проекта появится пункт **LocalBus_Master (MX LocalBus Master)**:



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

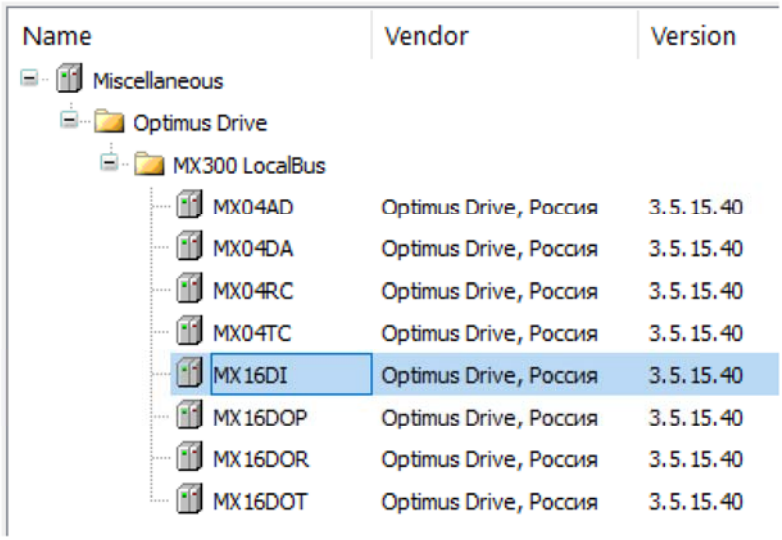


В открывшемся окне выберите нужный модуль:

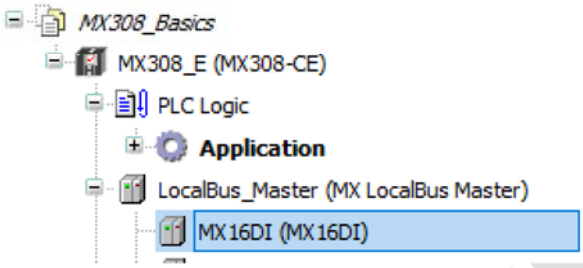


Работа с модулем дискретных входов MX16DI

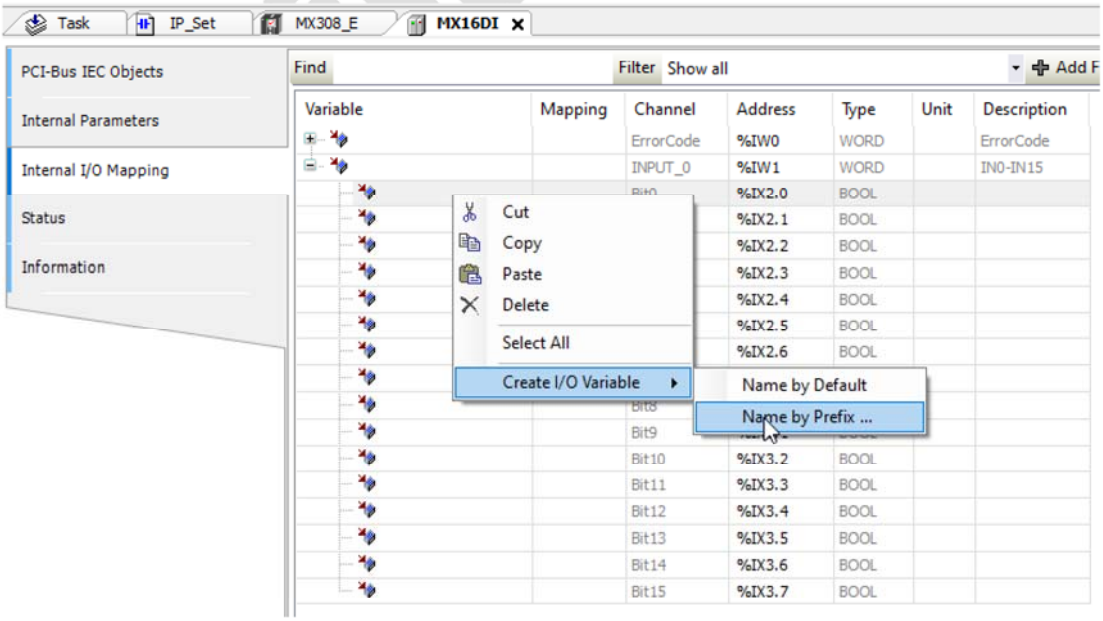
Добавьте в проект по вышеприведённой последовательности модуль MX16DI:



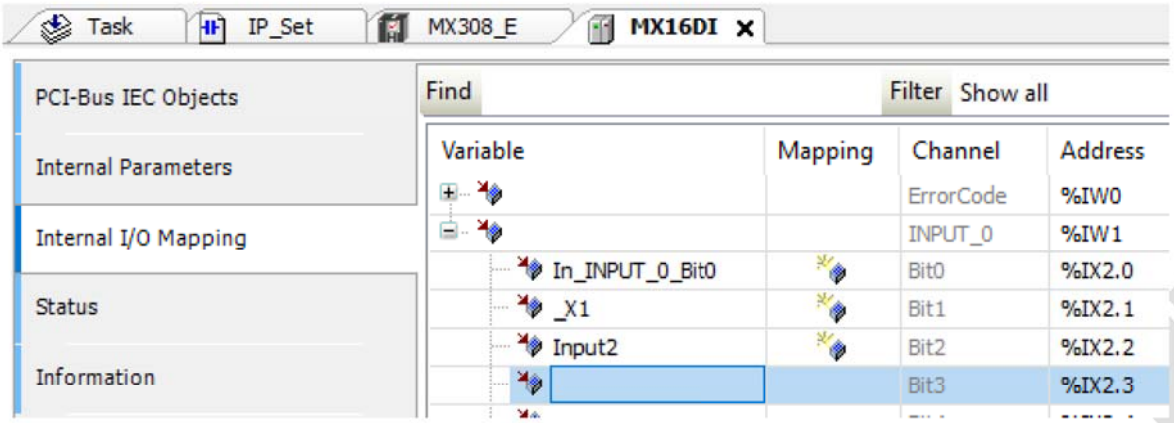
В проекте появится пункт:



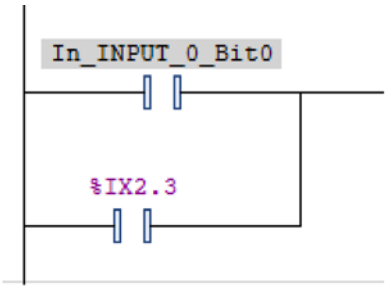
Щёлкните на пункте MX16DI дважды левой кнопкой мышки, отроется вкладка параметров модуля. Выберите в ней пункт Internal I/O Mapping, в которой будут видны физические адреса входов и можно задать тэги для входов. Для этого разверните список и щёлкните правой кнопкой мышки на самом верхнем входе. В открывшемся окне выберите пункт Create I/O variables и, например, Name by Prefix.



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

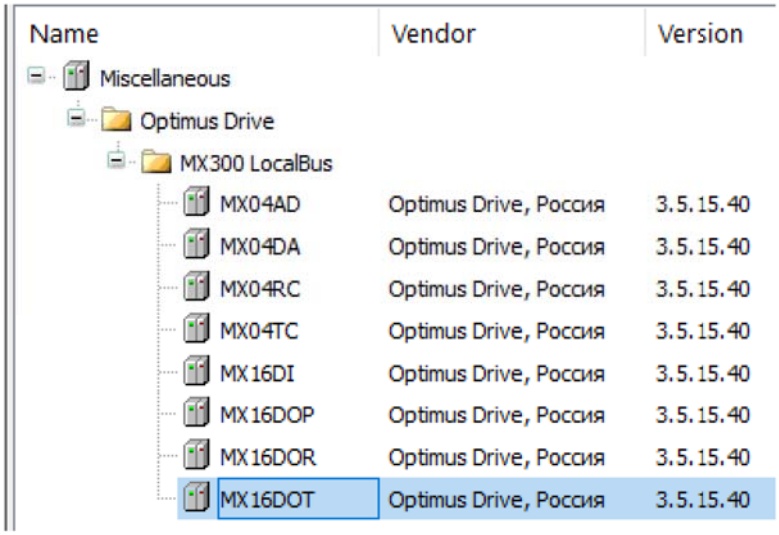


В программе состояние входов можно считывать как по тегам, так и по физическим адресам.

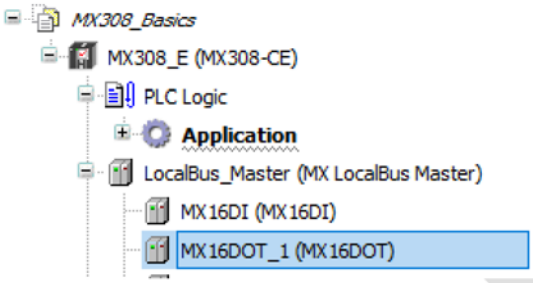


Работа с модулями дискретных выходов MX16DOT/MX16DOP/MX16DOR

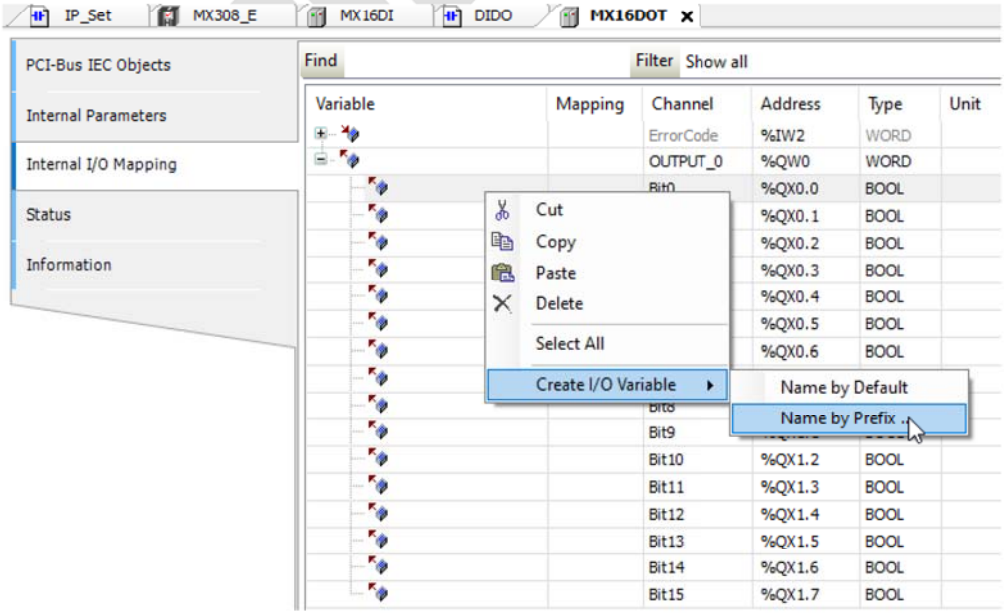
Добавьте в проект по вышеприведённой последовательности модуль MX16DOT:
(модули MX16DOP и MX16DOR настраиваются аналогично)



В проекте появится пункт:



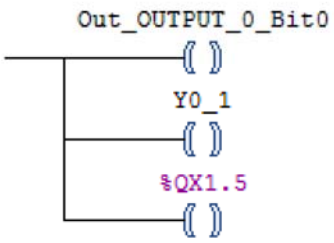
Щёлкните на пункте MX16DOT дважды левой кнопкой мышки, отроется вкладка параметров модуля. Выберите в ней пункт **Internal I/O Mapping**, в которой будут видны физические адреса выходов и можно задать тэги для выходов. Для этого разверните список и щёлкните правой кнопкой мышки на самом верхнем выходе. В открывшемся окне выберите пункт Create I/O variables и, например, Name by Prefix.



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

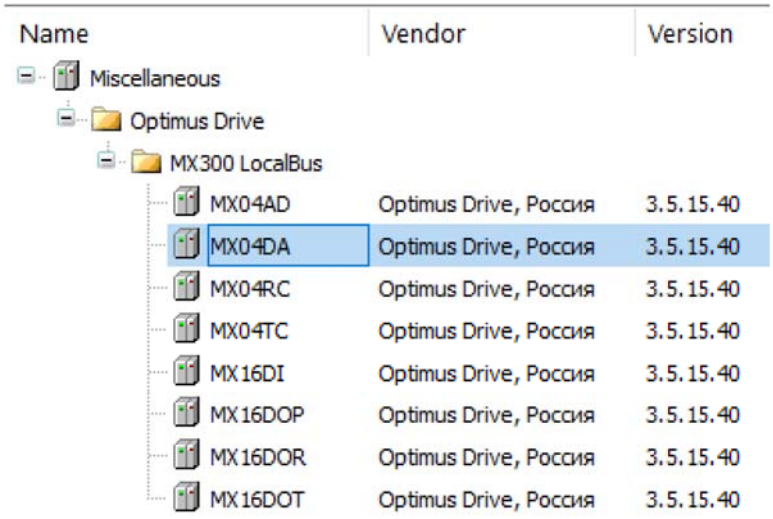
Variable	Mapping	Channel	Address	Type	Uni
		ErrorCode	%IW2	WORD	
		OUTPUT_0	%QW0	WORD	
Out_OUTPUT_0_Bit0		Bit0	%QX0.0	BOOL	
Y0_1		Bit1	%QX0.1	BOOL	
Output2		Bit2	%QX0.2	BOOL	
		Bit3	%QX0.3	BOOL	
		Bit4	%QX0.4	BOOL	

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

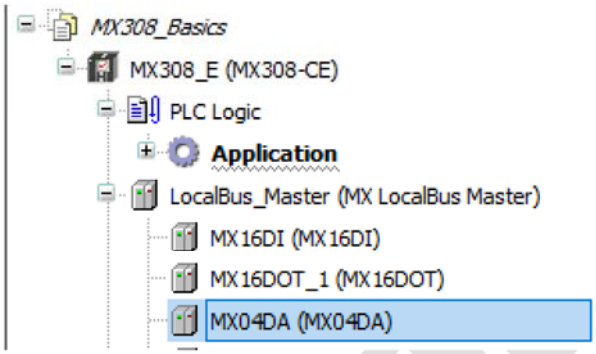


Работа с модулем аналоговых выходов MX04DA

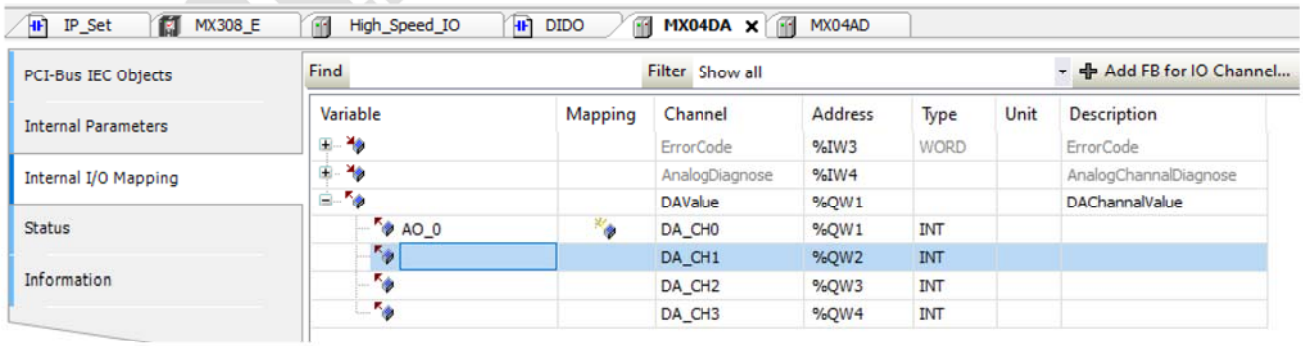
Добавьте в проект по вышеприведённой последовательности модуль MX04DA:



В проекте появится пункт:

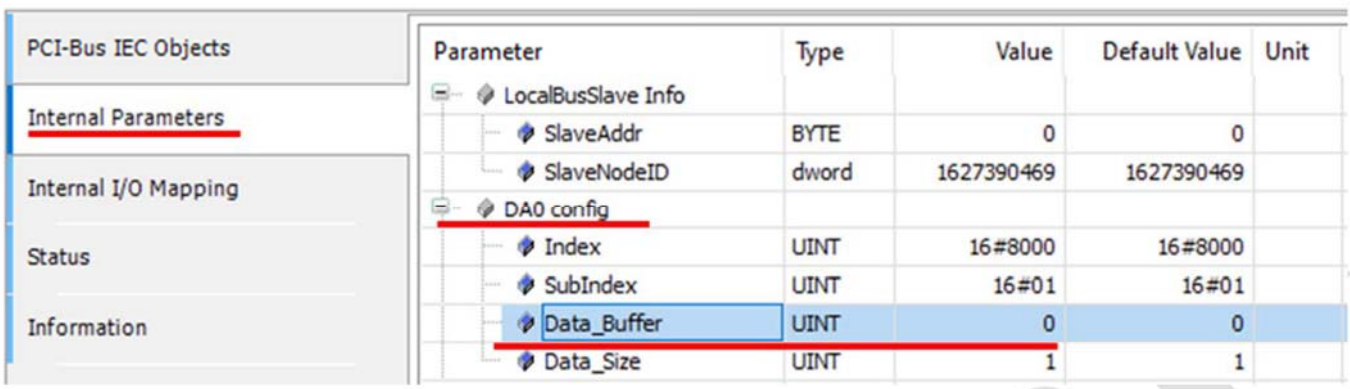


Щёлкните на пункте MX04DA дважды левой кнопкой мышки, отроется вкладка параметров модуля. Выберите в ней пункт **Internal I/O Mapping**, в которой будут видны физические адреса выходов и можно задать тэги для выходов. Для этого разверните список и введите нужное название тега.



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**.

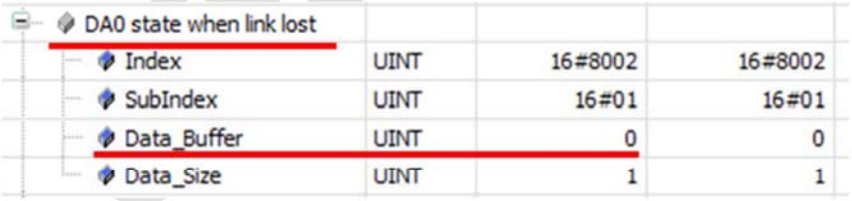


В пункт **Data Buffer** вносится номер режима работы в соответствии с нижеприведённой таблицей:

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.

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



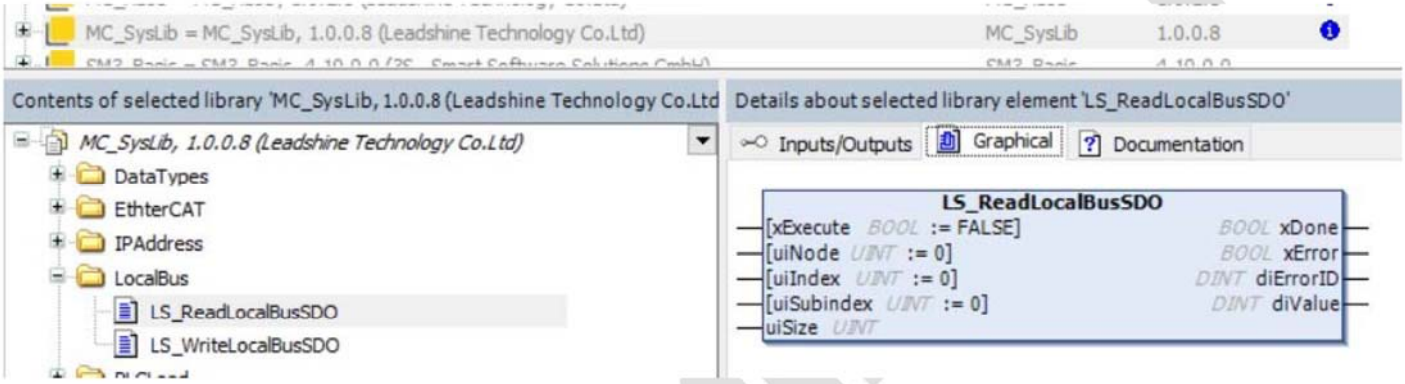
Значение в Data_Buffer	Реакция канала
0	Сохранять текущее значение
1	Сброс на 0
2	Выдача заранее предустановленного значения

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

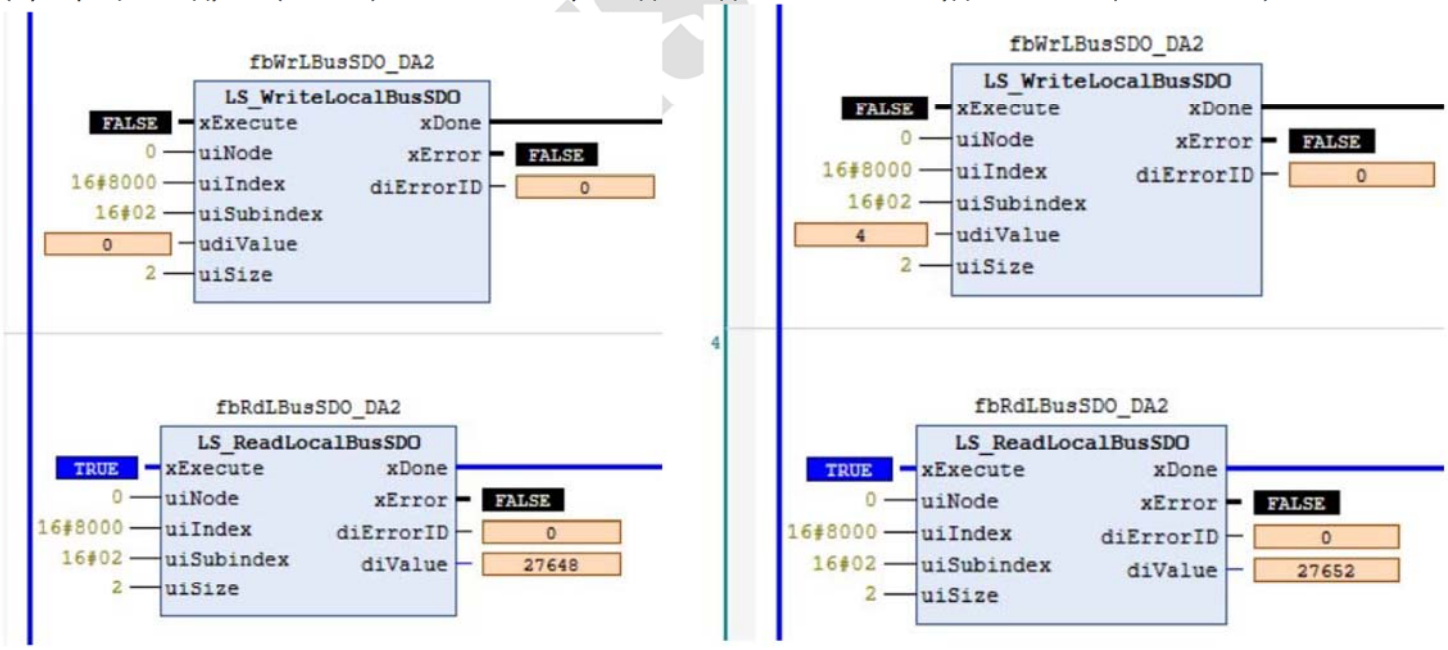
DA0 value when link lost			
Index	UINT	16#8003	16#8003
SubIndex	UINT	16#01	16#01
Data_Buffer	UINT	0	0
Data_Size	UINT	2	2

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 и **LS_WriteLocalBusSDO**



Например, запись режима 0 и чтение обратно для проверки:
(нумерация модулей (uiNode) начинается с нуля и далее до 31 по степени удаления от ЦПУ на шине)



Примечание:
В старых версиях встроенного ПО (firmware) контроллера при чтении могут появляться не те цифры, что вводились при записи параметров:
Data_Buffer = 0 - на записи, соответствует 27648 - на чтении
Data_Buffer = 4 соответствует 27652

В новых версиях встроенного ПО данный недостаток устранён.

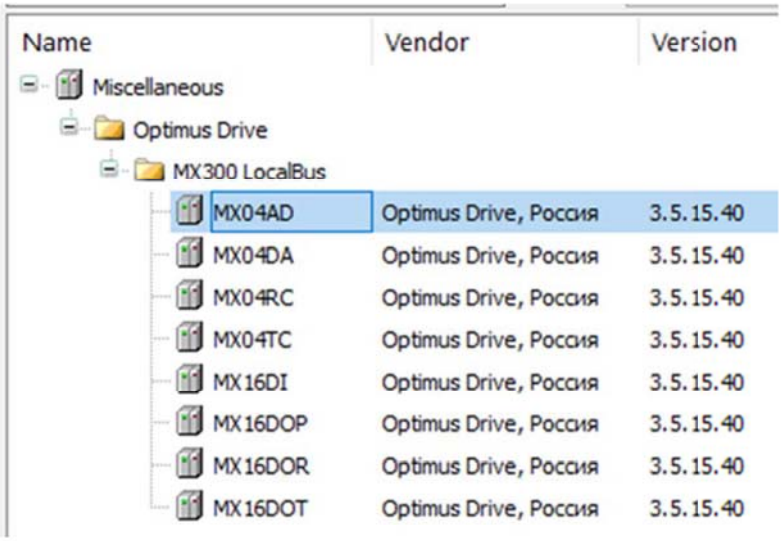
Внимание!

При работе с модулями ввода-вывода через локальную шину контроллера данные в колонках Value и Current Value в режиме онлайн не обновляются. Также, при изменении данных непосредственно в программе через SDO запросы не приводит к изменению данных в форме в колонках Value и Current Value. Актуальные данные необходимо брать из переменных в программе после выполнения SDO запросов. Колонка Value отображает значение, сохранившееся после последней загрузки проекта в контроллер.

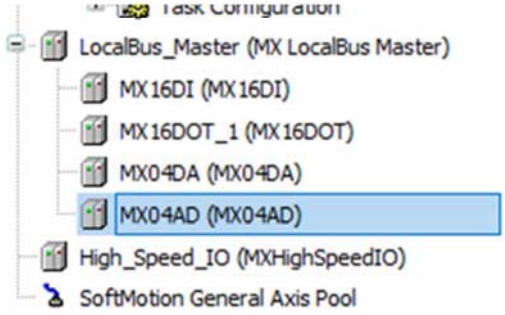
Parameter	Type	Current Value	Value	Default Value	Unit	D
LocalBusSlave Info		Only subelements up...				
DA0 config		Only subelements up...				
Index	UINT	32768	16#8000	16#8000		
SubIndex	UINT	32768	16#01	16#01		
Data_Buffer	UINT	32768	0	0		
Data_Size	UINT	32768	1	1		
DA1 config		Only subelements up...				
DA2 config		Only subelements up...				
DA3 config		Only subelements up...				
DA0 en		Only subelements up...				
DA1 en		Only subelements up...				

Работа с модулем аналоговых входов MX04AD

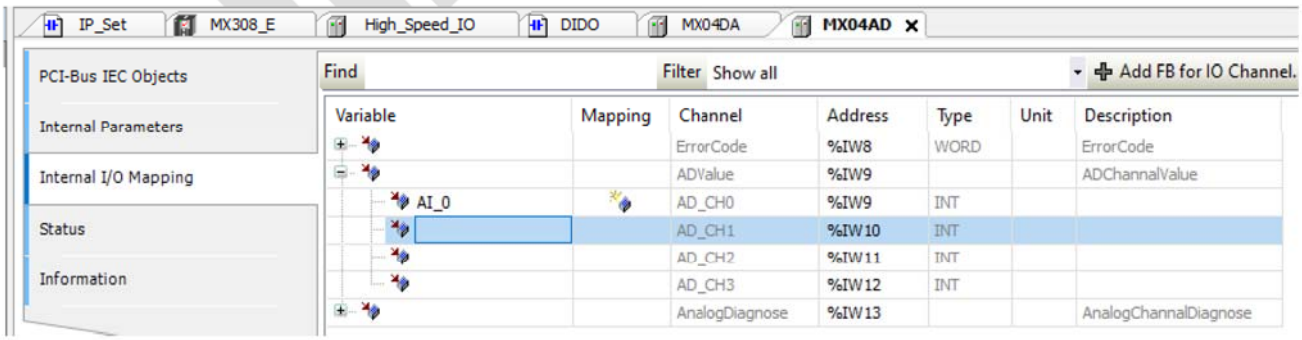
Добавьте в проект по вышеприведённой последовательности модуль MX04AD:



В проекте появится пункт:



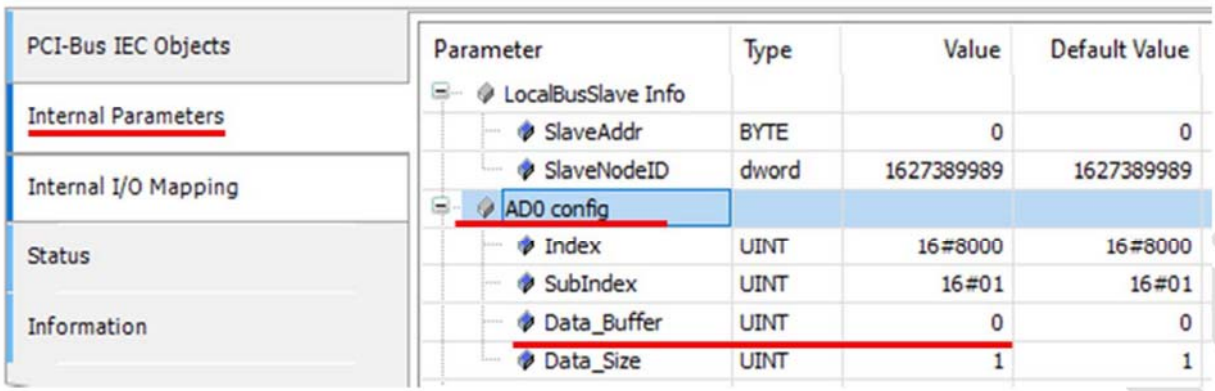
Щёлкните на пункте MX04AD дважды левой кнопкой мышки, отроется вкладка параметров модуля. Выберите в ней пункт **Internal I/O Mapping**, в которой будут видны физические адреса входов и можно задать тэги для входов. Для этого разверните список и введите нужное название тега.



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**.

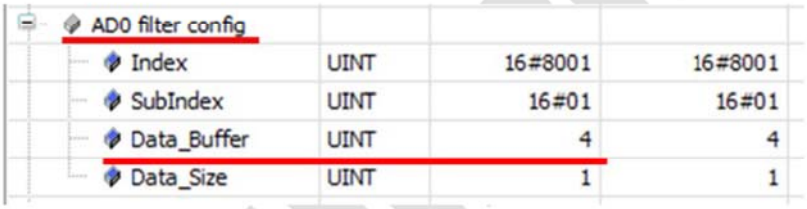
Выберите раздел типа **AD0 Config**. Для каждого канала AD0 – AD3 предусмотрен свой раздел. В данном разделе выберите пункт **Data Buffer**.



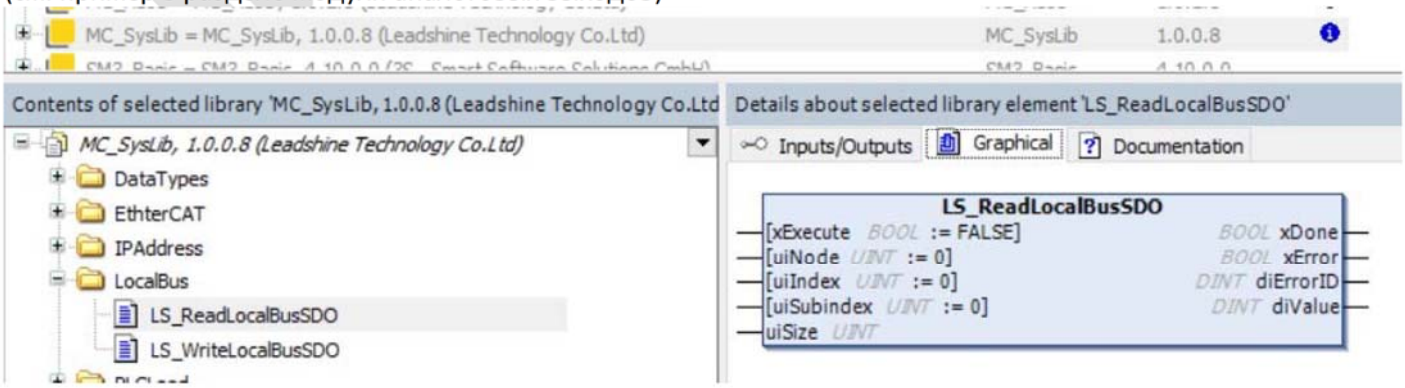
В пункт **Data Buffer** вносится номер режима работы в соответствии с нижеприведённой таблицей:

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

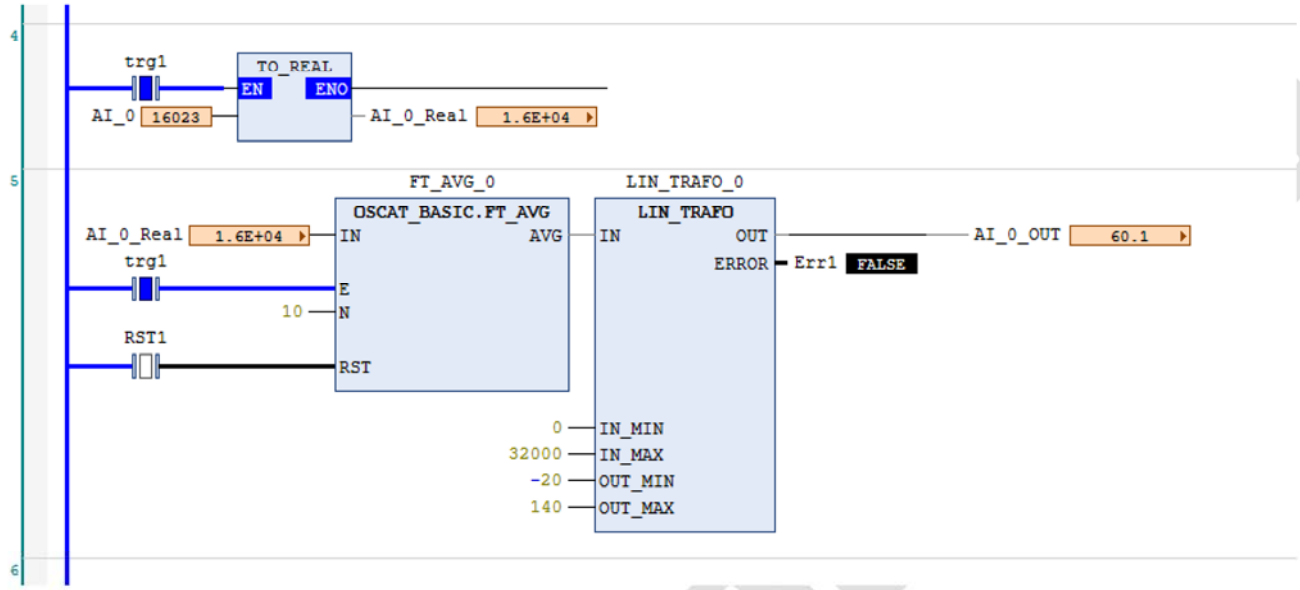
Далее, в разделе типа **AD0 filter config** в пункте **Data Buffer** нужно выставить цикл опроса входа в мс. По умолчанию стоит 4 мс.



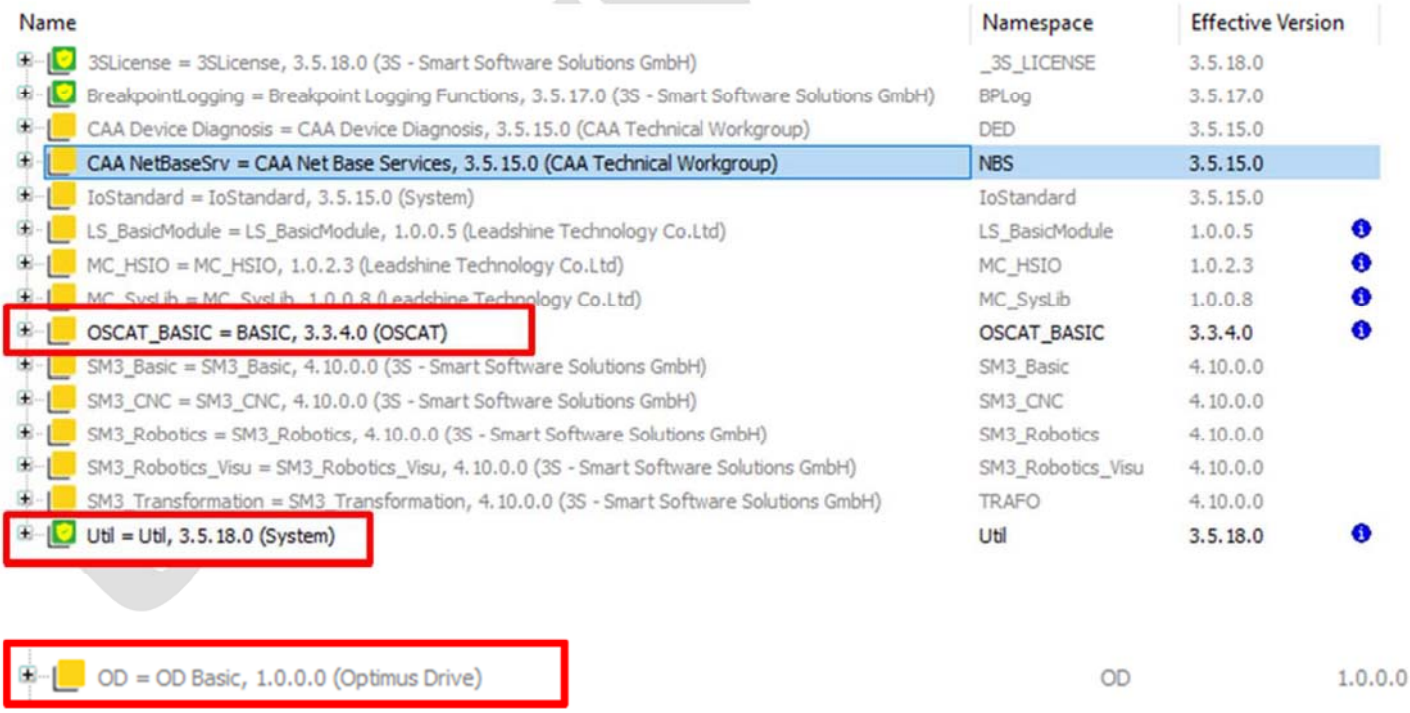
Модуль можно конфигурировать и из программы. Для этого используются команды типа:
LS_ReadLocalBusSDO
LS_WriteLocalBusSDO
(см. пример в разделе модуля аналоговых выходов)



Для осреднения входных значений по принципу «бегущее среднее» можно использовать ФБ **FT_AVG** из библиотеки **OSCAT_BASIC**. Также, команду **FT_AVG** можно взять из библиотеки **Optimus Drive OD_Basic**, которая идёт в составе основного пэкиджа и автоматически устанавливается совместно с описанием устройства. В данном случае библиотеку **OSCAT_BASIC** можно не устанавливать.
Для масштабирования шкалы АЦП а шкалу требуемой физической величины можно использовать ФБ **LIN_TRAFO** из библиотеки **Util**. Данная библиотека также подключается к проекту автоматически при установке основного пэкиджа с описанием устройства (контроллера).



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



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

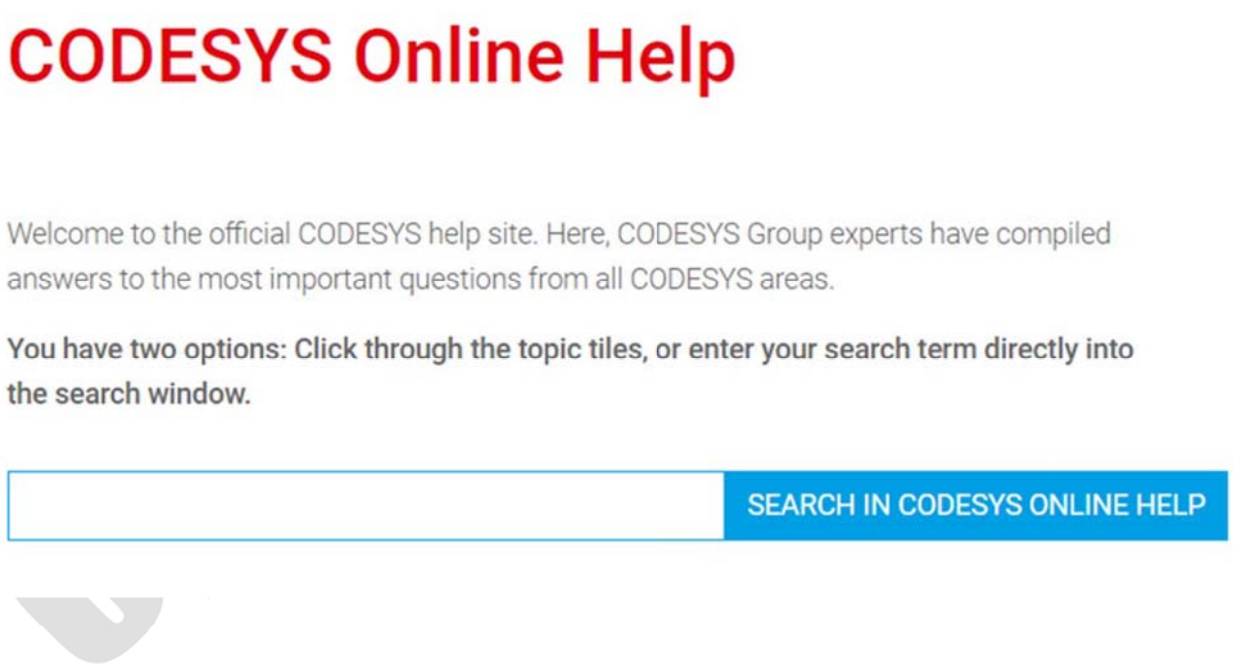
Среда программирования CODESYS является развитым, но достаточно сложным продуктом. В состав данного программного входит большое количество различных библиотек, каждая из которых содержит определённый набор команд. Существует несколько сообществ, разрабатывающих библиотеки независимо друг от друга. Кроме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

Контроллеры серии MX300 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:

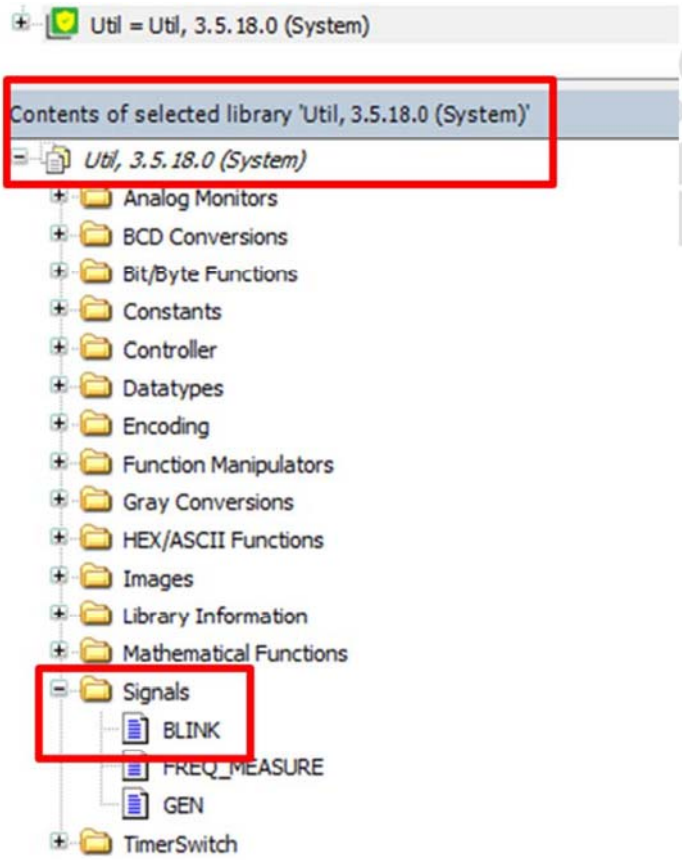
https://www.helpme-codesys.com/

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 MX300 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	SysFileOpen	FB	Открыть файл
	SysFileClose	FB	Закрыть файл
	SysFileWrite	FB	Записать файл
	SysFileRead	FB	Прочитать файл
	SysFileDelete	FB	Удалить файл
	SysFileCopy	FB	Копировать файл
	SysFileRename	FB	Переименование файла
	SysFileSetPos	FB	Установить местоположение чтения и записи файла
	SysFileGetPos	FB	Получить местоположение чтения и записи файла
	SysFileGetSize	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_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
	ETC_CO_SdoReadDWord	FB	Чтение данных EtherCAT через SDO, результат в формате DWORD

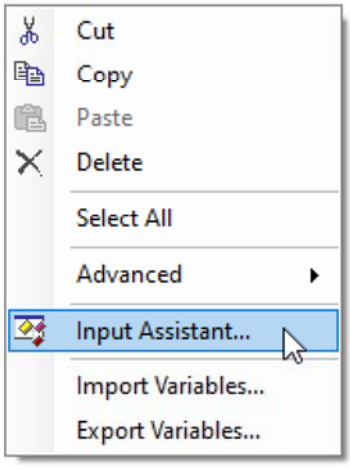
Инструкции EtherCAT	ETC_CO_SdoRead4	FB	Чтение данных EtherCAT через SDO, результат в формате ARRAY [1..4] OF BYTE
	ETC_CO_SdoRead	FB	Чтение данных EtherCAT через SDO, результат в формате CAA.SIZE
	ETC_CO_SdoWrite_DWord	FB	Запись данных EtherCAT через SDO, данные в формате DWORD
	ETC_CO_SdoWrite4	FB	Запись данных EtherCAT через SDO, данные в формате ARRAY [1..4] OF BYTE
	ETC_CO_SdoWrite	FB	Запись данных EtherCAT через SDO, данные в формате CAA.SIZE
	IoDrvEthercat_Diag	FB	Расширение для IoDrvEtherCAT, предоставляющее диагностические функции
	ETCSlave	FB	Создание экземпляра типа ETCSlave для каждого ведомого устройства EtherCAT в древе проекта
Инструкции Ethernet/IP	IoDrvEtherNetIP_Diag	FB	Расширение для IoDrvEtherNetIP
	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_Home_P	FB	Импульсная ось возвращается в исходное положение
	LS_MotionControl_P	FB	привязка оси импульса
	LS_ReadAxisPara_P	FB	Получите эквивалентное значение импульса оси импульса

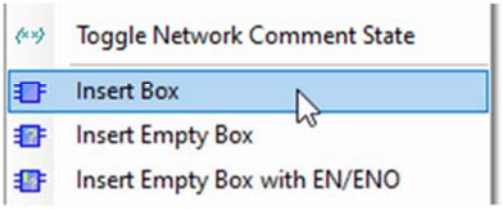
	LS_ResetAxis_P	FB	Сброс оси импульса
Высокоскоростной счетчик	LS_Counter	FB	Инструкция высокоскоростного счетчика
	LS_PresetValue	FB	Команда предустановленного значения счетчика

Категория инструкций	Название инструкции	FB/FC	Описание
Операции с SD картой	LS_CopyFromSDCard	FB	Копирование файлов с SD карты на локальный диск
	LS_CopyToSDCard	FB	Копирование локальных файлов на SD карту
	LS_GetSDCardInformation	FB	Получение информации о файле на SD карте
Системные команды	LS_GetIpAddress	FB	Прочитать IP адрес контроллера
	LS_SetIpAddress	FB	Установить IP адрес контроллера
	GetPLCLoad	FC	Прочитать степень загрузки контроллера
	GetPLCVersion	FC	Прочитать версию прошивки контроллера
	ColdResetApp	FC	«Холодная» перезагрузка контроллера
	WarmResetApp	FC	«Тёплая» перезагрузка контроллера
	LS_ReadDintDT	FB	Получить системное время в формате DINT
	LS_ReadStringDT	FB	Получить системное время в формате String
	LS_SetDintDT	FB	Установить системное время в формате DINT
	LS_SetStringDT	FB	Установить системное время в формате String

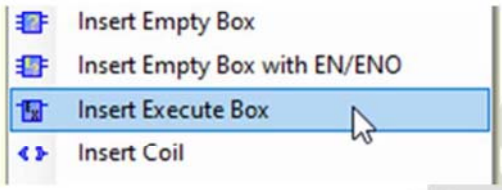
Для добавления в проект команды типа FB или FC на языке ST необходимо просто вызвать нажатием правой кнопки мышки и выбрать пункт **Input Assistant**:

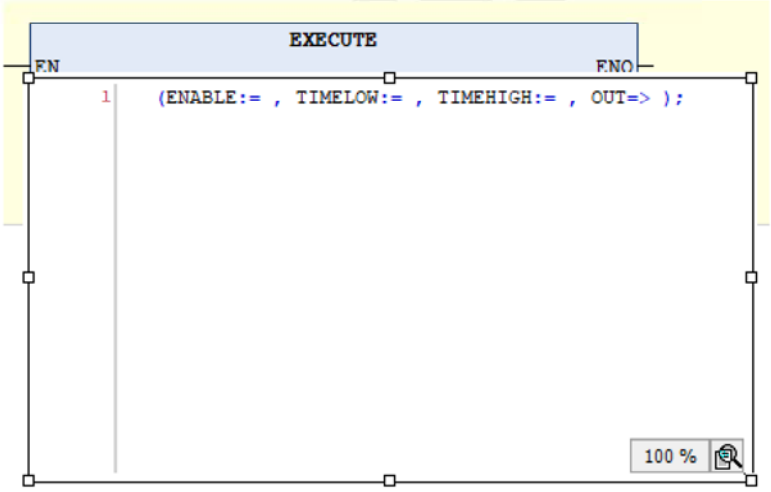


На языке LD команды типа FB вводятся через меню



а команды типа FC вводятся через

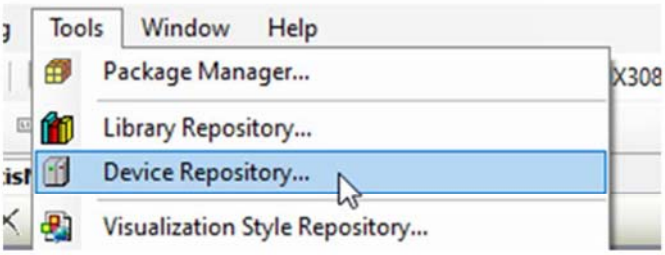


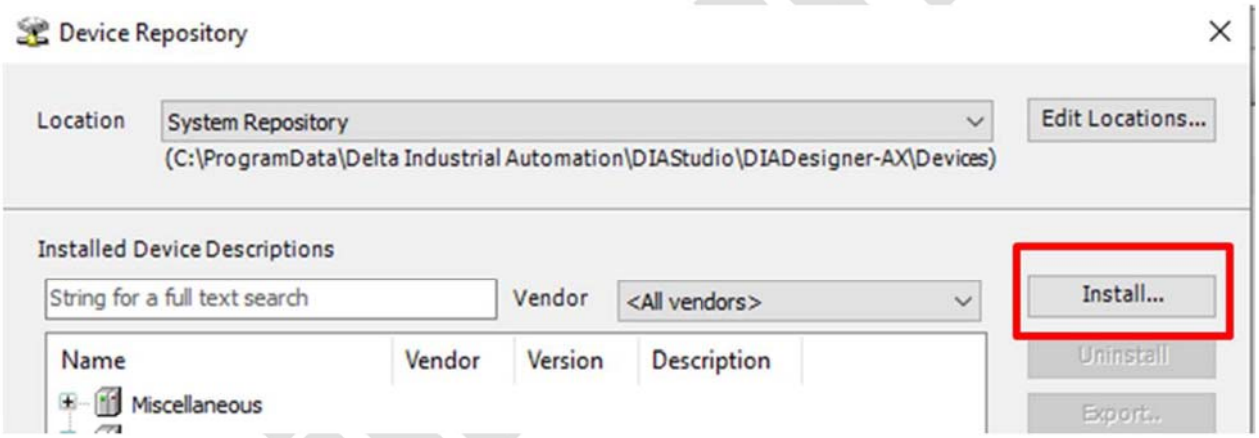


Добавление в проект сервопривода

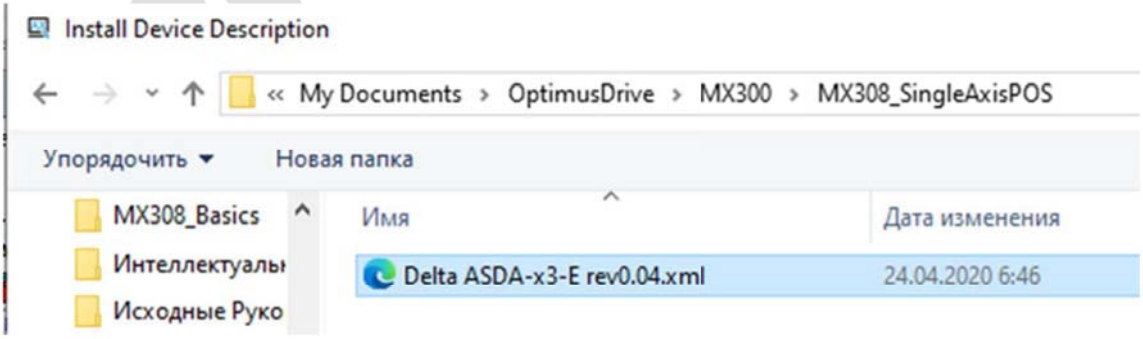
Контроллеры серии MX300 позволяют добавить в проект в качестве оси движения любой сервопривод с интерфейсом EtherCAT CoE (CANopen over EtherCAT) и поддерживающим стандарт CiA402, для которого производитель предоставляет корректный XML файл с описанием устройства. Сервопривод добавляется в проект как полноценная ось (Motion Axis), поддерживающая интерполированное движение. Вариант «лёгкой оси» для перемещений типа точка-точка не поддерживается.

Для добавления в проект сервопривода сначала необходимо установить файл с описанием устройства в депозитарий (**Tools – Device Repository**):

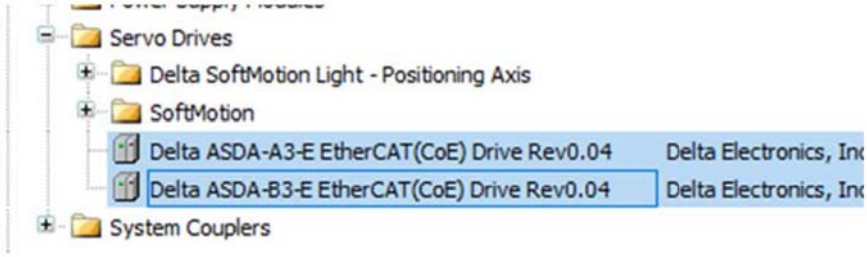




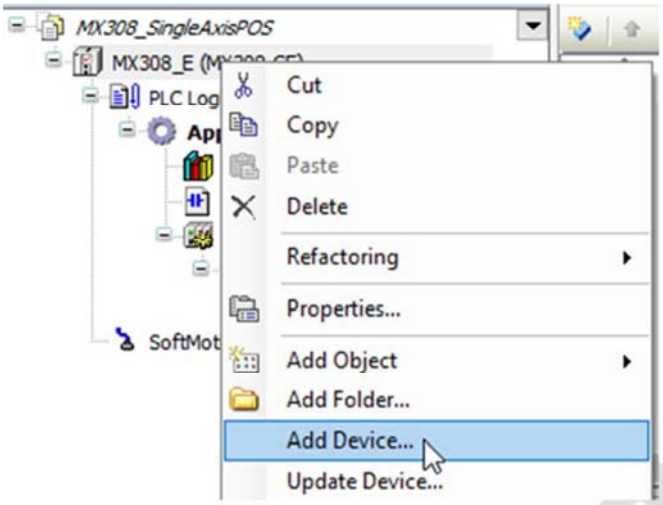
В качестве примера установим XML файл для сервопривода Delta ASD-B3-E:



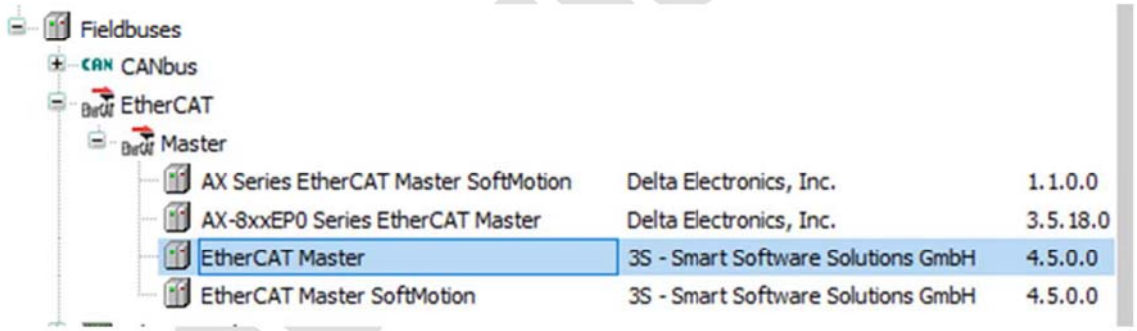
После установки файла в списке устройств появятся две записи:



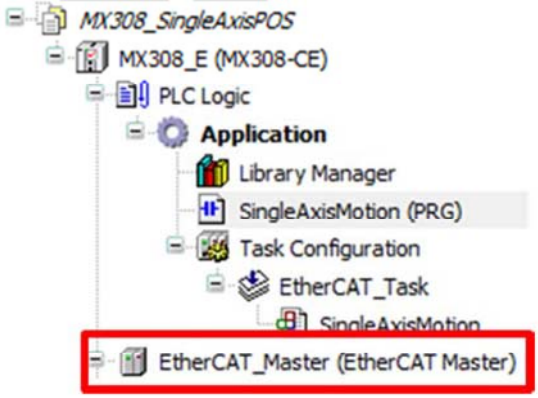
Далее необходимо добавить в проект EtherCAT адаптер. Для этого щёлкните правой кнопкой на названии контроллера и в меню выберите пункт **Add Device**:



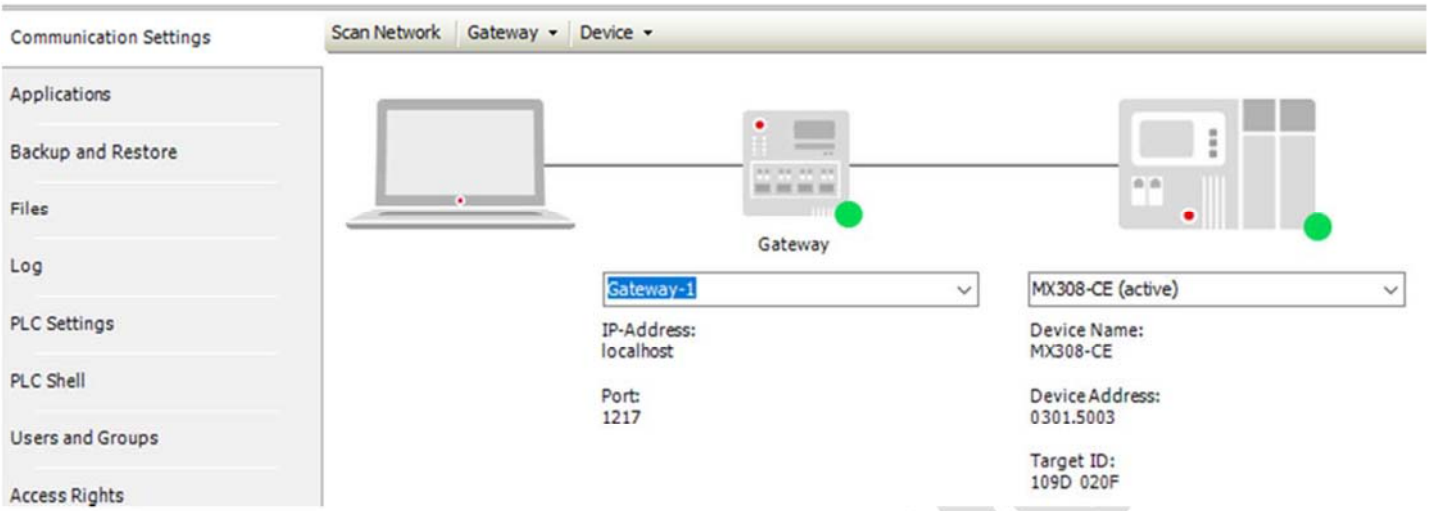
В открывшемся окне выберете пункт **EtherCAT - Master - EtherCAT Master**:



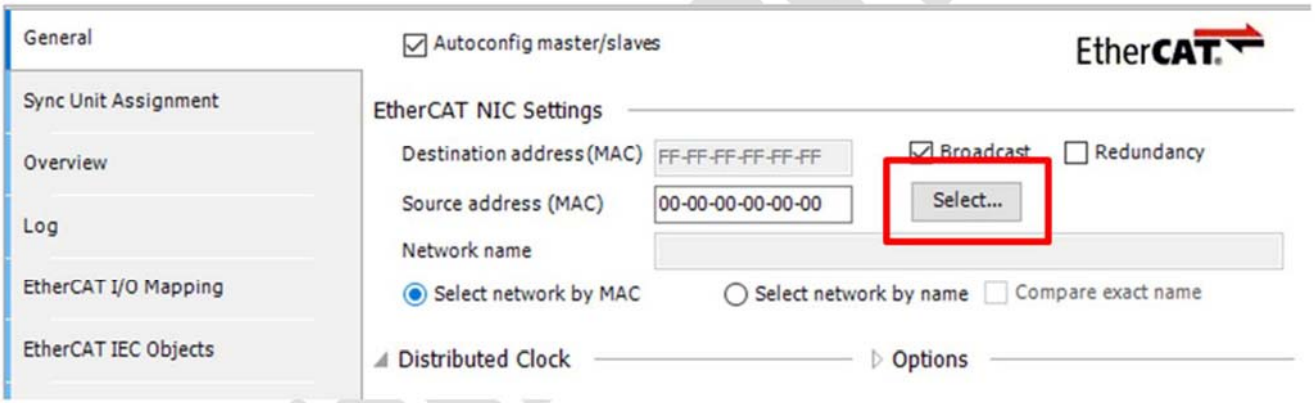
В древе проекта появится пункт:



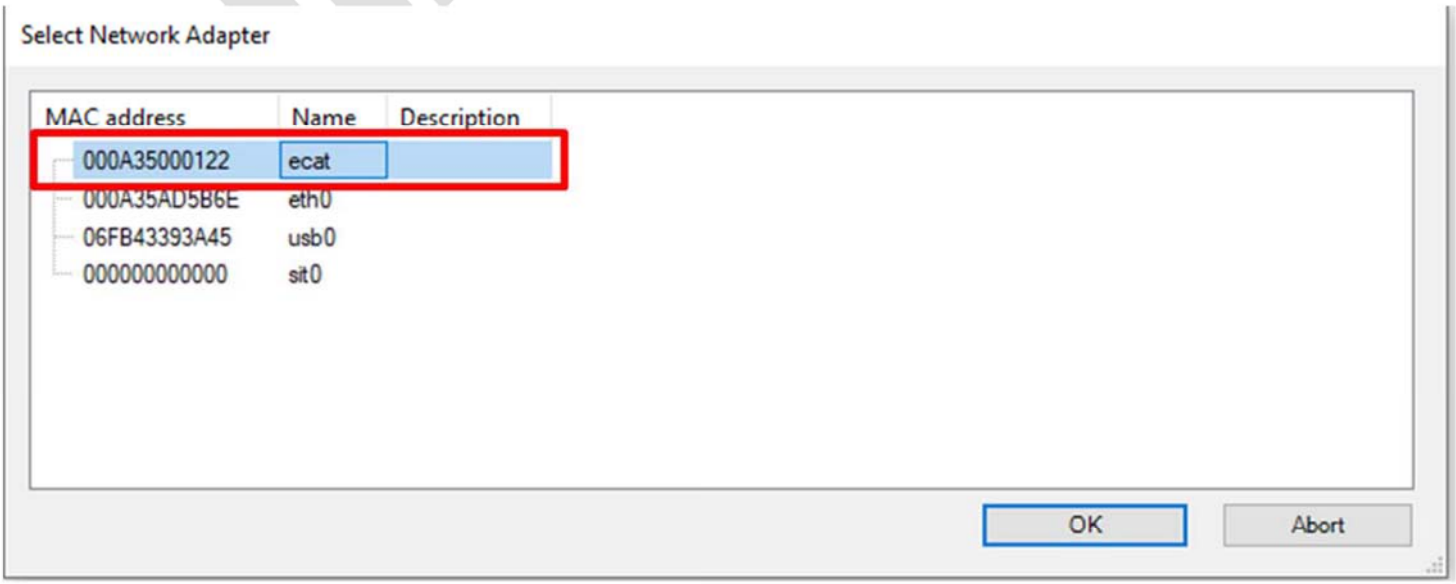
Далее необходимо назначить в проекте Мастера сети EtherCAT. В нашем примере это будет контроллер MX300, поэтому необходимо установить с ним связь (см. соответствующий раздел данного Руководства):



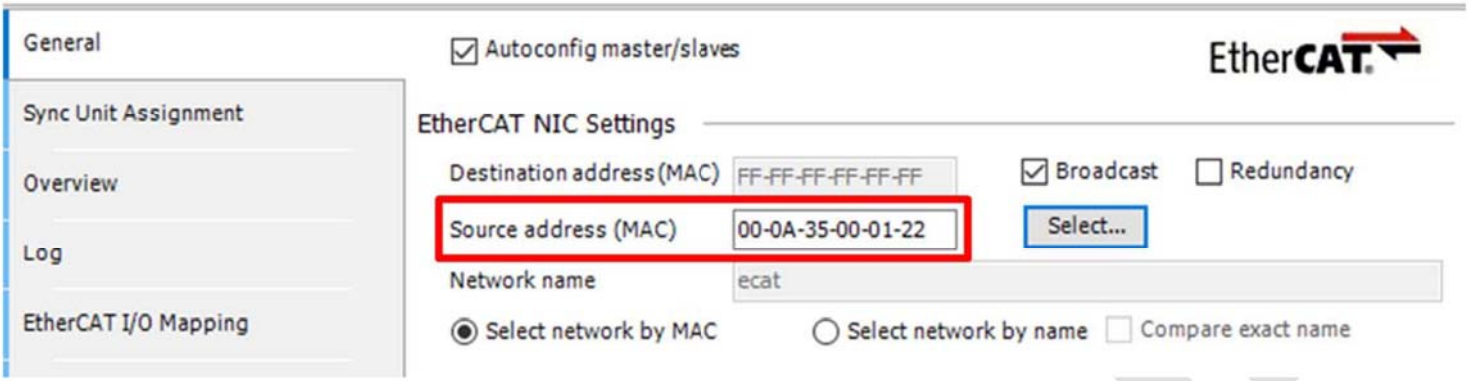
Далее щёлкните дважды левой кнопкой мышки на пункте **EtherCAT Master** и в открывшейся вкладке выберите пункт **General** и нажмите кнопку **Select**:



В открывшемся окне выберите пункт (MAC адрес у каждого контроллера свой):

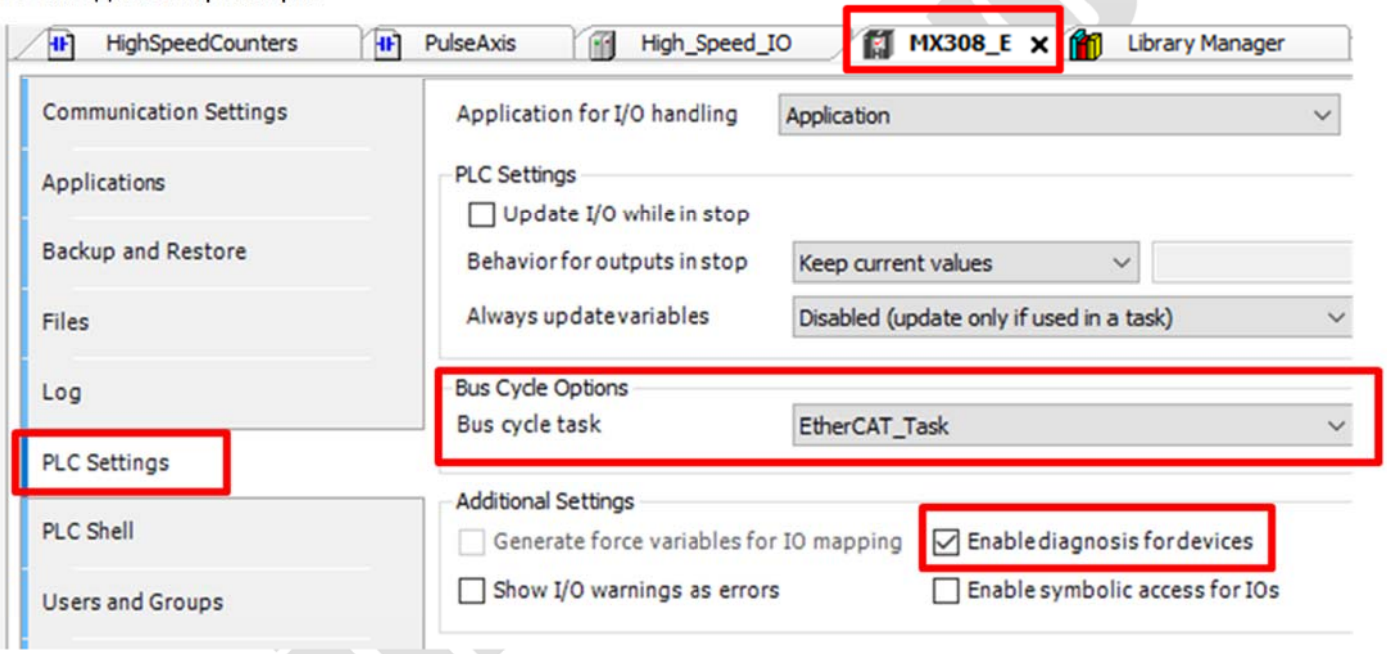


MAC адрес должен установиться в данном поле:

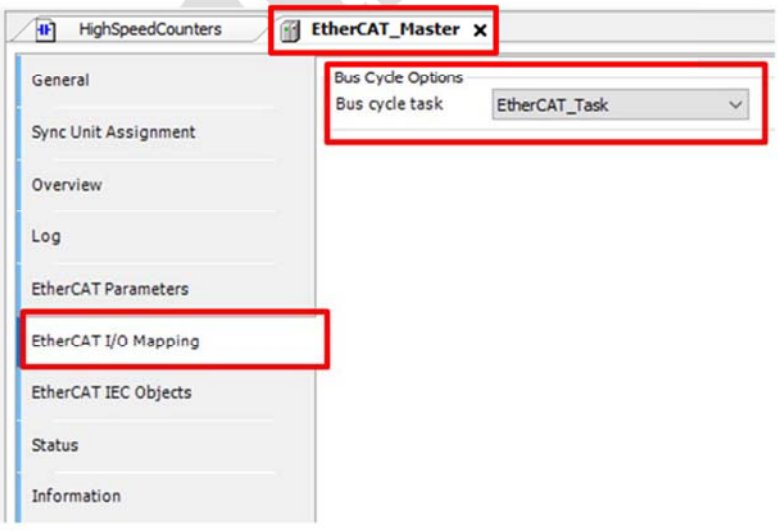


Далее необходимо активировать задачу EtherCAT_Task. Для этого сделайте следующие настройки:

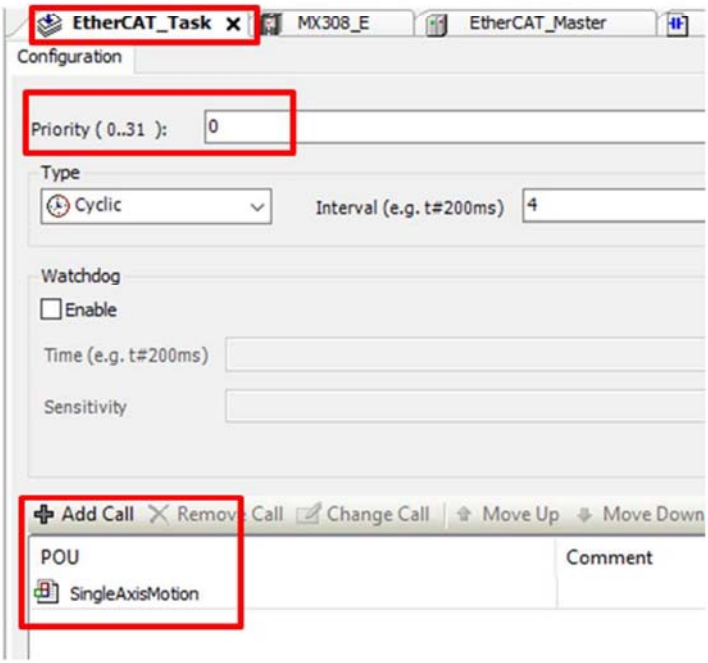
Во вкладке Контроллера:



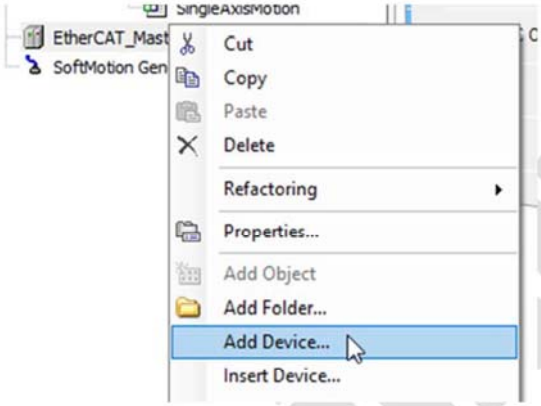
Во вкладке EtherCAT_Master:



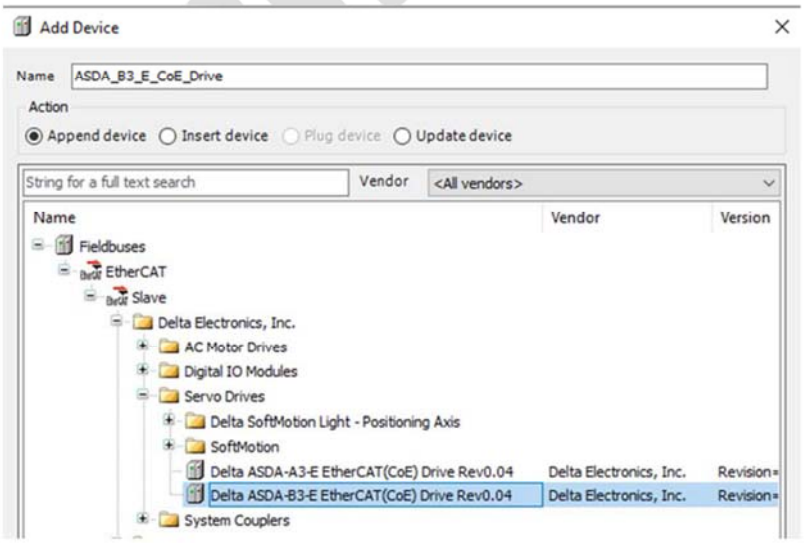
Во вкладке задачи выставить высший приоритет (0), привязать нужные POU:



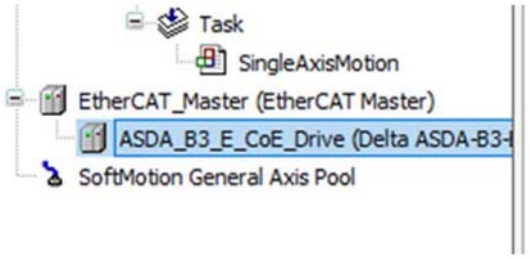
Далее необходимо добавить в проект сервопривод. Для этого щёлкните правой кнопкой мышки на пункте **EtherCAT Master** и в открывшемся окне выберите пункт **Add Device**:



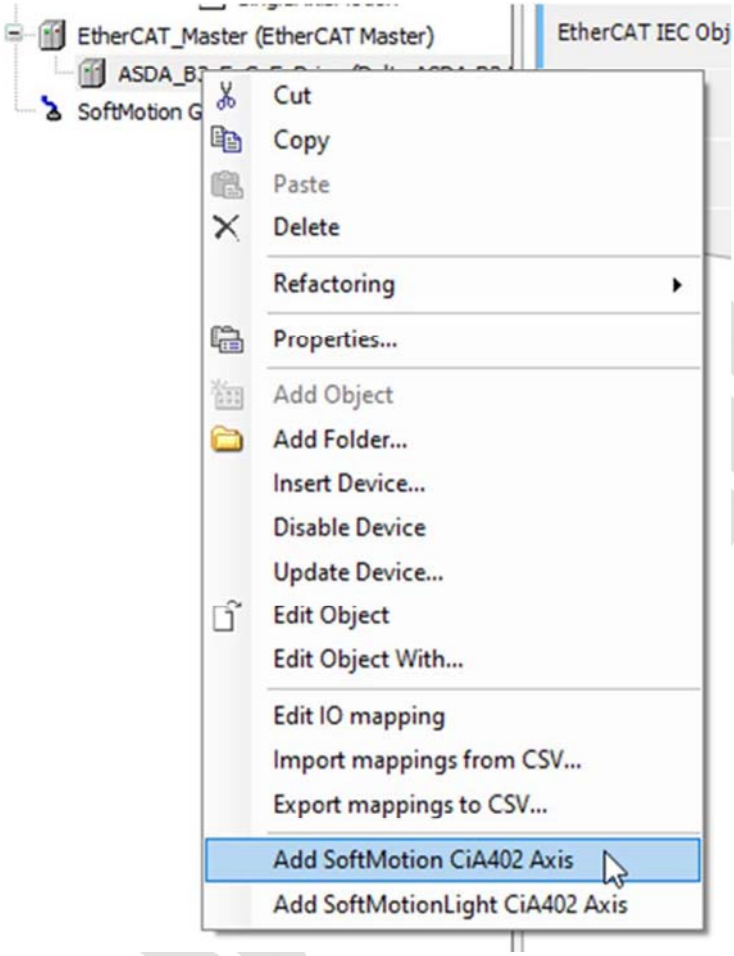
В открывшемся окне выберите сервопривод Delta ASD-B3-E:



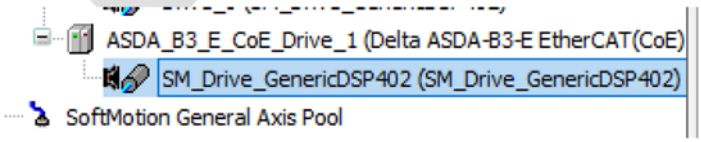
В древе проекта появится пункт:



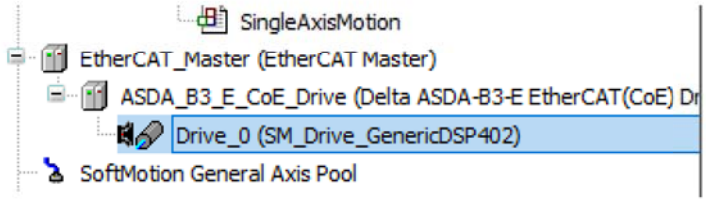
Далее к данному узлу необходимо добавить ось движения стандарта CiA402. Для этого щёлкните правой кнопкой мышки на пункте **ASDA_B3_E_CoE_Drive** и выберите пункт **Add SoftMotion CiA402 Axis**:



Появится пункт:



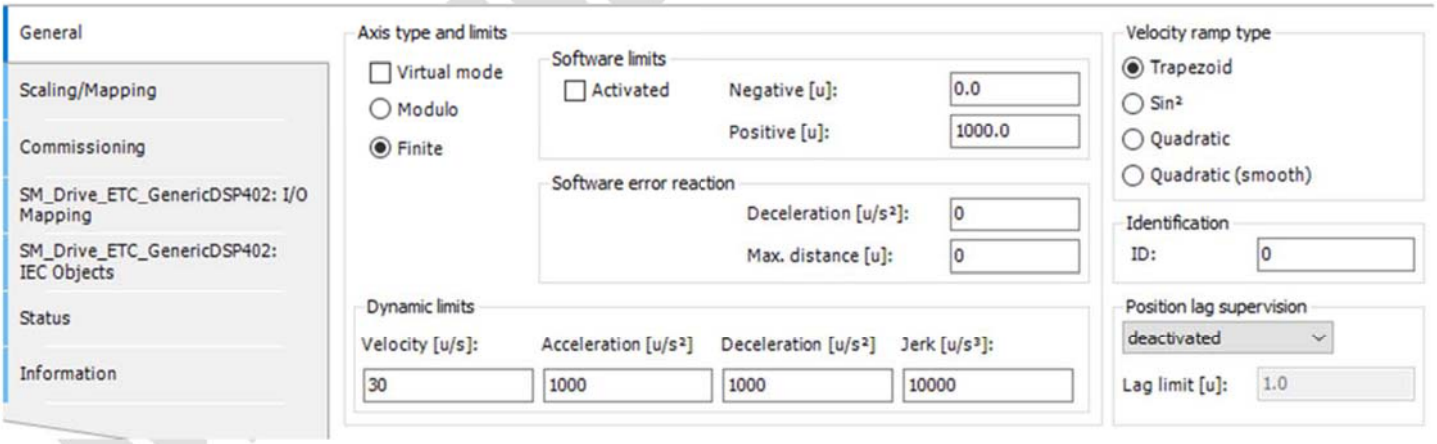
Название оси можно переименовать, например в Drive_0:



Далее необходимо произвести настройку узла и оси движения.

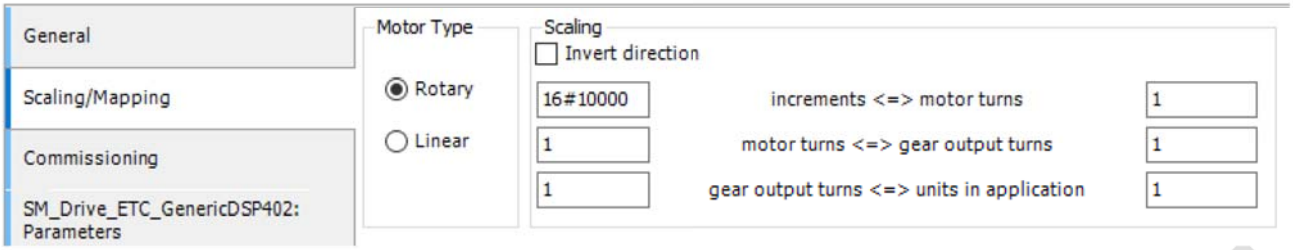
Для настройки узла щёлкните на пункте **ASDA_B3_E_CoE_Drive** два раза левой кнопкой мышки и в отрывшейся вкладке выберите пункт **Process Data**, в котором отметьте нужные наборы PDO:

Для настройки оси движения щёлкните на пункте **Drive_0** два раза левой кнопкой мышки и в отрывшейся вкладке выберите пункт **General**.

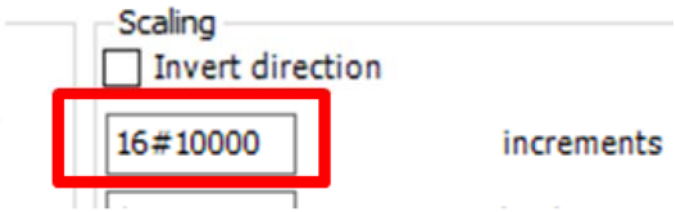


Так как в инструкциях управления движением все параметры, связанные с движением, задаются в единицах пользователя, то перед началом настройки оси необходимо вычислить все необходимые параметры движения и внести в разделы **General** и **Scaling/Mapping**.

Вначале необходимо заполнить ключевые параметры в разделе **Scaling/Mapping:**



В верхней строке в левом поле необходимо ввести количество импульсов на оборот сервопривода. Это должно строго совпадать с настройками привода.



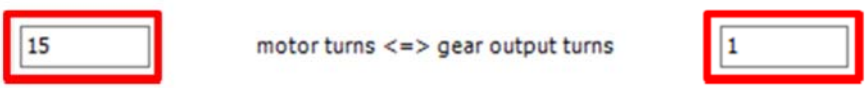
В нашем примере используется сервопривод Delta ASD-B3-E, мотор которого имеет энкодер 24 бит. Это составляет 16 777 216 импульсов на оборот. По умолчанию числитель и знаменатель стоят 1. Значит в данное поле при таких настройках нужно ввести число 16777216 (на 1 оборот, т.е. поле справа остаётся равным 1).

Если к примеру установить в приводе вот такие числитель и знаменатель:

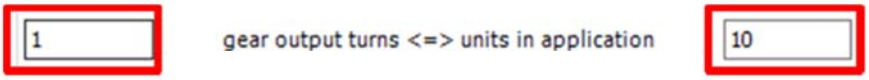
Числитель P1-044 = 16 777 216
Знаменатель P1-045 = 100 000

то мы получим число 100 000 импульсов на оборот, которое нужно будет ввести в поле:

Следующее поле – это редуктор. Например, если у Вас понижающий редуктор 1:15, то нужно ввести такие числа:



Во многих случаях редуктор не вводят, а учитывает его в конечном перемещении, которое вводится в третьей строчке. В нашем примере это 10 единиц.



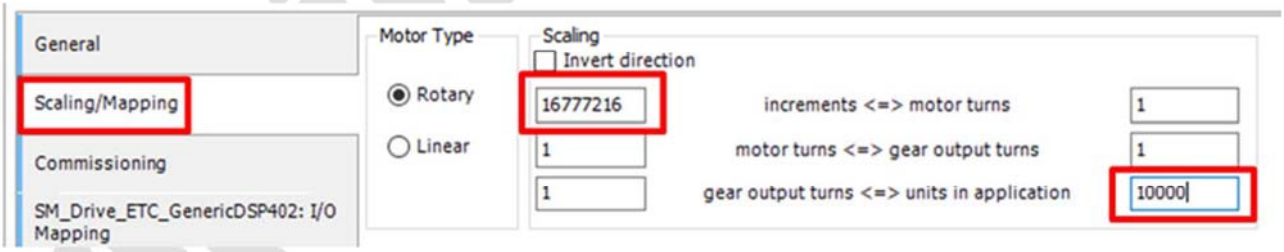
Это ключевой параметр, который увязывает обороты мотора сервопривода и перемещение конечного механизма. В примере выше – 1 оборот мотора сервопривода вызывает перемещение конечного механизма (например ШВП) на 10 пользовательских единиц, которые могут соответствовать любым единицам длины (мкм, мм, дюймы и т.д.).
Таким образом, Вы устанавливаете жёсткую связь между заданием в единицах длины и единицами мотора (импульсами на оборот). В командах движения задание перемещения устанавливается в единицах пользователя, т.е. длины, и это автоматически пересчитывается в задание для мотора в его единицах (импульсы на оборот по внутреннему энкодеру).

В итоге для данного случая получатся следующие настройки:

Для улучшения точности, перемещение конечного механизма на 1 оборот лучше измерять прямым способом высокоточным измерительным инструментом (микрометром) и уже на основе объективных данных устанавливать линейные единицы пользователя.
Также, при прямом измерении можно учесть редуктор (коэффициент редукции), т.е. учесть его в перемещении конечного механизма.

Кроме того, для устранения нарастающей погрешности при вычислениях перемещение конечного механизма лучше задавать в крупных числах. Например, вместо 10 мм использовать 10 000 мкм.

Для дальнейших вычислений скорости и ускорения в единицах пользователя примем следующие параметры линейного перемещения и количества импульсов на оборот двигателя:

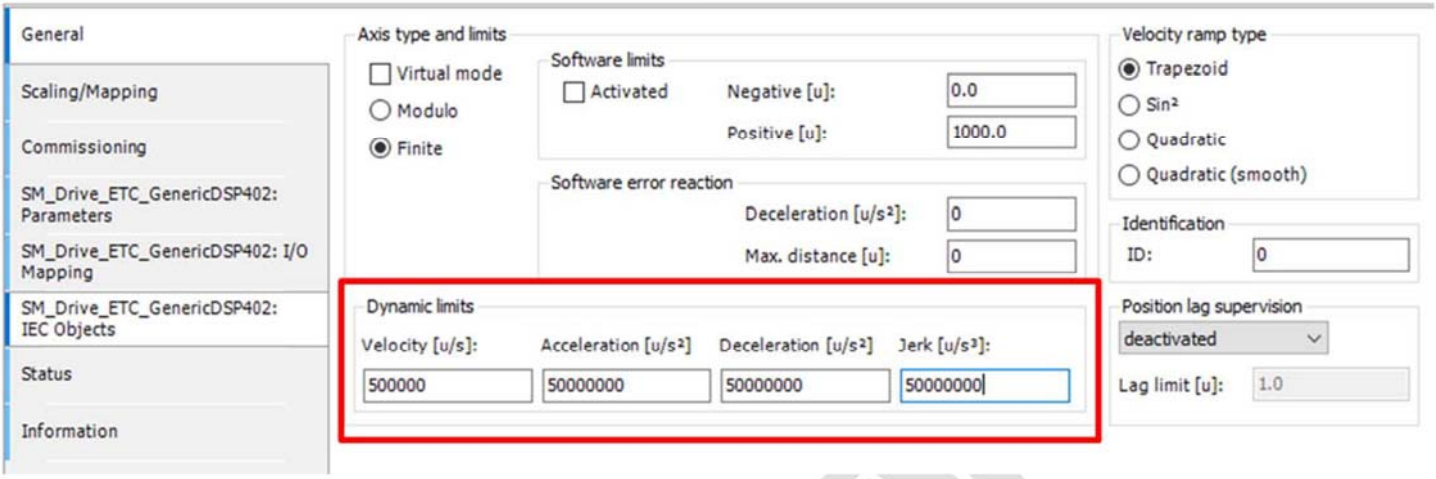


Расчёт максимальной скорости:
Максимальная скорость вращения мотора 3000 об/мин. В секунду 3000 : 60 = 50 об/сек
За один оборот будет пройдено 10 000 ед. длины в единицах пользователя
Следовательно максимальная скорость: 50 * 10000 = 500 000 ед. в секунду
Это максимально допустимая скорость, которую можно задавать в инструкциях движения.

Расчёт максимально допустимого ускорения/замедления.
Сначала рассчитывается теоретически достижимое ускорение при разгоне до 3000 об/мин за 1 мс.
Исходя из полученных выше расчётов за 1 секунду мы можем достичь 500 000 ед. длины/сек. Для достижения такой скорости за 1 мс это значение надо умножить на 1000 (т.е. перевести секунды в мс).

Получается число 500 000 000 ед./сек2
Это предельно теоретически возможное ускорение в единицах пользователя, соответствующее разгону привода до 3000 об/мин за 1 мс.

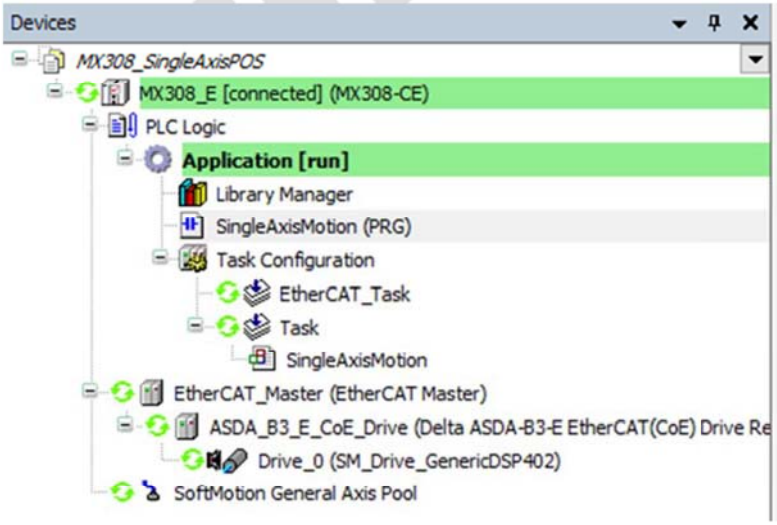
Теперь необходимо выставить предельные значения скорости и ускорения во вкладке **General**:



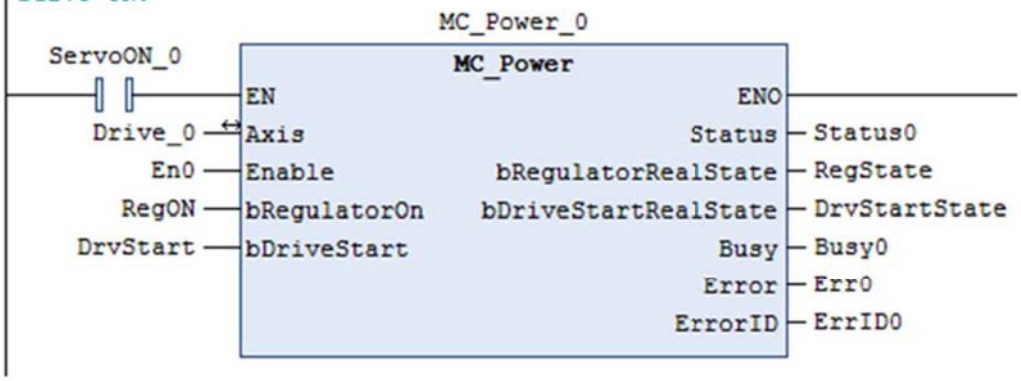
В целях безопасности предельное ускорение уменьшено в 10 раз. Т.е. приводу разрешается разгоняться до 3000 об/мин как минимум за 10 мс (50 000 000 ед./сек2). Такое же значение нужно выставить и для параметра Jerk

В инструкциях движения позиция, скорость и ускорение задаются в единицах пользователя согласно полученных вычислений.
Например (при перемещении на 10000 ед. за 1 оборот двигателя):
Ускорение 100 мс = 5 000 000 ед/сек2
Ускорение 20 мс = 25 000 000 ед/сек2
Скорость 3000 об/мин = 500 000 ед/сек
Скорость 300 об/мин = 50 000 ед/сек
Скорость 15 об/мин = 2500 ед/сек
Перемещение на 10 мм = 10000 ед.
Перемещение на 20,456 мм = 20456 ед.
Перемещение на 300 мм = 300 000 ед.

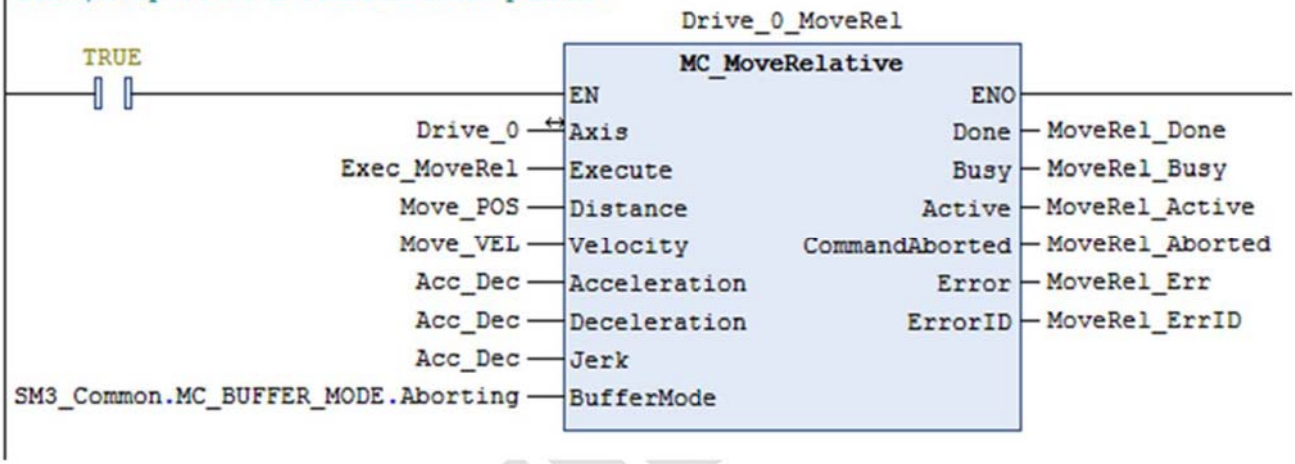
Ниже приводятся для примера ряд инструкций для одноосевого движения:

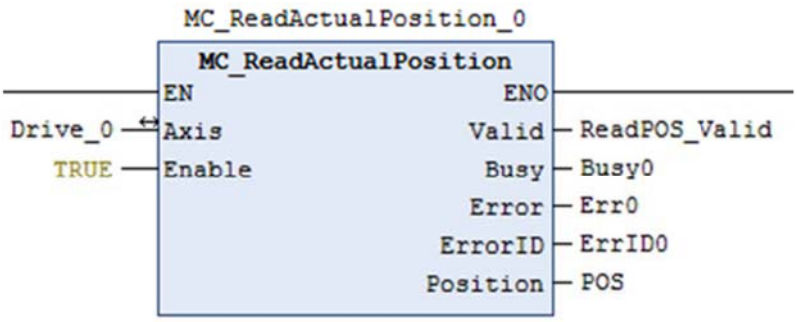


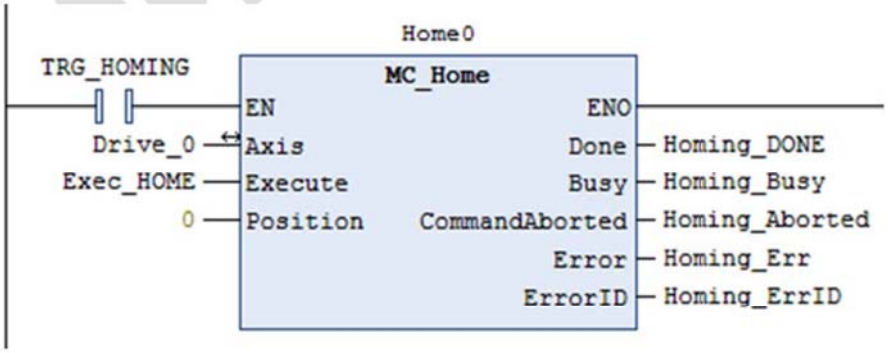
Servo-ON.



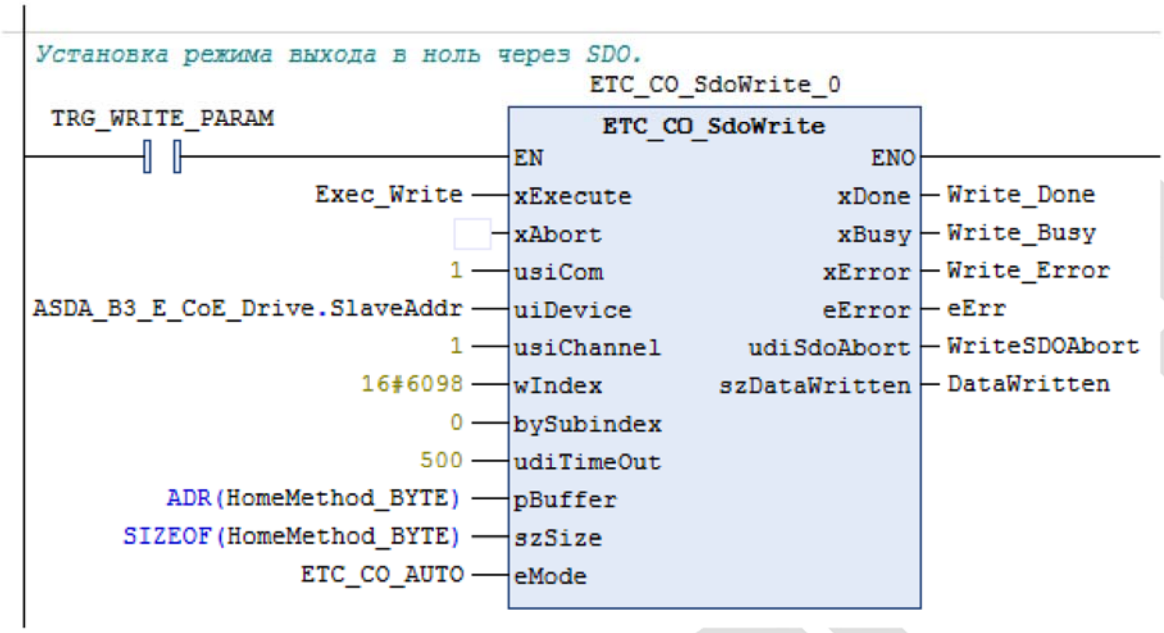
Позиционирование в относительном режиме.

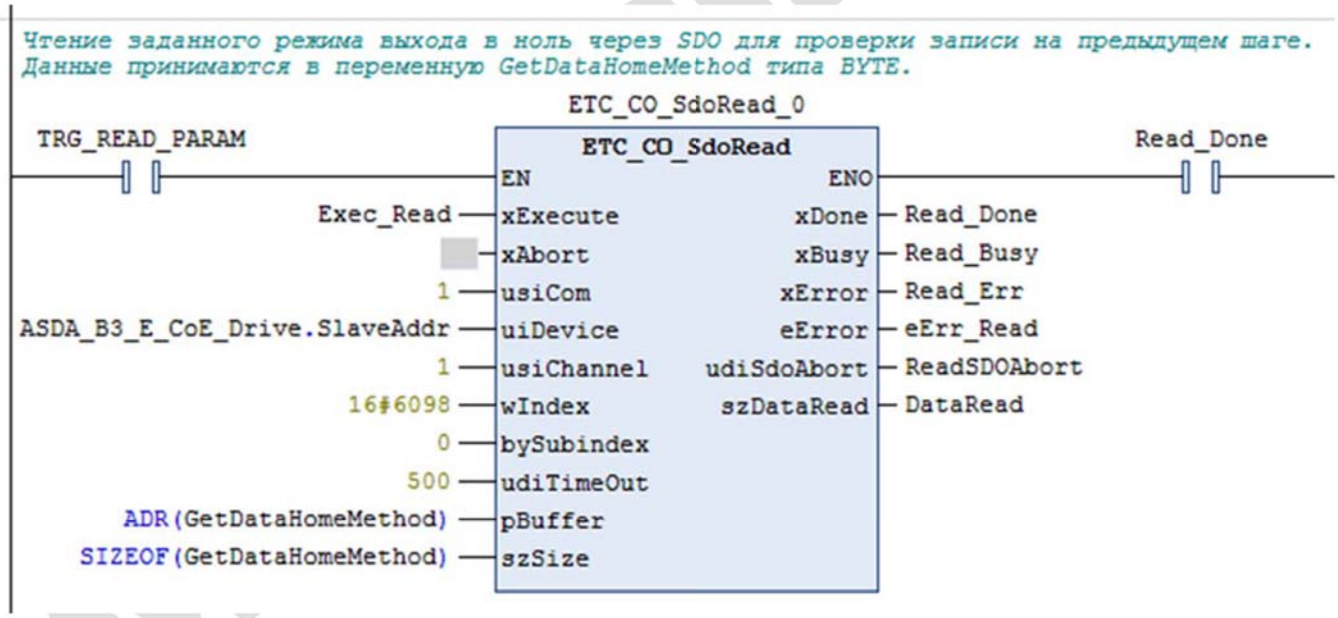




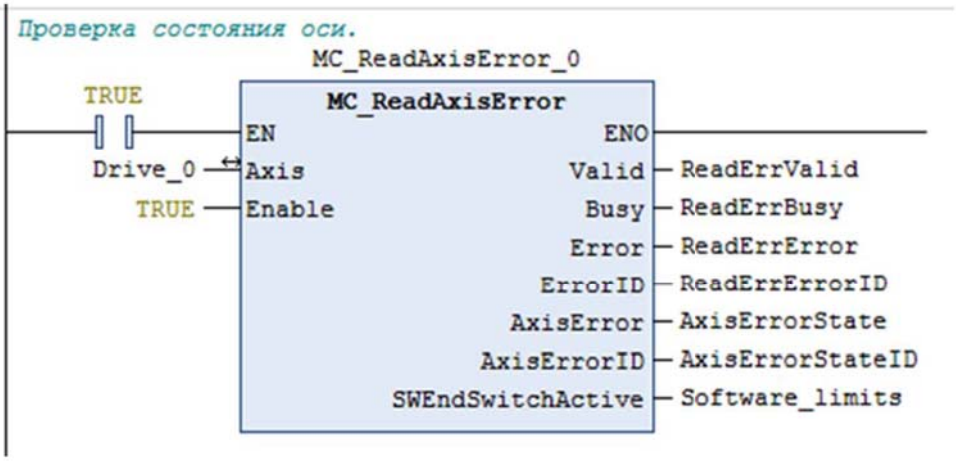


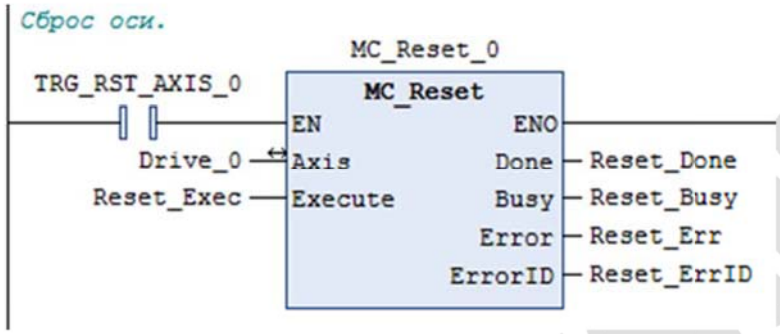
Для установки режима выхода в ноль, а также записи/чтению других параметров привода нужно использовать команды:

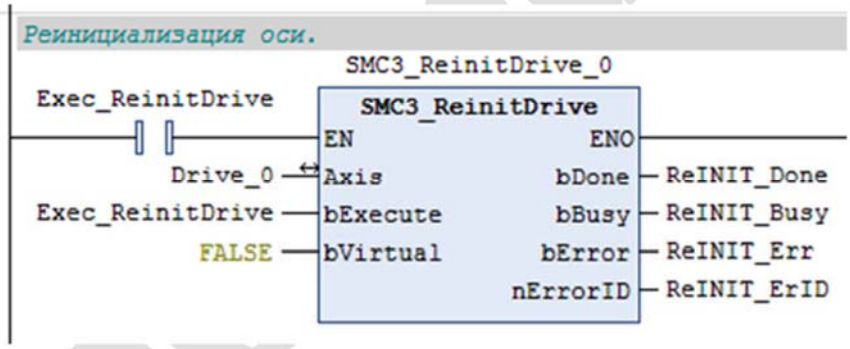




Для сброса оси и реинициализации привода используются команды:



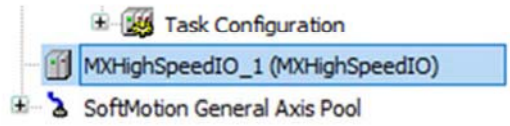




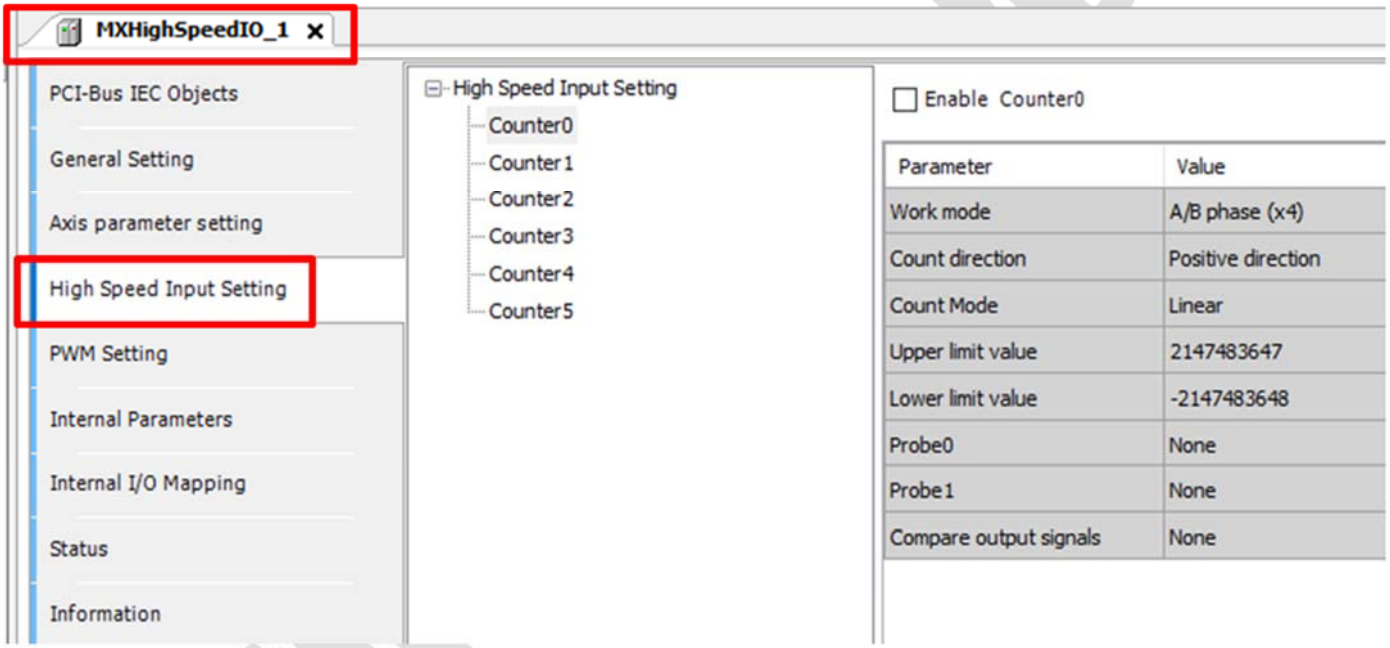
Работа с высокоскоростными счётчиками

Контроллер имеет на борту 6 AB-каналов высокоскоростного счёта (входы 00-11). Далее описывается последовательность настройки высокоскоростных счётчиков.

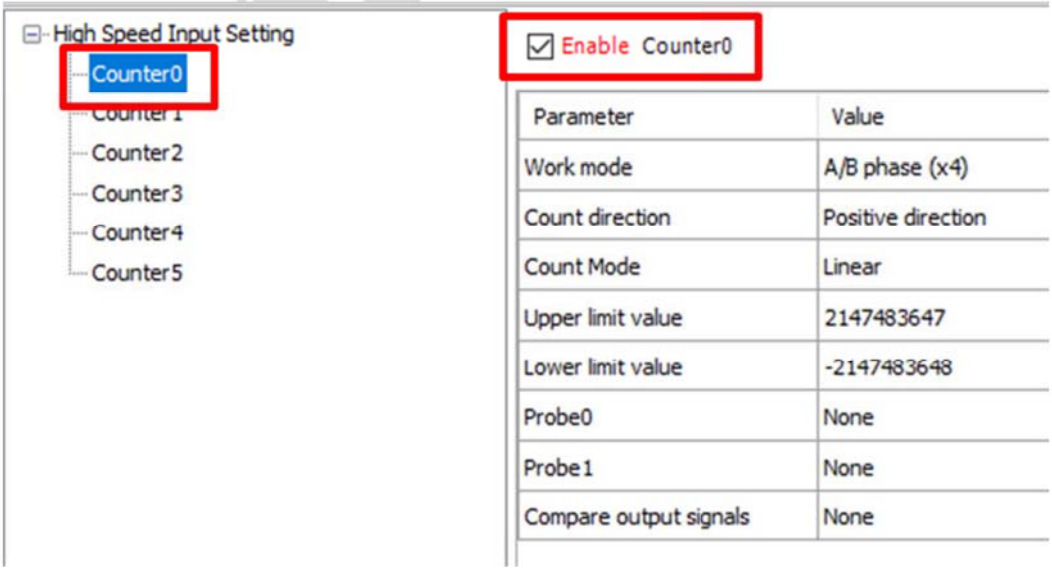
Добавьте в проект адаптер входов-выходов ЦПУ **MXHighSpeedIO** (см. данное Руководство). В древе проекта появится пункт **MXHighSpeedIO_1 (MXHighSpeedIO)**:



Задание параметров работы высокоскоростных счётчиков
Щёлкните дважды на пункте“ **MXHighSpeedIO_1 (MXHighSpeedIO)**” в древе проекта и в открывшейся вкладке выберите раздел “**High Speed Input Setting**”:



Выберите нужный счётчик и поставьте флажок Enable Counter*:

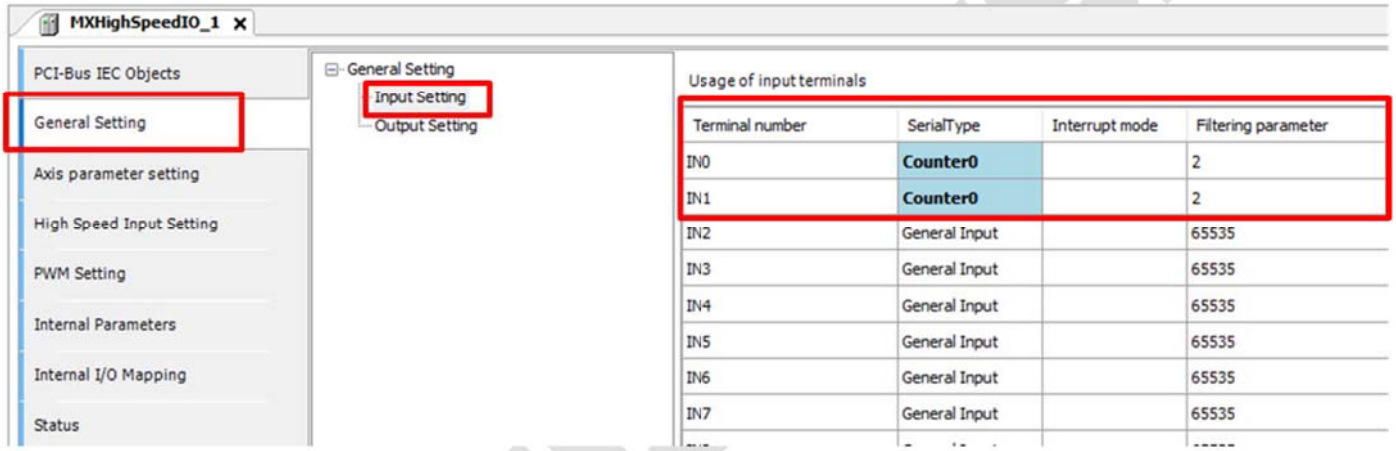


В данной вкладке можно сделать настройки высокоскоростного счётчика:

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
Probe0 – вход для инструкции LS_TouchProbe (привязка по названию энкодерной оси)
Probe1 – вход для инструкции LS_TouchProbe (привязка по названию энкодерной оси)
Compare Output Signals – выход для инструкции LS_Compare (привязка по названию энкодерной оси)

Пункт **Internal Parameters** служит для информативных целей в нём ничего менять не надо. Он сам модифицируется в зависимости от настроек в пункте **High Speed Input Setting**.

После выбора счётчика во вкладке **General Setting** будут отмечены входы, к которым данный счётчик привязан, и можно установить время фильтра:

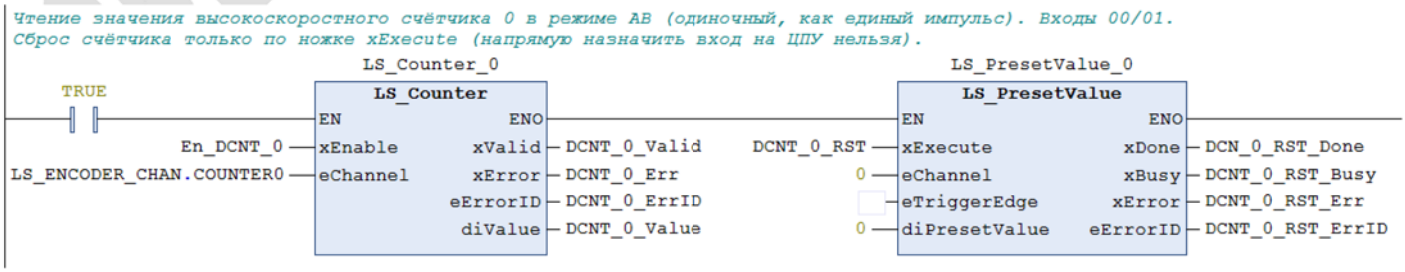


Для обращения к счётчикам в программе используйте команды из библиотеки **MC_HSIO**:

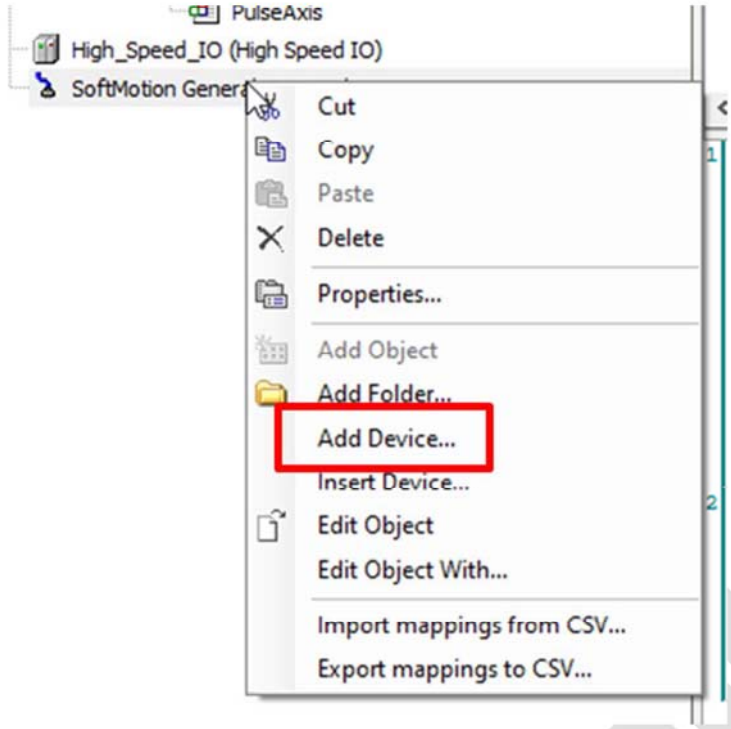
Высокоскоростной счетчик	LS_Counter	FB	Инструкции высокоскоростного счетчика
	LS_PresetValue	FB	Команда предустановленного значения счетчика

LS_Counter - для получения текущего значения счётчика (ножка eChannel – номер счётчика, 0 – 5)

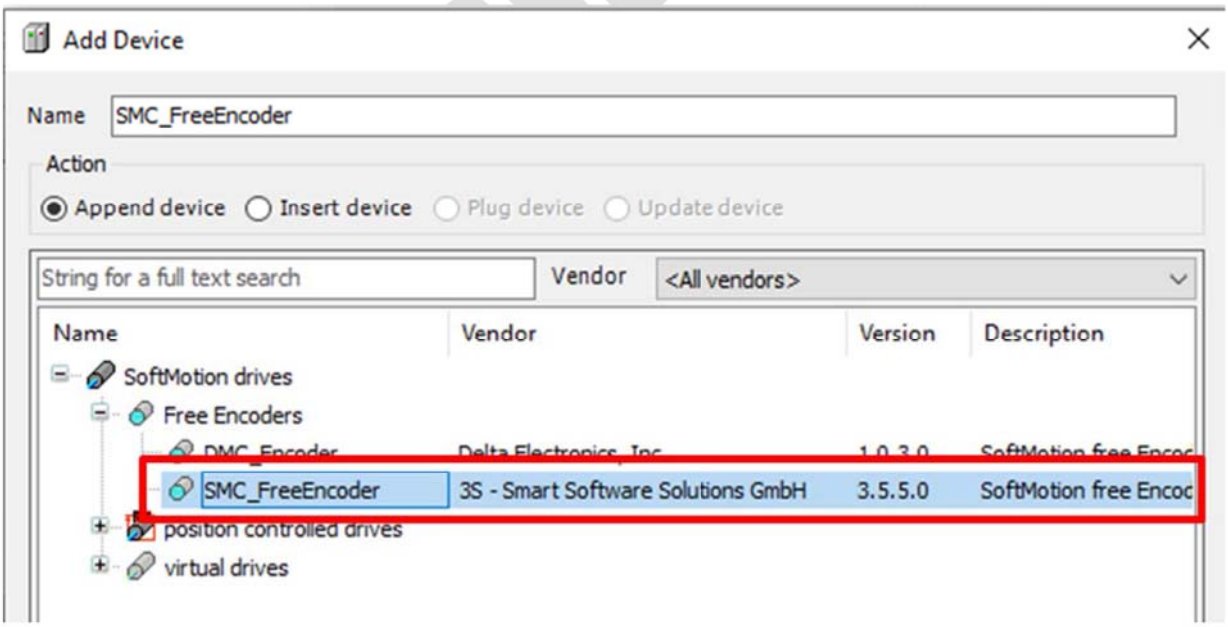
LS_PresetValue - для обнуления счётчика (по сигналу на ножке xExecute)



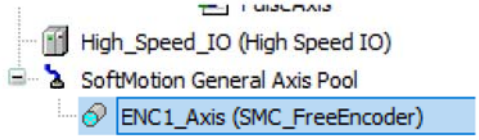
Также, высокоскоростной счётчик можно привязать к энкодерной оси и использовать в качестве Мастер-оси в различных инструкциях движения.
Для этого щёлкните правой кнопкой мышки на пункте **SoftMotion General Axis Pool** и в отрывшемся меню выберите пункт **Add Device**:



В открывшемся окне выберите пункт **SMC_FreeEncoder**:

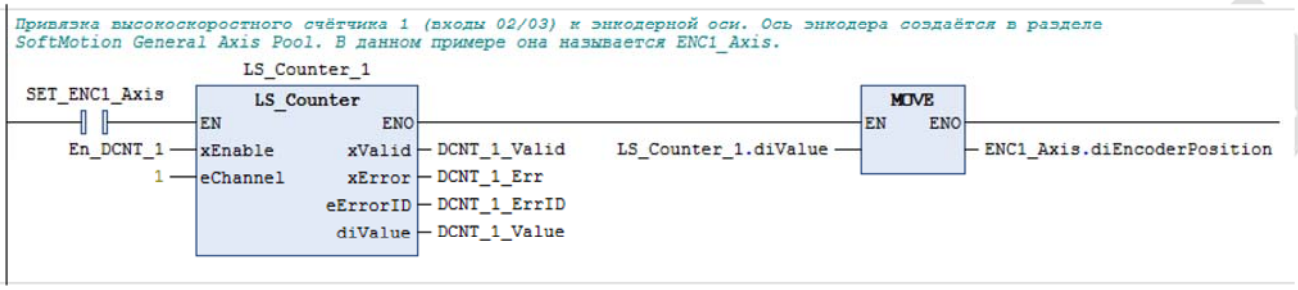


В древе проекта появится энкодерная ось, которую можно переименовать в любое удобное название, например **ENC1_Axis**:



Высокоскоростной счётчик привязывается к энкодерной оси командой присвоения (для данного примера):

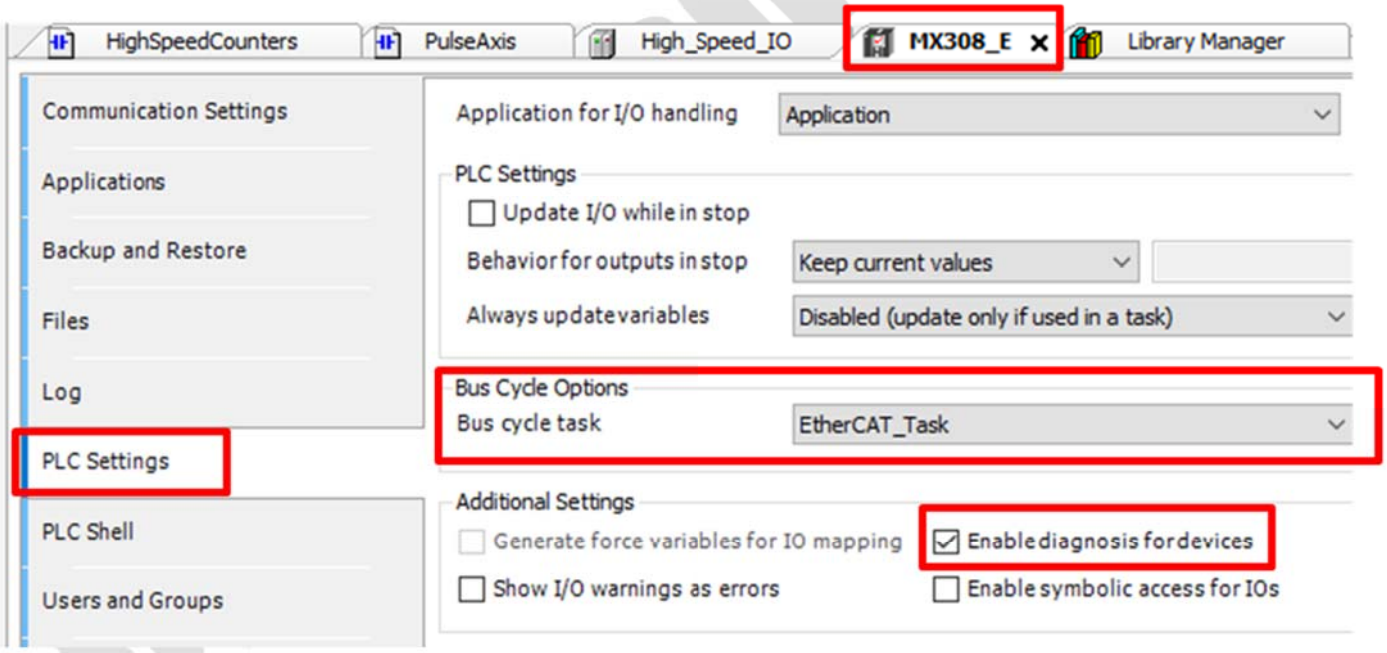
ENC1_Axis.diEncoderPosition:=LS_Counter_1.diValue;



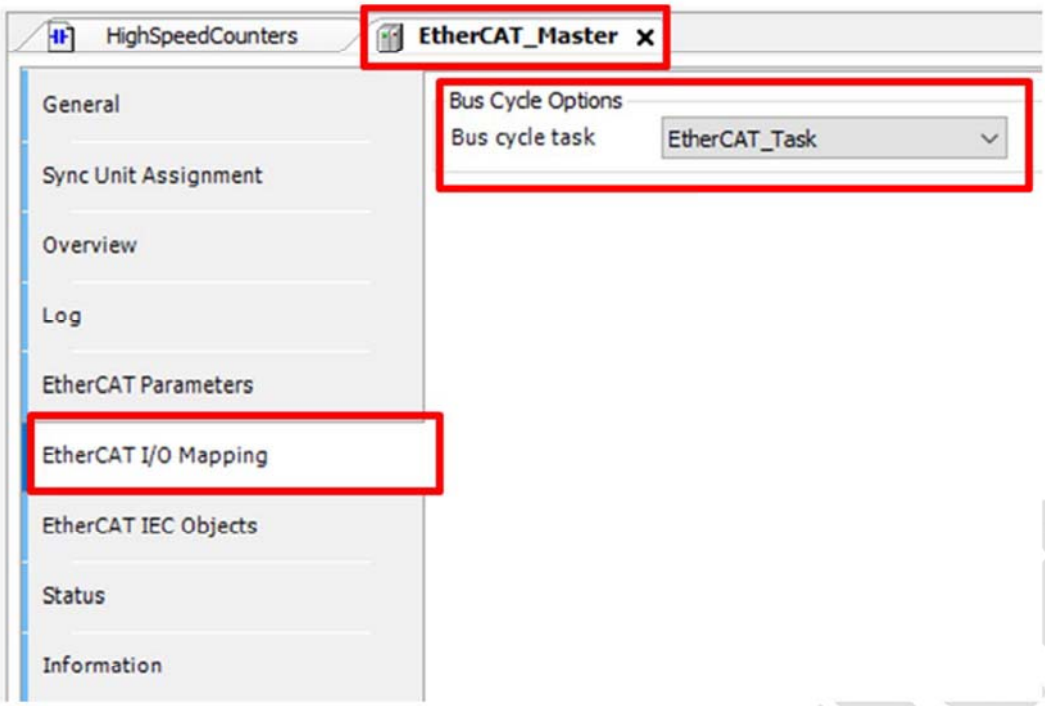
Внимание! POU, где используется энкодерная ось и счётчик необходимо привязать к задаче EtherCAT_Task

Для этого необходимо сделать ряд настроек:

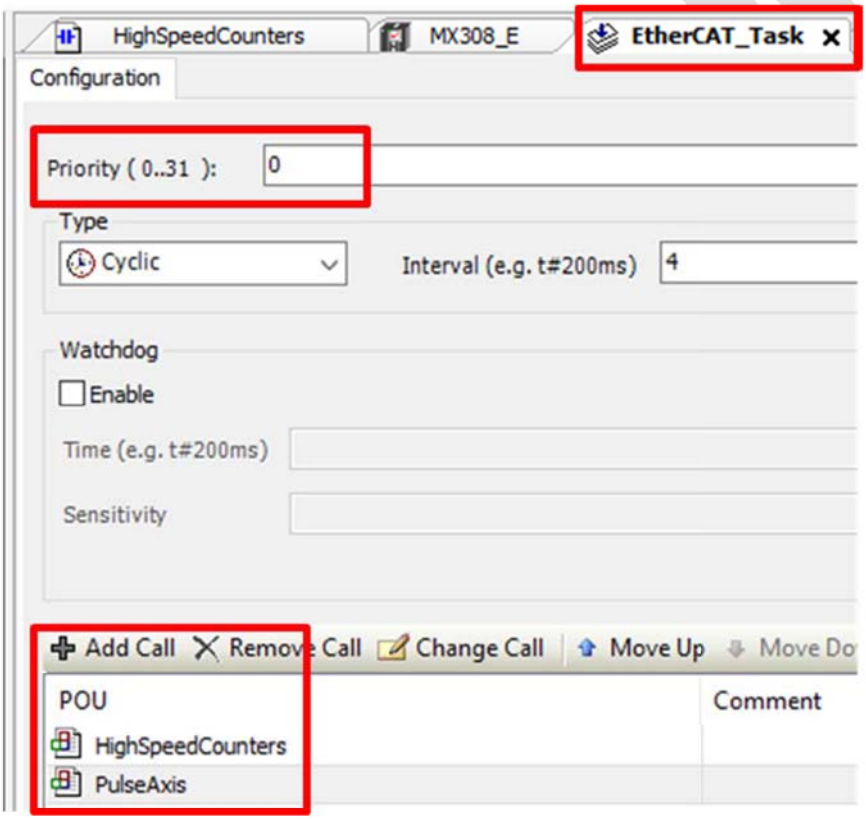
Во вкладке контроллера:



Во вкладке EtherCAT_Master:



Во вкладке **EtherCAT_Task** данной задаче необходимо поставить высший приоритет (0) и привязать необходимые POU:



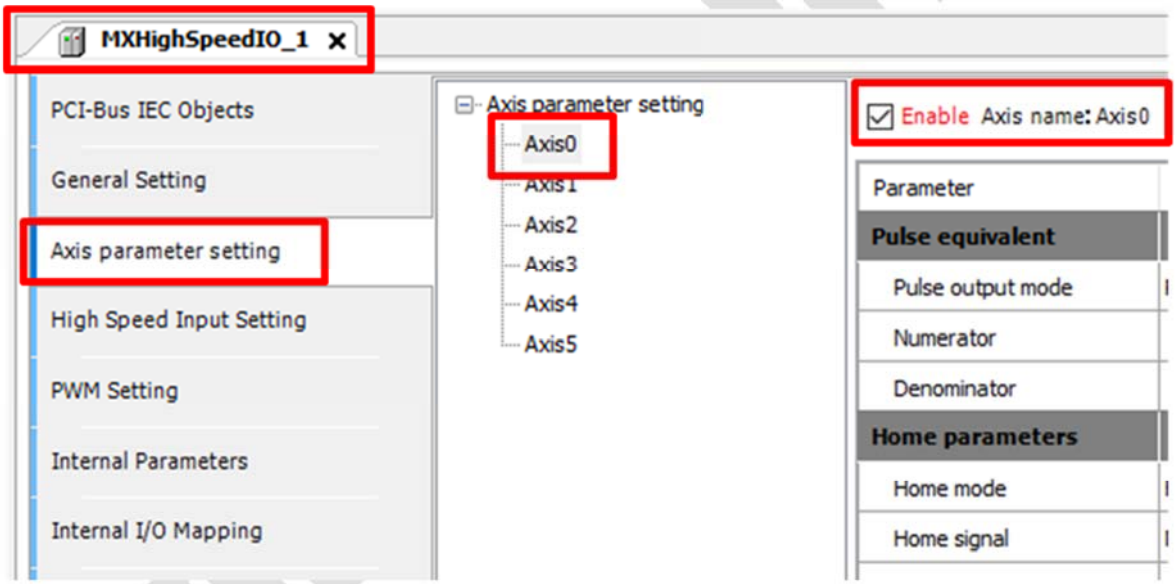
Импульсная ось движения

Контроллеры серии MX300 имеют 12 импульсных выходов по 200 кГц, сгруппированных в 6 осей. Каждая ось соответственно занимает по 2 импульсных выхода: импульс/направление или AB режим. Остальные 4 выхода 12-15 являются обычными.

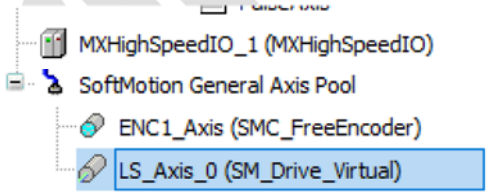
0	Импульсный выход	AXIS 0 PUL	8	Импульсный выход	AXIS 4 PUL
1	Импульсный выход	AXIS 0 DIR	9	Импульсный выход	AXIS 4 DIR
2	Импульсный выход	AXIS 1 PUL	10	Импульсный выход	AXIS 5 PUL
3	Импульсный выход	AXIS 1 DIR	11	Импульсный выход	AXIS 5 DIR
4	Импульсный выход	AXIS 2 PUL	12	Обычный выход	
5	Импульсный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

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

Для добавления импульсной оси щёлкните дважды мышкой в древе проекта на пункте **MXHighSpeedIO** и в открывшейся вкладке выберите пункт

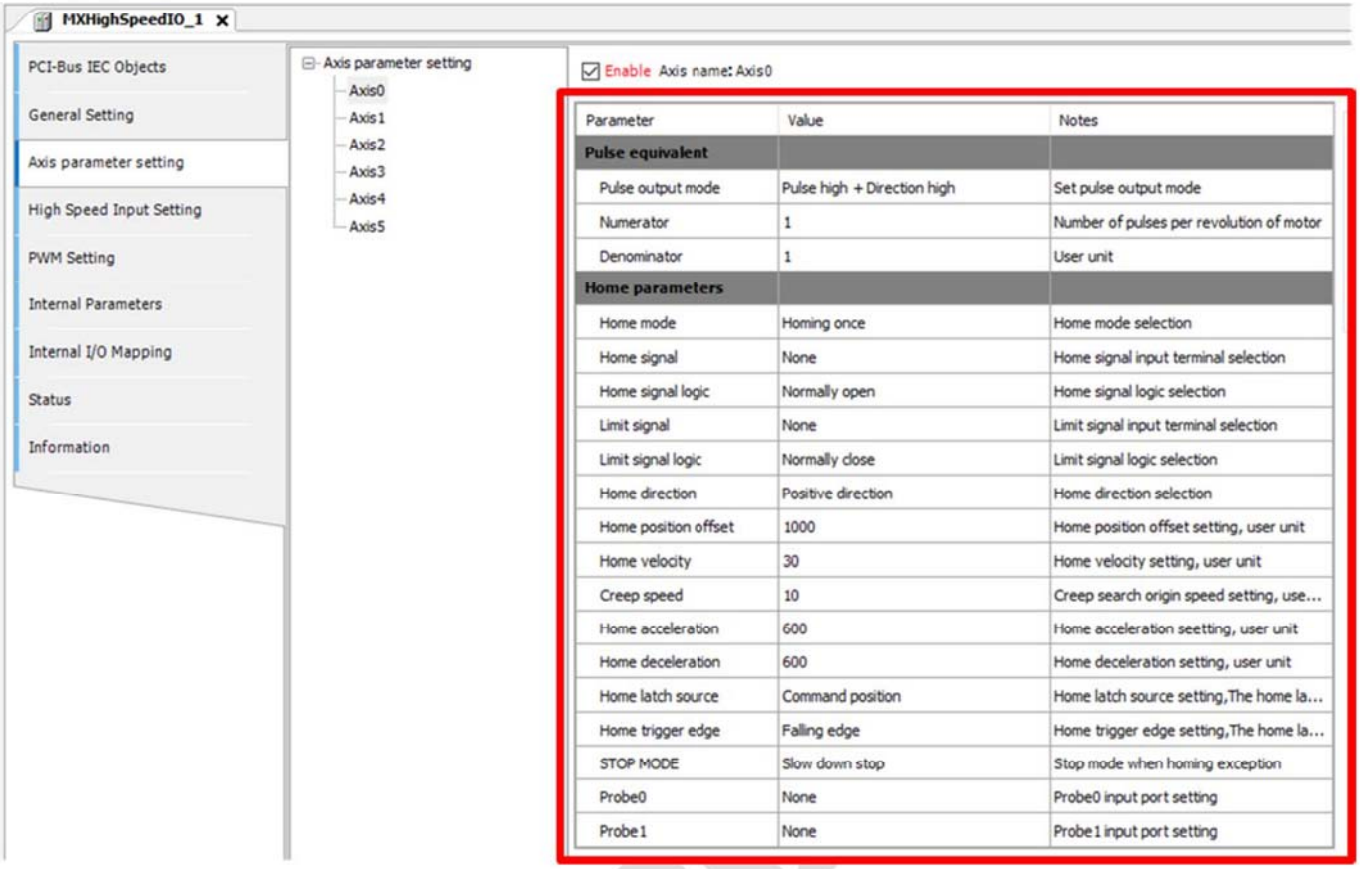


В древе проекта появится ось **LS_Axis_0 (SM_Drive_Virtual)**:



Внимание!
Название оси менять нельзя! С данным названием ось будет использоваться во всех ФБ управления движением. А так как импульсная ось создаётся на основе стандартной виртуальной оси, то импульсная ось доступна во всех ФБ управления движения как любая другая серво ось.

Станут доступны настройки параметров оси:



Parameter	Value	Notes
Pulse equivalent		
Pulse output mode	Pulse high + Direction high	Set pulse output mode
Numerator	1	Number of pulses per revolution of motor
Denominator	1	User unit
Home parameters		
Home mode	Homing once	Home mode selection
Home signal	None	Home signal input terminal selection
Home signal logic	Normally open	Home signal logic selection
Limit signal	None	Limit signal input terminal selection
Limit signal logic	Normally close	Limit signal logic selection
Home direction	Positive direction	Home direction selection
Home position offset	1000	Home position offset setting, user unit
Home velocity	30	Home velocity setting, user unit
Creep speed	10	Creep search origin speed setting, use...
Home acceleration	600	Home acceleration seetting, user unit
Home deceleration	600	Home deceleration setting, user unit
Home latch source	Command position	Home latch source setting,The home la...
Home trigger edge	Falling edge	Home trigger edge setting,The home la...
STOP MODE	Slow down stop	Stop mode when homing exception
Probe0	None	Probe0 input port setting
Probe1	None	Probe1 input port setting

Pulse Output mode – Режим выдачи импульсов
Numerator – Количество импульсов на 1 оборот вала мотора
Denominator – Расстояние, проходимое конечным механизмом на 1 вала мотора
Home Mode – Режим выхода в ноль
Home Signal – Сигнал датчика нуля
Home Signal Logic – НО/НЗ исходное состояние датчика нуля
Limit Signal – Датчик предела
Limit Signal Logic – НО/НЗ исходное состояние датчика предела
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
Probe0 – Вход 1 на ЦПУ для захвата позиции оси по внешнему сигналу. Работает с **LS_TouchProbe** FB
Probe1 – Вход 2 на ЦПУ для захвата позиции оси по внешнему сигналу. Работает с **LS_TouchProbe** FB

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

Команды **LS_TouchProbe** и **LS_Home_P** находятся в библиотеке **MC_HSIO**.

Numerator/ Denominator

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

Данное значение заносится в пункт **Numerator** .

В пункт **Denominator** заносится перемещение на 1 оборот в единицах пользователя.

Для примера зададим Numerator = 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

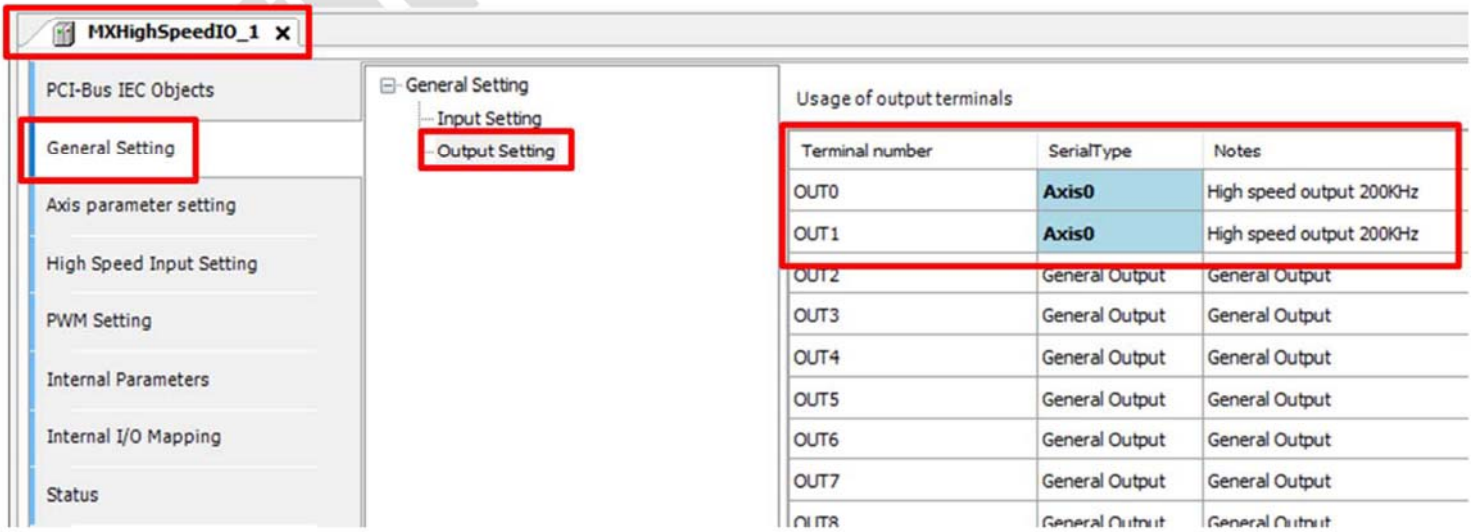
Если при этом задать Denominator = 10 ед. пользователя, то это будет означать, что если в командах движения будет задана позиция 10, то контроллер выдаст 10000 импульсов, т.е. привод сделает 1 оборот.

Режим выхода в ноль (**Home Mode**):

Home mode (Режим выхода в ноль)	Значение
Поиск нуля в одном направлении	0
Поиск нуля в прямом направлении + поиск сигнала при обратном движении	1
Two homing processing	2
Mark homing position	3

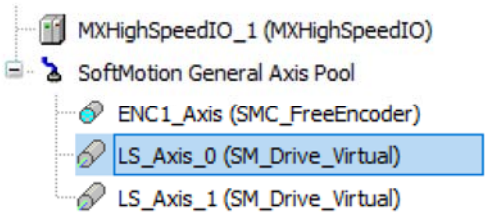
Пункт **Internal Parameters** является внутренним и заполняется автоматически по мере выбора параметров в пункте **Axis Parameter Setting**.

В пункте **General Setting – Output Setting** появится обозначение привязки выходов к оси:

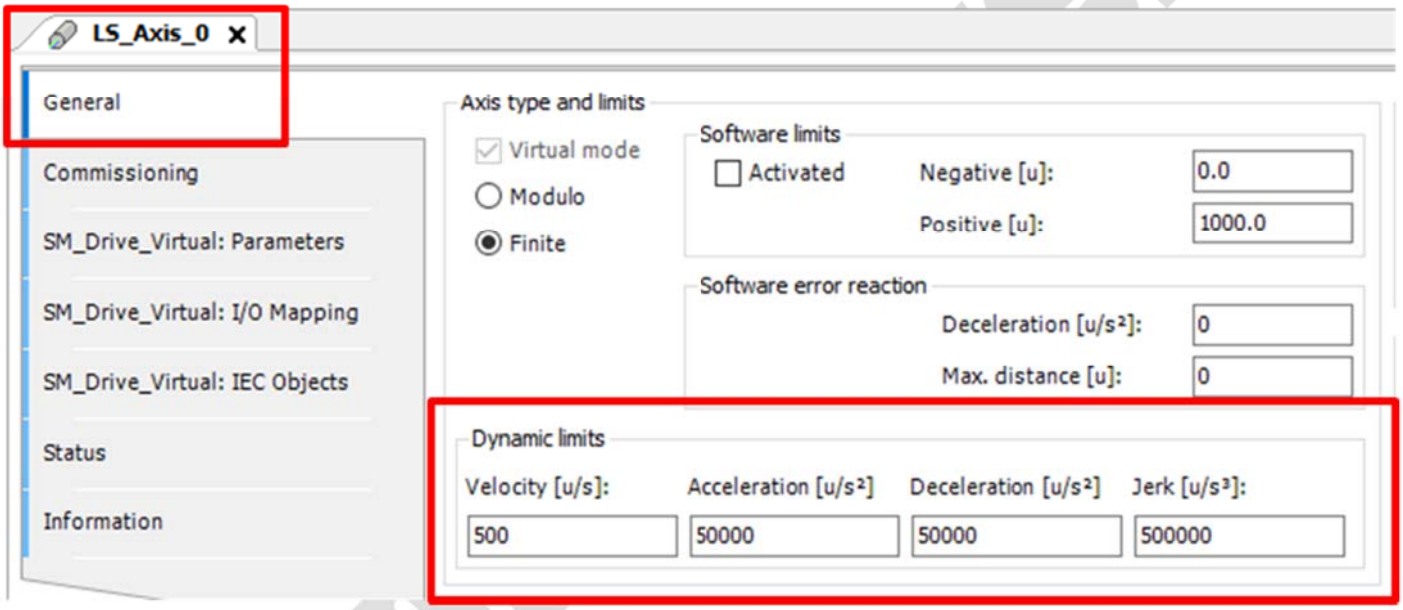


После настройки режимов работы оси необходимо и задания расстояния в пользовательских единицах (Denominator) необходимо определить максимально допустимые величины скорости и ускорения в пользовательских единицах.

Для этого щёлкните дважды левой кнопкой мышки на пункте **LS_Axis_0 (SM_Drive_Virtual** в древе проекта:



В открывшейся вкладке выберите пункт **General** для задания предельных параметров скорости и ускорения:



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:

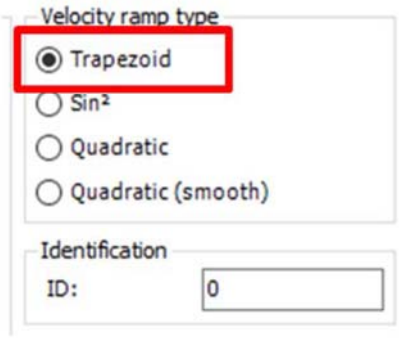
Сначала определяется предельное теоретическое ускорение за 1 мс. Для этого единицы скорости переводятся в мс, т.е. нужно умножить на 1000:

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)

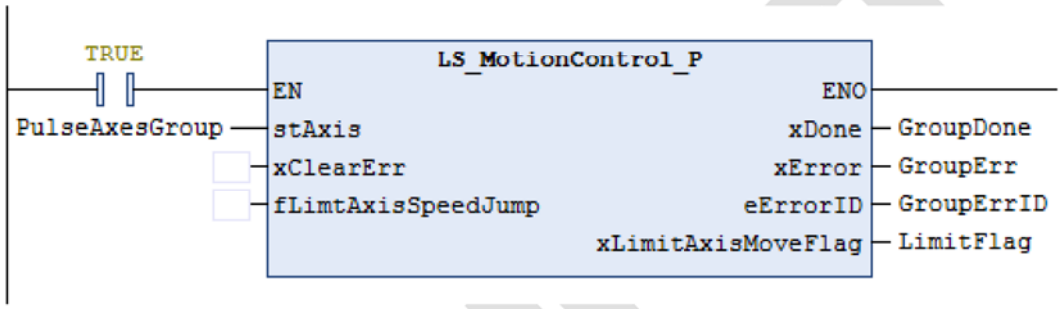
Как правило, максимальное ускорение всегда ограничивают хотя бы 10 мс.
Т.е. надо разделить на 10 и получится значение 50 000, которое и стоит в настройках выше. Параметр Jerk как правило ставят в 10 раз больше, чем ускорение. В нашем примере это будет 500 000.

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

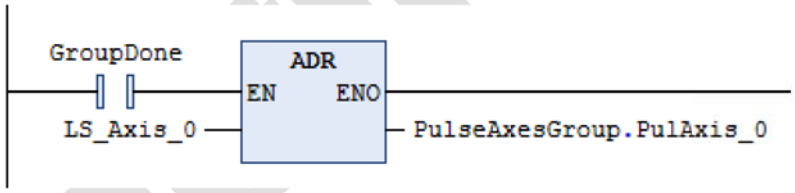


Для запуска оси в программе необходимо создать группу импульсных осей командой **LS_MotionControl_P**.

В примере ниже показано создание группы импульсных осей с названием PulseAxesGroup (название определяет сам пользователь). ***Данная команда должна быть постоянно включена! Привязывать входные переменные не надо!*** Только название группы (как в примере ниже):



Далее необходимо командой **ADR** привязать созданные виртуальные оси к данной группе импульсных осей (PulseAxesGroup):

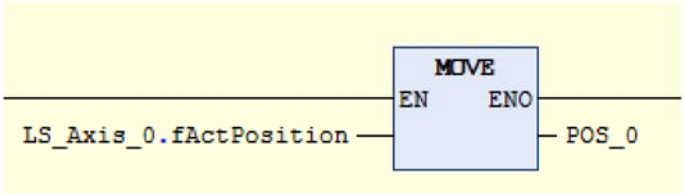


Данная команда также должна быть постоянно включена!

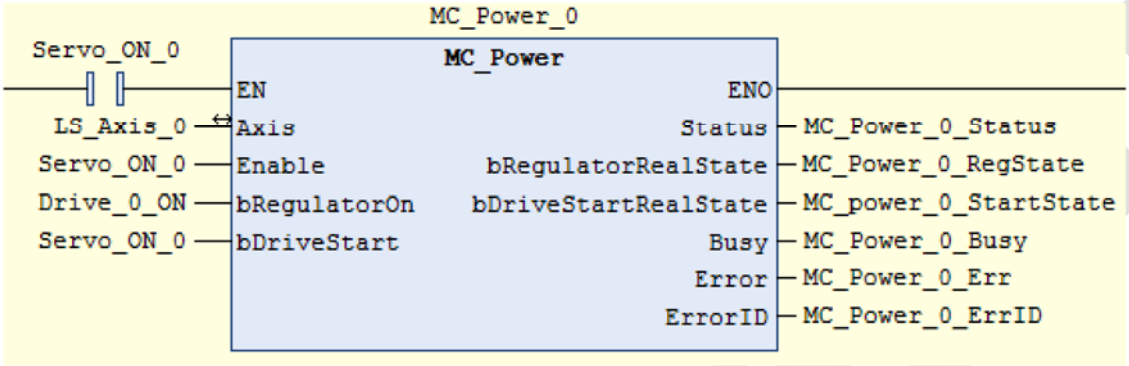
После указанных выше процедур импульсные оси станут доступны во всех инструкциях управления движением как и оси EtherCAT. Процедуры работы такие же. Необходимо учитывать только тот момент, что импульсным приводам необходимо подать физический сигнал Servo-ON и не будет обратной связи. Для обратной связи необходимо завести импульсные выходы привода на счётные входы контроллера. Но этого в большинстве случаев не требуется, так как серво привод имеет свою обратную связь и чётко отрабатывает задание позиции.

Количество выданных импульсов осью можно отслеживать через элемент структуры оси ***.fActPosition.**

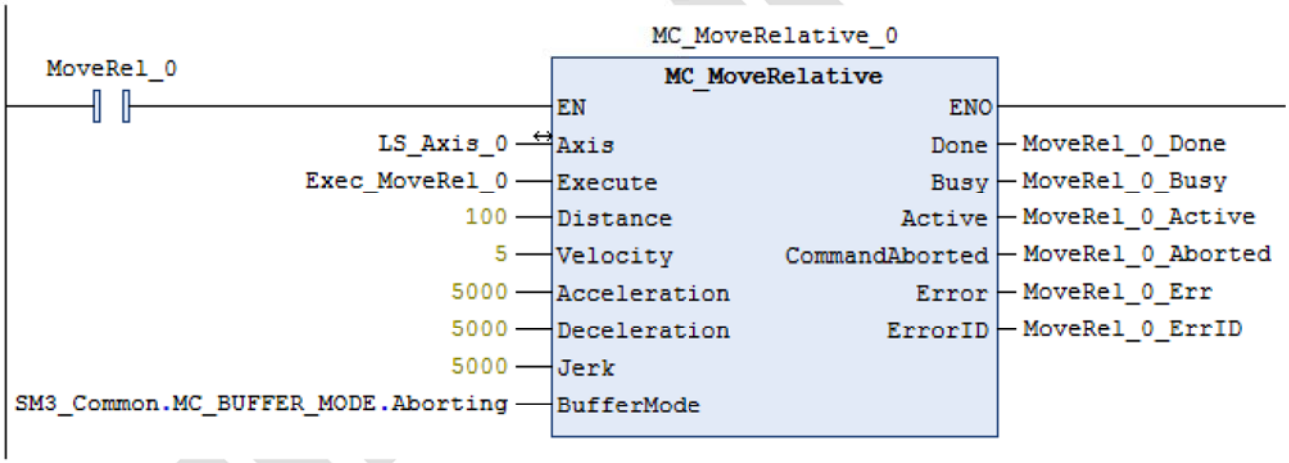
Например для оси с именем *LS_Axis_0* обращение к данному элементу в программе будет выглядеть так: LS_Axis_0.fActPosition



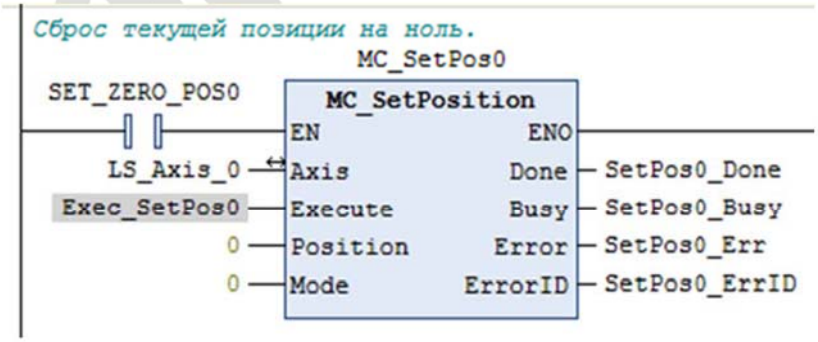
Импульсная ось также должна быть сначала запущена командой MC_Power, а далее используются те же команды, что и EtherCAT оси.



Команда относительного перемещения:



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

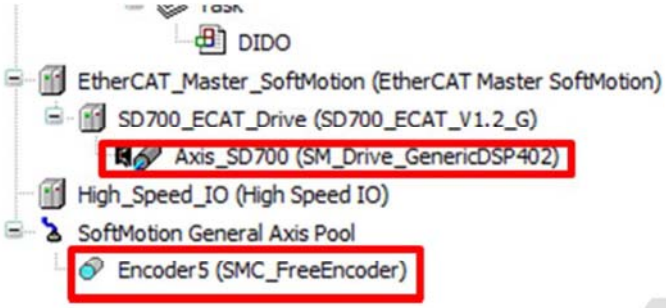


Применение штурвального энкодера в качестве мастер-оси

В данном параграфе рассматривается порядок действий для организации «ручного задатчика». В качестве Мастер-оси выступает штурвальный энкодер, а в качестве ведомой оси сервопривод Veichi SD700, управляемый по шине EtherCAT.

Штурвальный энкодер имеет фазы A и B, которые подключаются к любой высокоскоростной паре входов на ЦПУ. Процедура организации высокоскоростных счётчиков и привязки энкодерных осей рассмотрены в параграфе «Работа с высокоскоростными счётчикам», а создание оси сервопривода рассмотрено в параграфе «Добавление в проект сервопривода».

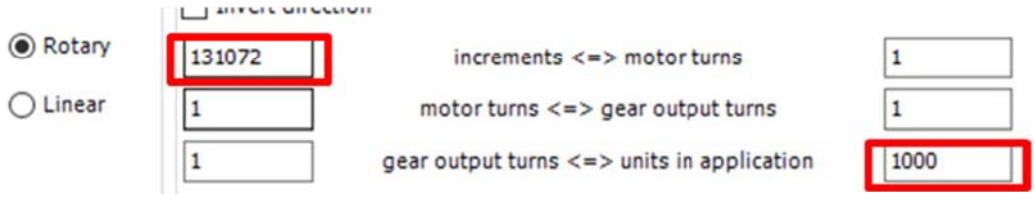
Для данного примера используется счётчик № 5 (нумерация 0 – 5), к которому привязана энкодерная ось **Encoder5**, и ось сервопривода **Axis_SD700**:



Как правило, штурвальные энкодеры имеют 100 импульсов на оборот. Для корректного подсчёта импульсов счётчик должен быть настроен на режим 4AB, т.е. будет приниматься 400 импульсов на 1 оборот энкодера.

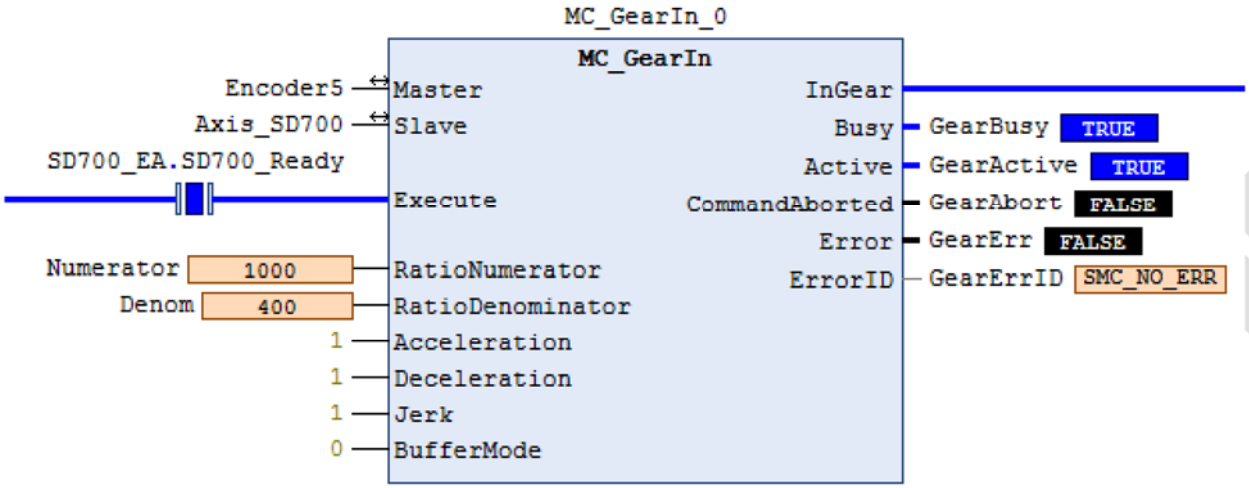
HS counter 6		
Counter_Channel	DINT	5
Counter_PresetInputNum	DINT	0
Counter_SetWorkMode	UDINT	0
Counter_Dir	UDINT	0
Counter_CountMode	UDINT	0
Counter_MaxValue	DINT	2147483647
Counter_MinValue	DINT	-2147483648

В сервоприводе параметры, отвечающие за числитель и знаменатель коэффициента редукции, должны быть установлены в 1 (это параметры по умолчанию в SD700). В настройках оси надо задать количество импульсов энкодера на 1 оборот двигателя (в SD700 энкодер 17 бит, следовательно это число 131072), а также задать линейное перемещение на 1 оборот вала мотора. В нашем примере это 1000 ед. пользователя (это могут быть любые единицы длины). Остальные параметры должны быть установлены в 1.



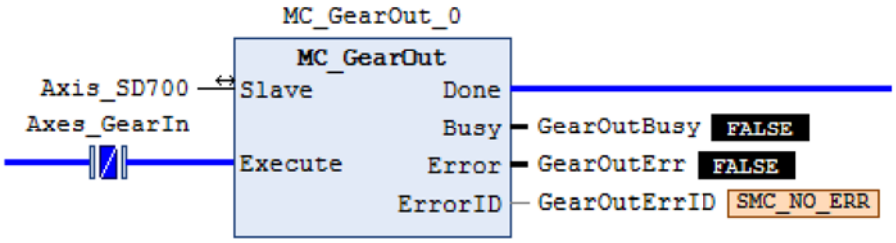
Перед началом работы на природ необходимо подать команду Servo-ON (блок **MC_Power**).

Связка Мастер ось – Ведомая ось устанавливается через команду **MC_GearIn**:



В нашем примере Мастер ось (ножка Master) – **Encoder5**, Ведомая ось (ножка Slave) – **Axis_SD700**.
Числитель (Numerator) равен 1000 ед. Это из настроек оси сервопривода, расстояние в единицах пользователя.
Знаменатель (Denominator) равен 400 ед. Это количество импульсов, которое должен принять счётчик на 1 оборот энкодера. У энкодера 100 импульсов на оборот, но так как включен режим 4AB, то счётчик будет считать все передние и задние фронты, т.е. получится 400 импульсов (если у энкодера другое количество импульсов на оборот, то нужно указать количество какое есть фактически).

Расцепление осей осуществляется командой **MC_GearOut**:

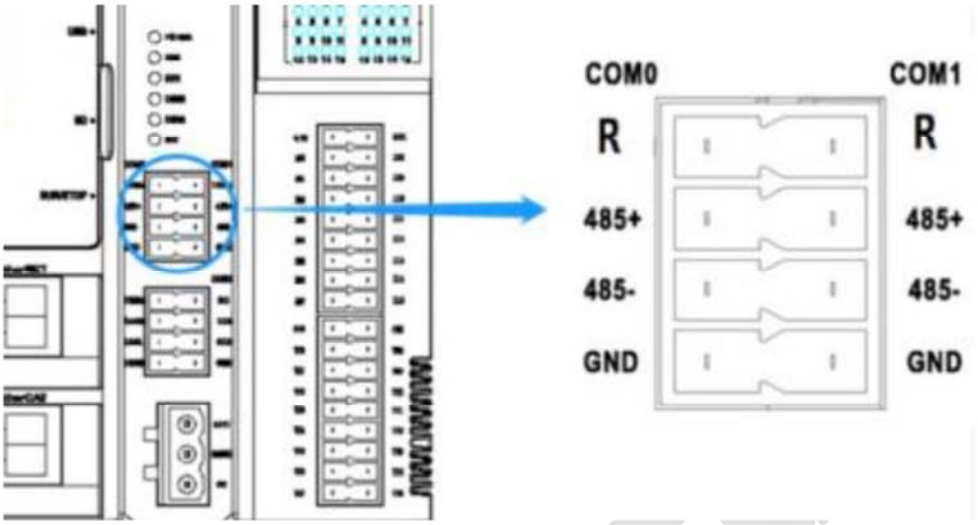


Последовательная связь по протоколу Modbus RTU Master

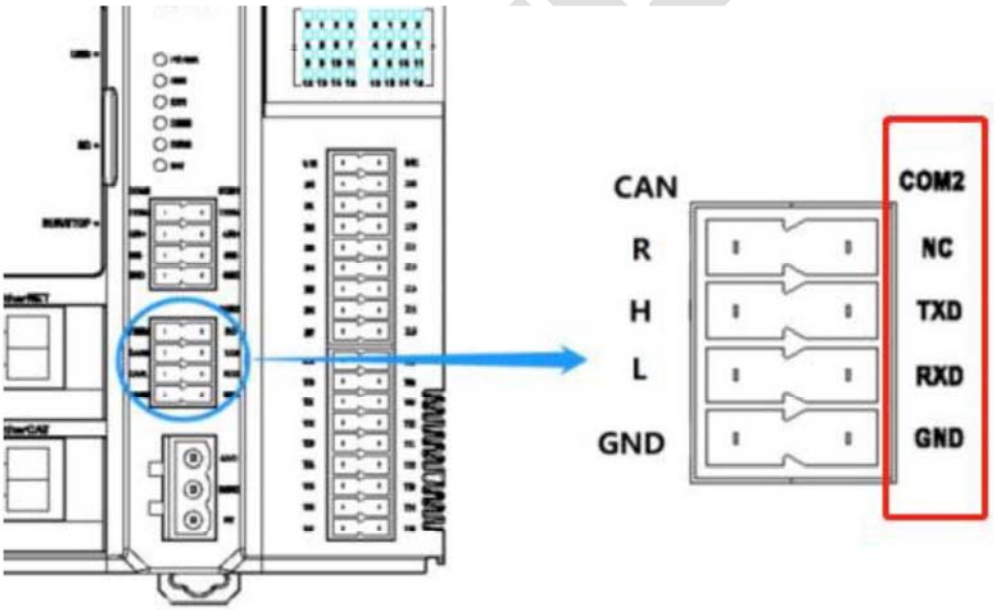
Контроллеры серии MX300 имеют на борту 1 порт RS232 и 2 порта RS485. Все три порта могут работать как в режиме Мастера, так и Ведомого. В данной главе рассмотрена работа в режиме Мастера по протоколу Modbus RTU.

Физически порты имеют на контроллере следующее расположение:

RS485:

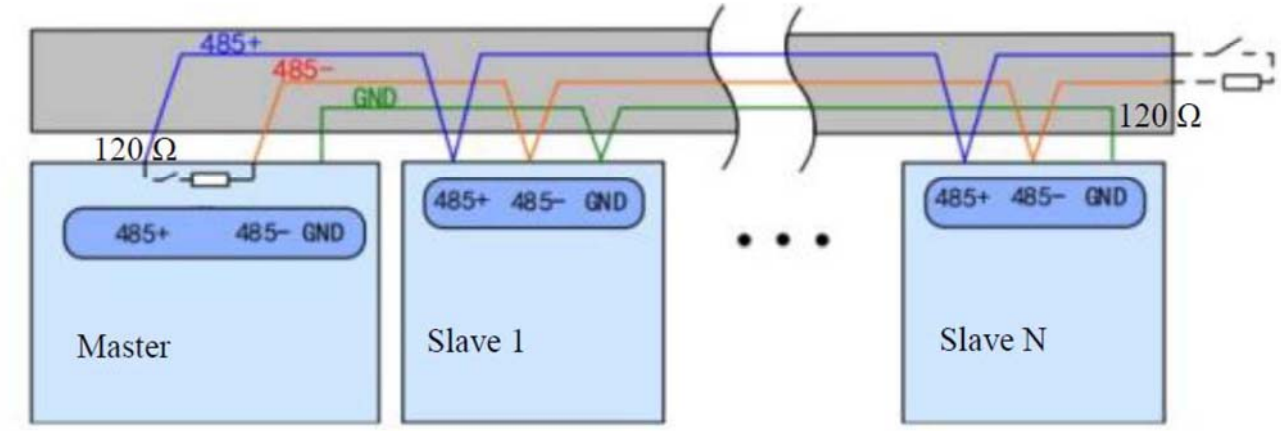


RS232:



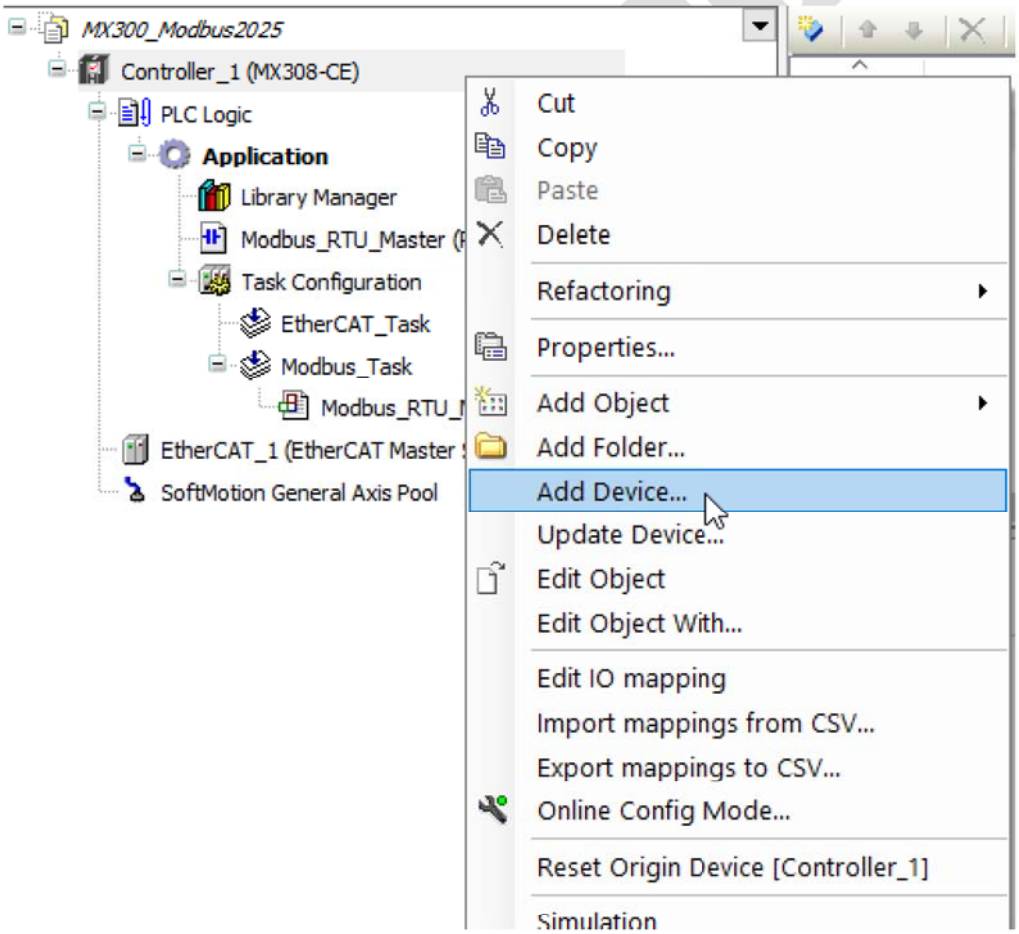
По RS232 возможно соединение только точка-точка. Максимальная длина кабеля на скорости 9600 максимум 15 метров. На скорости 115200 – 3 метра.

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.

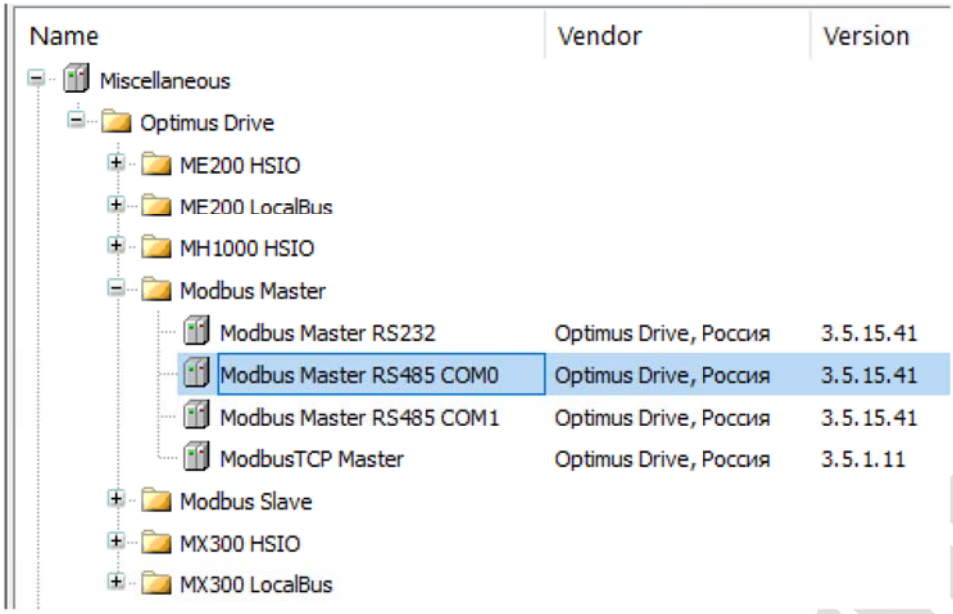


У контроллеров MX300 имеется встроенный резистор 120 Ом, который подключается путём установки перемычки между клеммами 485+ и R. Таким образом, мы соединяем клемму RS485+ через резистор 120 Ом на клемму RS485-.

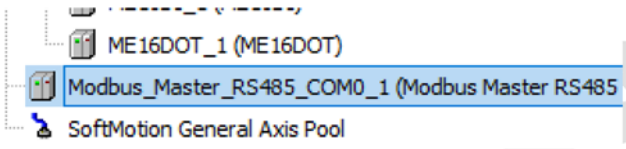
Для организации связи по Modbus RTU в проект необходимо добавить адаптер соответствующего порта. Щёлкните правой кнопкой мышки на вкладке Device и в открывшемся меню выберите пункт **Add Device**:



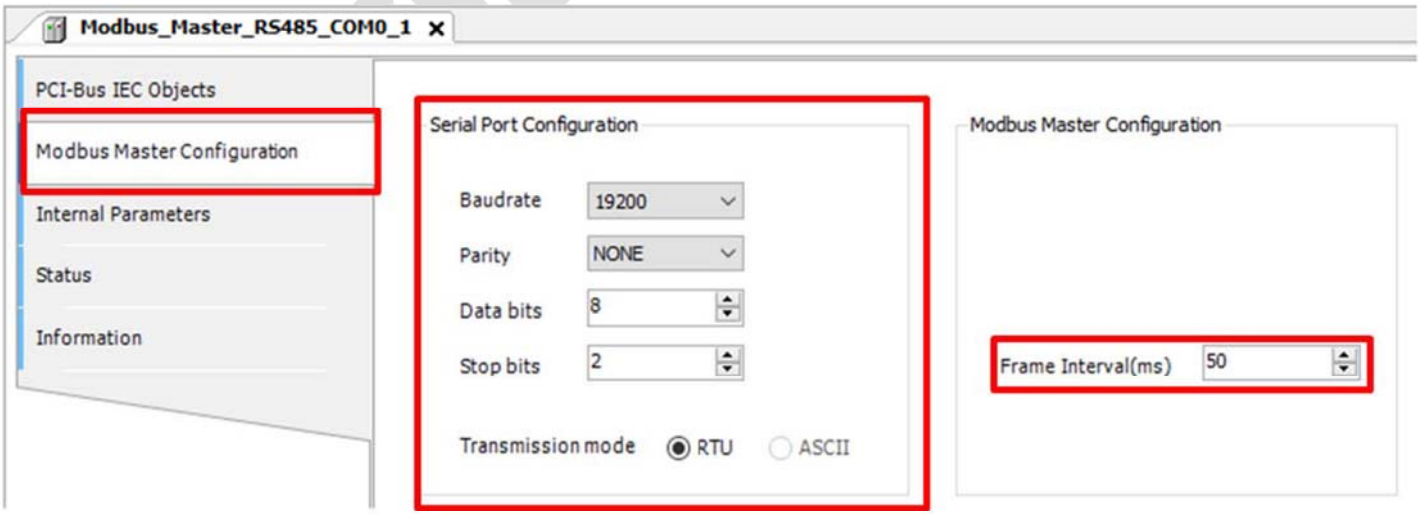
В открывшемся окне выберите папку **Miscellaneous – Optimus Drive – Modbus Master – Modbus Master RS485 COM0**:



В древе проекта появится пункт **Modbus_Master_RS485_COM0_1**:

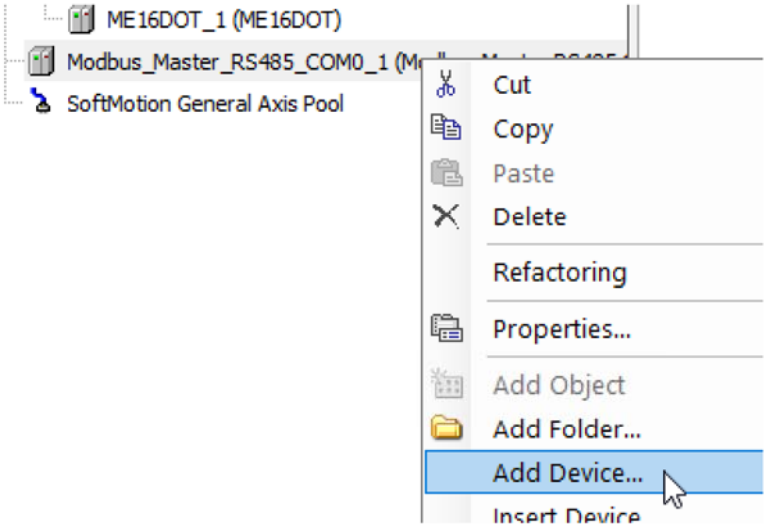


Щёлкните по этому пункту дважды левой кнопкой мышки и в открывшейся вкладке сделайте настройки связи в разделе **Modbus Master Configuration** для порта RS485_COM0:

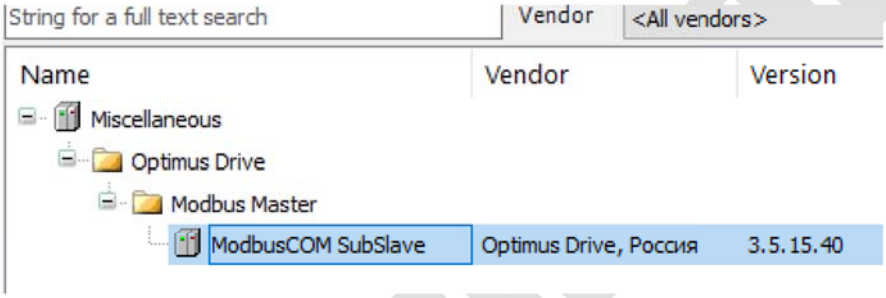


Поддерживается только режим Modbus RTU. В нашем примере выставлена скорость 19200, протокол 8, N, 2. Пункт справа **Frame Interval (ms)** определяет темп отправки пакетов. В нашем примере стоит 50 мс.

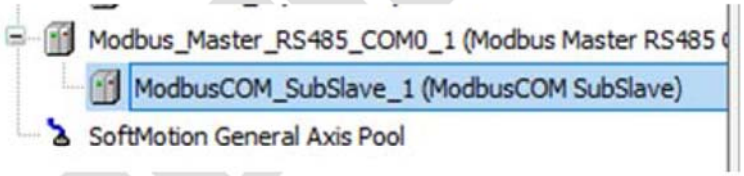
Далее щёлкните на пункте Modbus_Master_RS485_COM0_1 правой кнопкой мышки и в открывшемся окне выберите пункт **Add Device** и добавьте Ведомое устройство **Modbus RTU**:



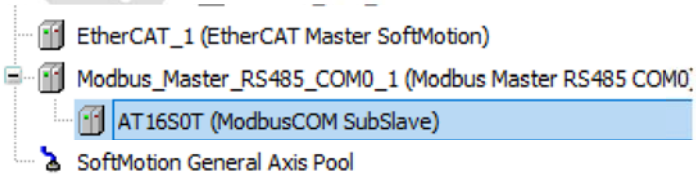
В открывшемся окне выберите пункт **ModbusCOM_SubSlave**:



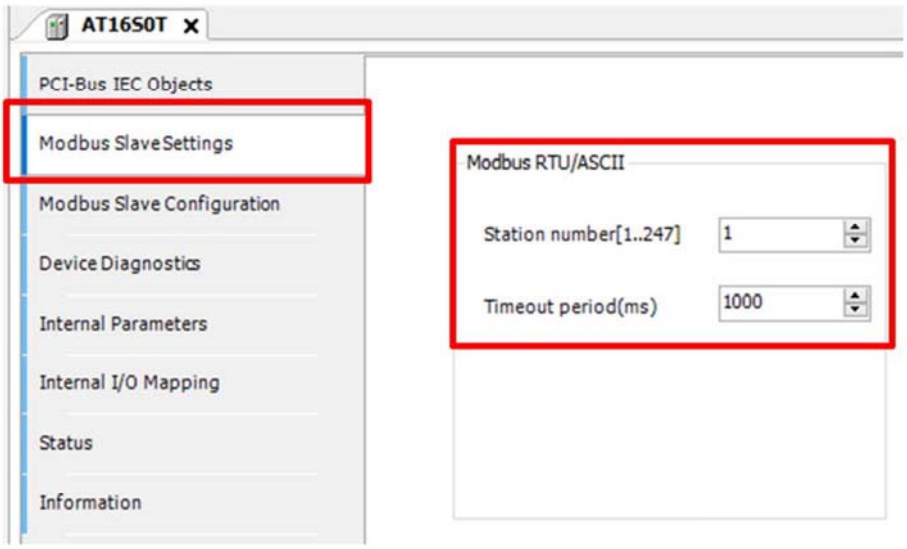
В древе проекта появится пункт **ModbusCOM_SubSlave_1**:



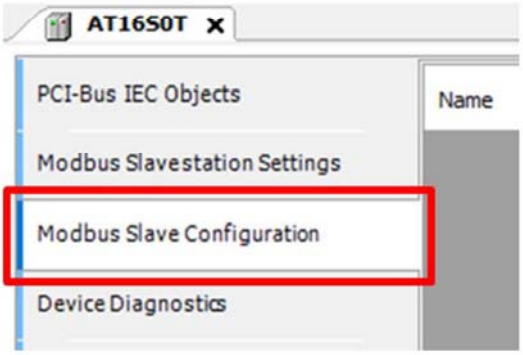
Название Ведомого можно поменять. Например заменить на контроллер AT16S0T:



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

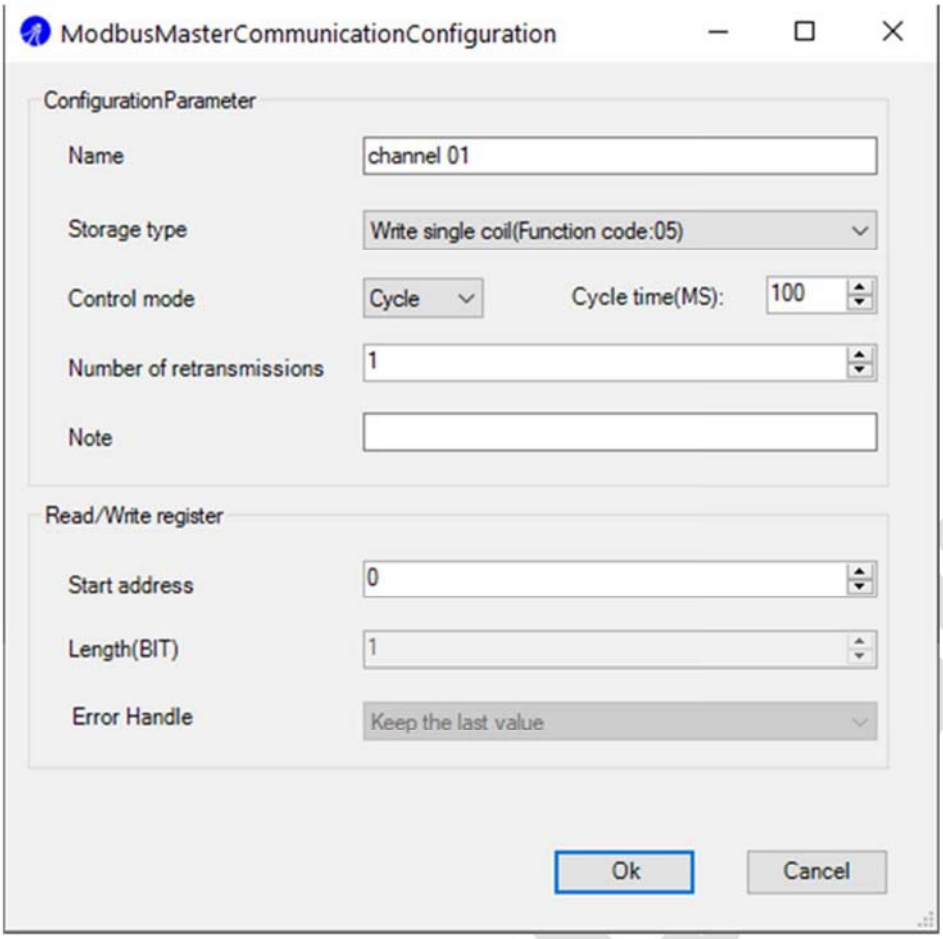


Далее в пункте **Modbus Slave Configuration** необходимо задать регистры Ведомого и их количество для чтения/записи Мастером. Всего может быть создано 31 соединение с Ведомым.



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

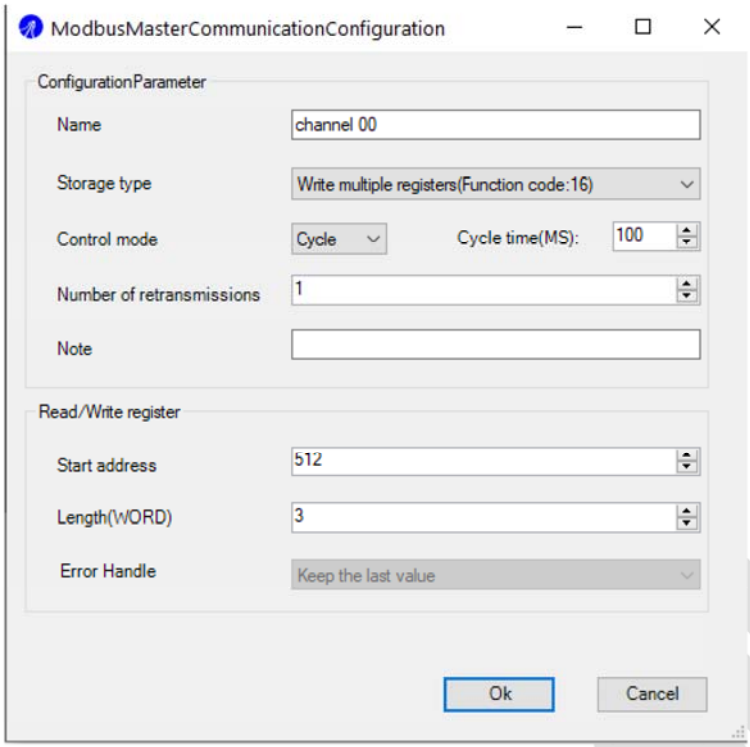
Start Address – Адрес регистра Ведомого, задаётся в HEX записью вида 0x0000

Length – Длина данных для чтения/записи

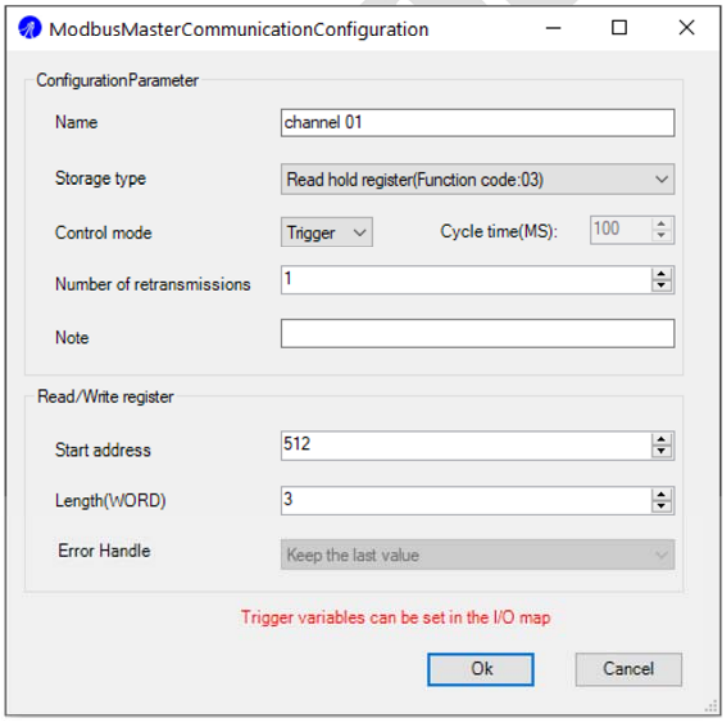
Error Handling – Фиксировано на сохранять состояние регистров при потере связи

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.

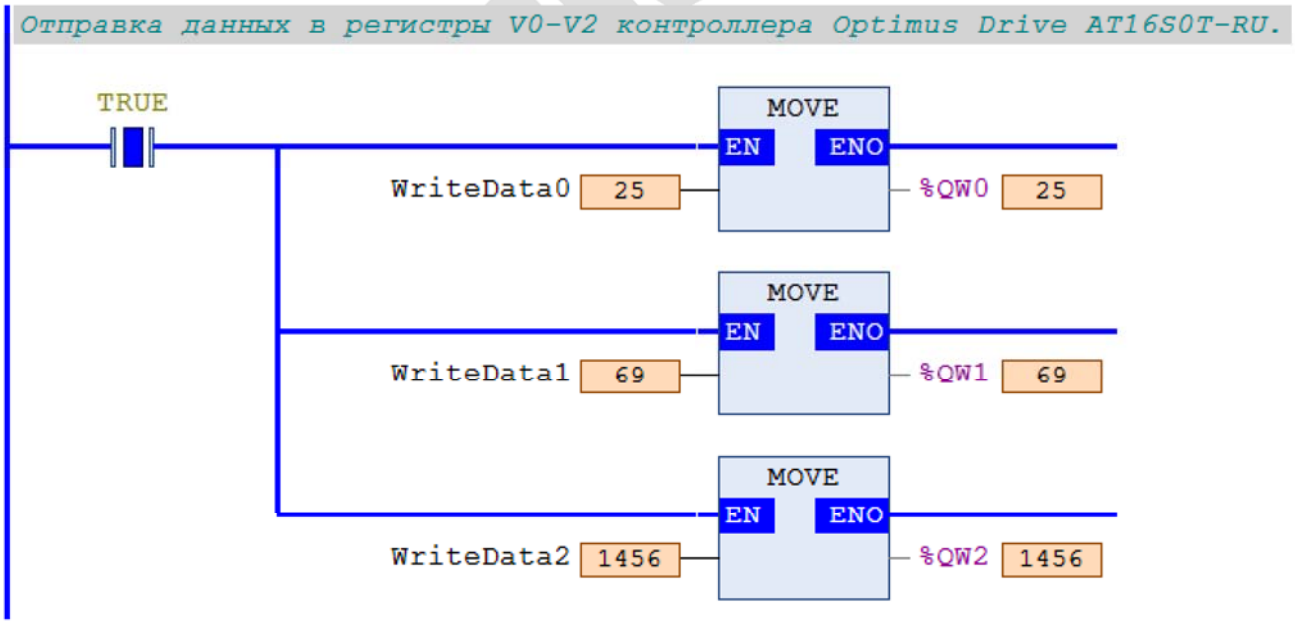


После создания каналов опроса система автоматически выделит под них регистры в соответствии с заявленным количеством регистров в Ведомом устройстве. В нашем примере будет 3 регистра на запись и 3 на чтение. Посмотреть можно в пункте **Internal I/O Mapping**:

Channel	Address	Type	Unit	Description
ErrorCode	%IW0	WORD		ModbusRTU ErrorCode
channel 00	%QW0	ARRAY [0..2] OF WORD		Write multiple registers
channel 00[0]	%QW0	WORD		Write #0512
channel 00[1]	%QW1	WORD		Write #0513
channel 00[2]	%QW2	WORD		Write #0514
channel 00	%IX2.0	BOOL		Processing complete state
channel 01	%IW2	ARRAY [0..2] OF WORD		Read hold register
channel 01[0]	%IW2	WORD		Read #0512
channel 01[1]	%IW3	WORD		Read #0513
channel 01[2]	%IW4	WORD		Read #0514
channel 01	%IX10.0	BOOL		Processing complete state
channel 01	%QX6.0	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.

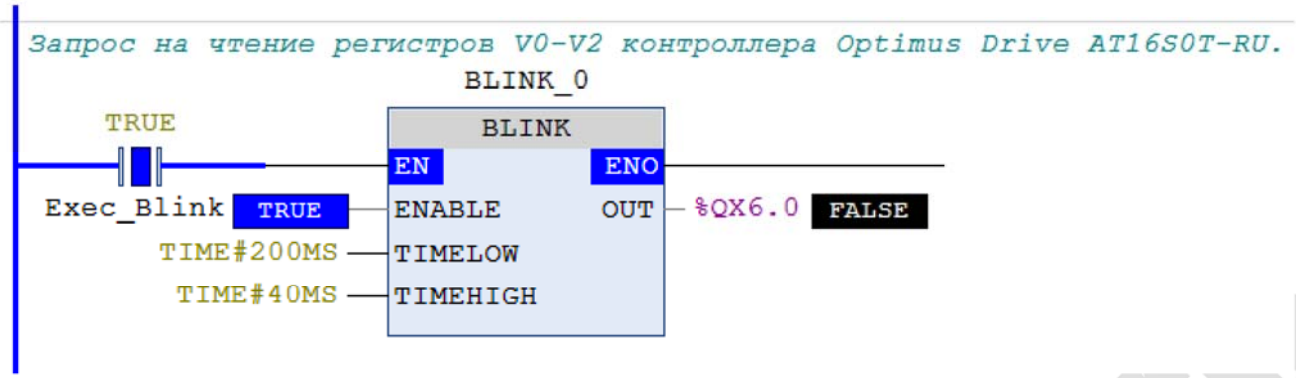
Запись данных в Ведомое устройство (Write). Запросы отправляются циклически без дополнительных условий.



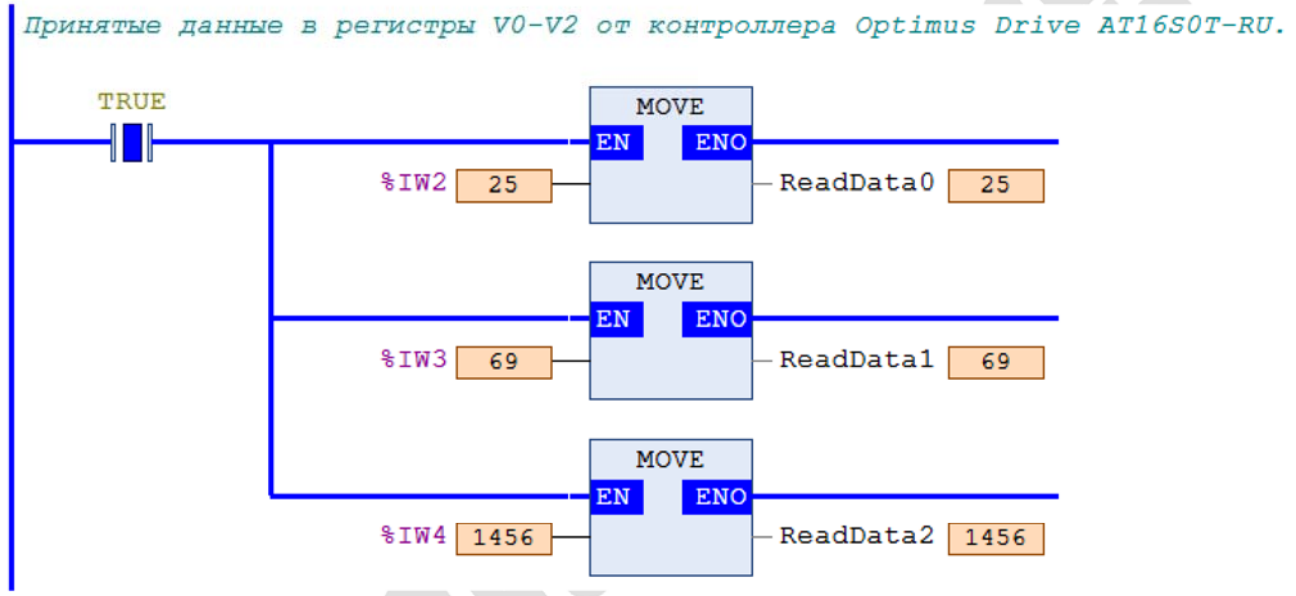
Приём данных от Ведомого устройства (Read). Запросы отправляются по тригеру:

channel 01	%QX6.0	BOOL		Trigger control bit

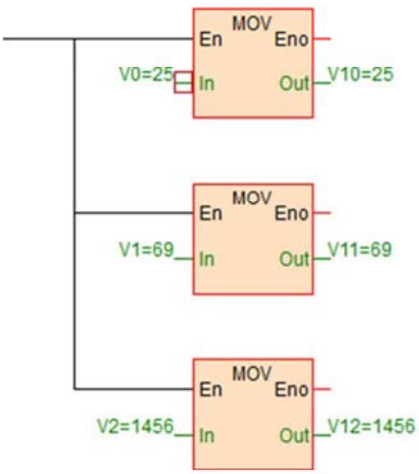
Триггер можно организовать при помощи команды Blink:



Принятые данные от Ведомого:



Мониторинг программы контроллера AT16S0T-RU:



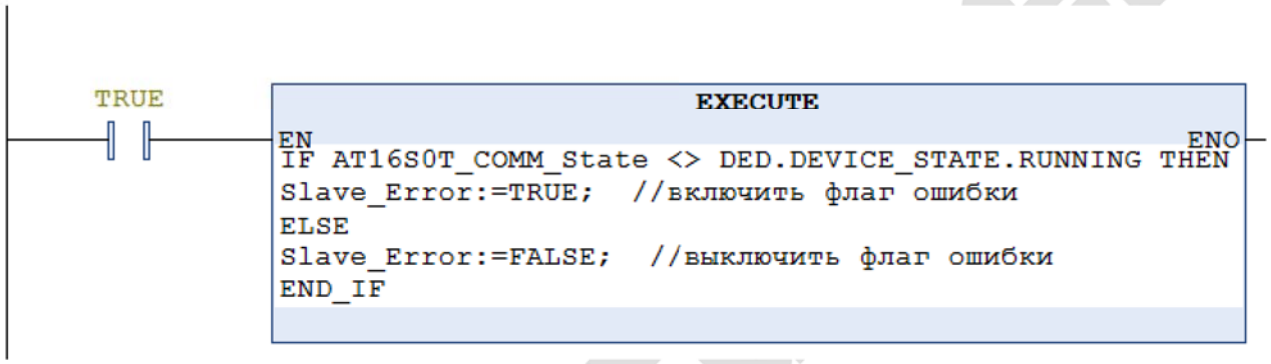
Ошибки связи можно проверять через свойство узла **GetDeviceState** и переменную типа **DED.DEVICE_STATE**

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**:

VAR **Slave_Error** BOOL

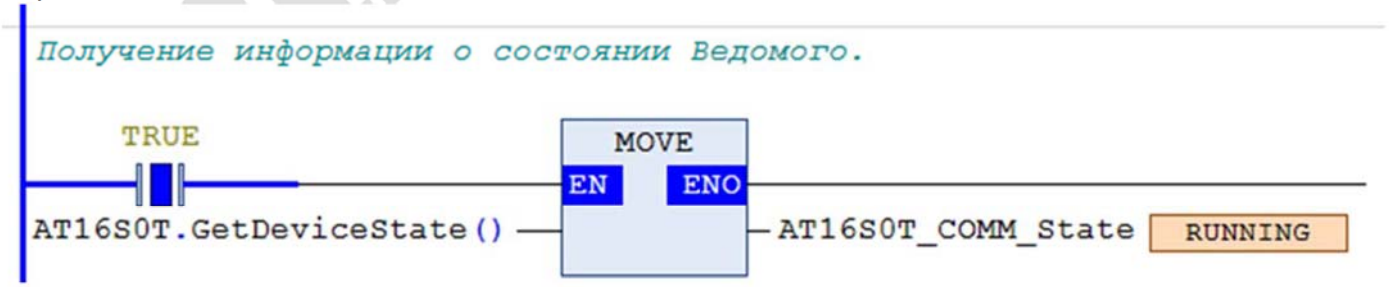


Текст программы из блока Execute:

```
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:

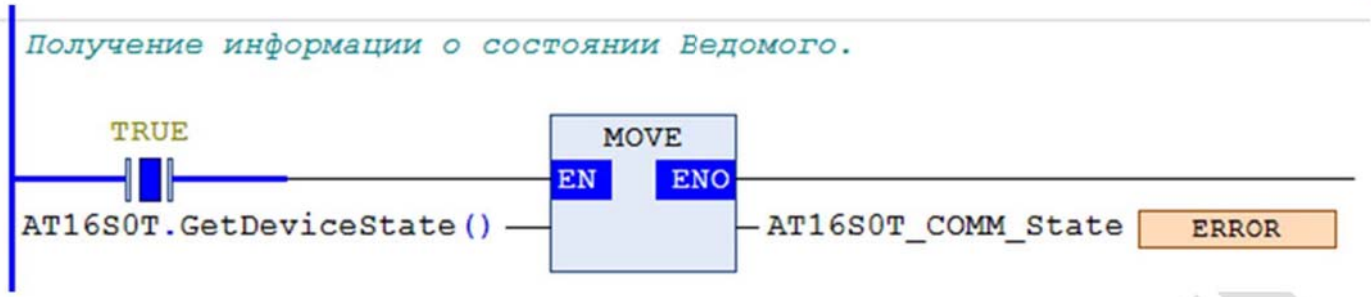
Нормальное состояние:



Watch 1

Expression	Application	Type	Value
Modbus_RTU_Master.AT16S0T_COMM_State	Controller_1.Application	DEVICE_STATE	RUNNING
Modbus_RTU_Master.Slave_Error	Controller_1.Application	BOOL	FALSE

Ошибка связи:



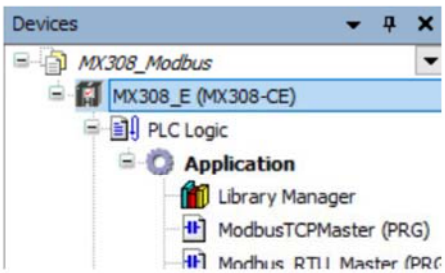
Watch 1

Expression	Application	Type	Value
Modbus_RTU_Master.AT16S0T_COMM_State	Controller_1.Application	DEVICE_STATE	ERROR
Modbus_RTU_Master.Slave_Error	Controller_1.Application	BOOL	TRUE

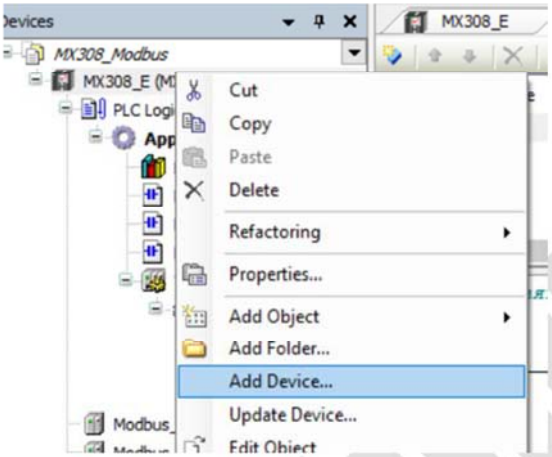
Последовательная связь по протоколу Modbus RTU Slave

Контроллеры серии MX300 имеют на борту 1 порт RS232 и 2 порта RS485. Все три 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.

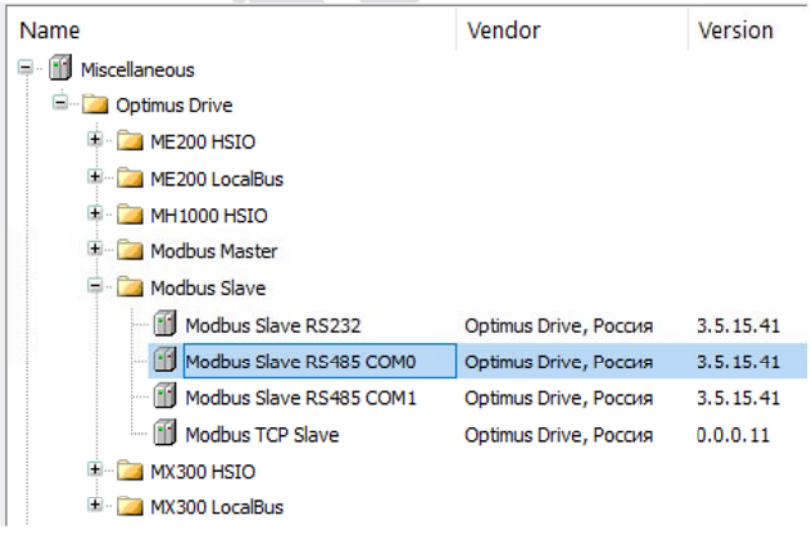
Для перевода порта последовательной связи в режим Modbus RTU Slave необходимо подключить к проекту адаптер связи типа **Modbus_Slave_***. Для этого встаньте в древе проекта на пункт **Device** и нажмите правую кнопку мышки.



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



В открывшемся окне выберите пункт **Miscellaneous – Optimus Drive – Modbus Slave** и адаптер нужного порта, в нашем примере COM0 RS485:

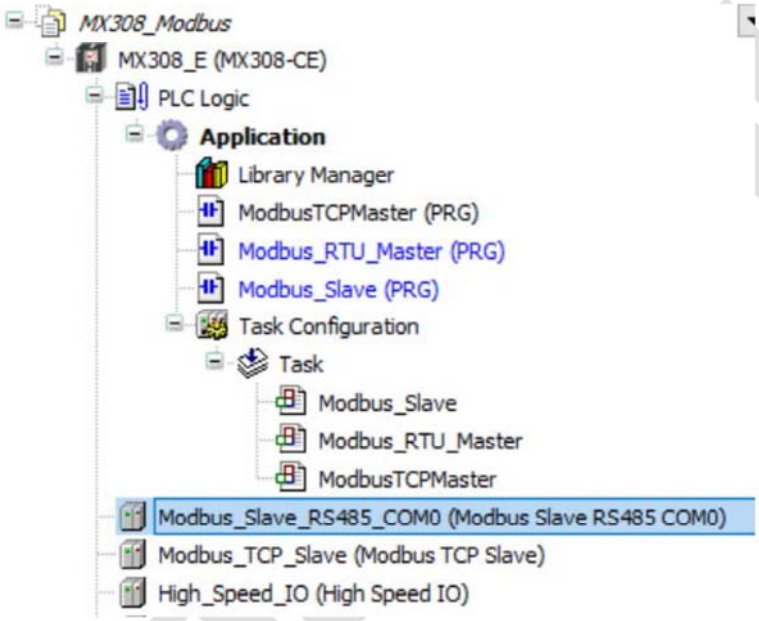


После добавления адаптера в проект и его загрузки, контроллер откроет доступ к своей памяти по протоколу Modbus RTU через соответствующий порт. Адресация будет одинаковая для всех портов. Согласно спецификации протокола Modbus, он имеет возможность читать булевы и словные регистры без обозначения типов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

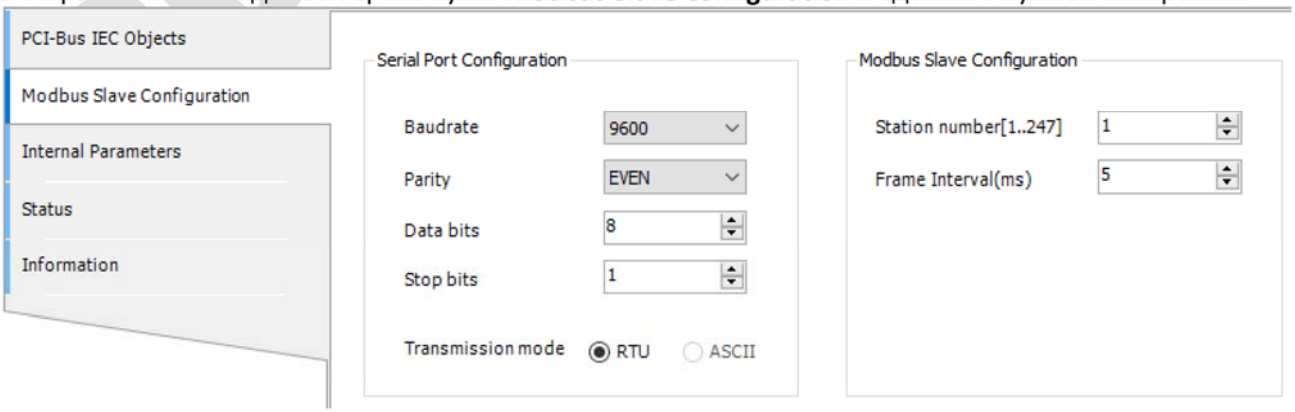
Тип	Диапазон	Функция	Инициализация	Количество
Q (выходы)	(QX0.0 ~ QX8191.7)	0x01,0x05,0x0f	0	65536
I (входы)	(IX0.0 ~ IX8191.7)	0x02	0	65536
M (данные)	%MW0~%MW65535	0x03,0x06,0x10	0	65536

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:



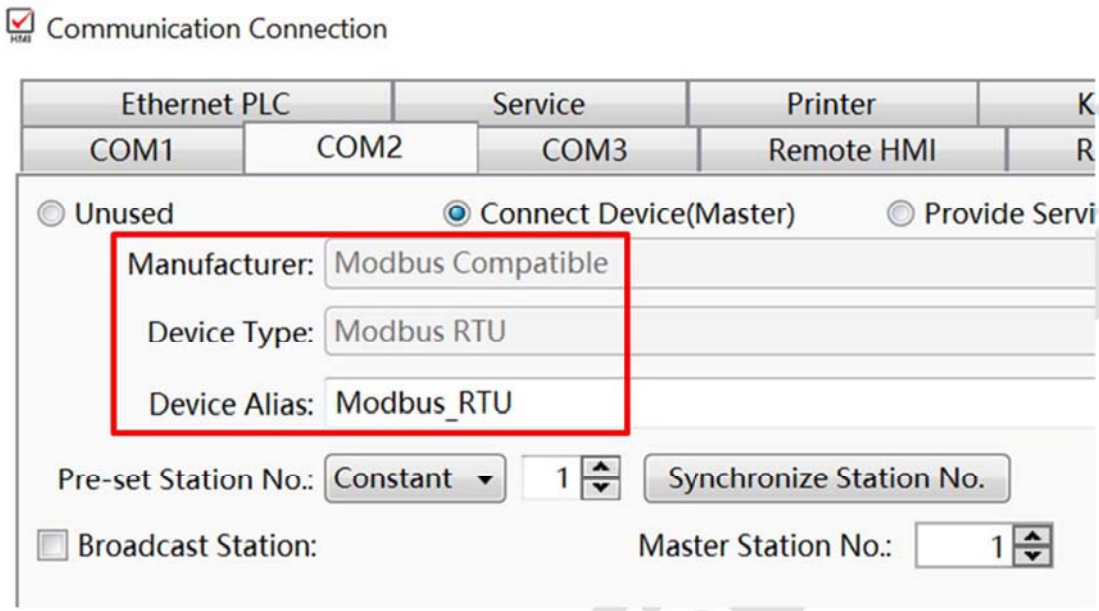
В открывшейся вкладке выберите пункт **Modbus Slave Configuration** и сделайте нужные настройки:



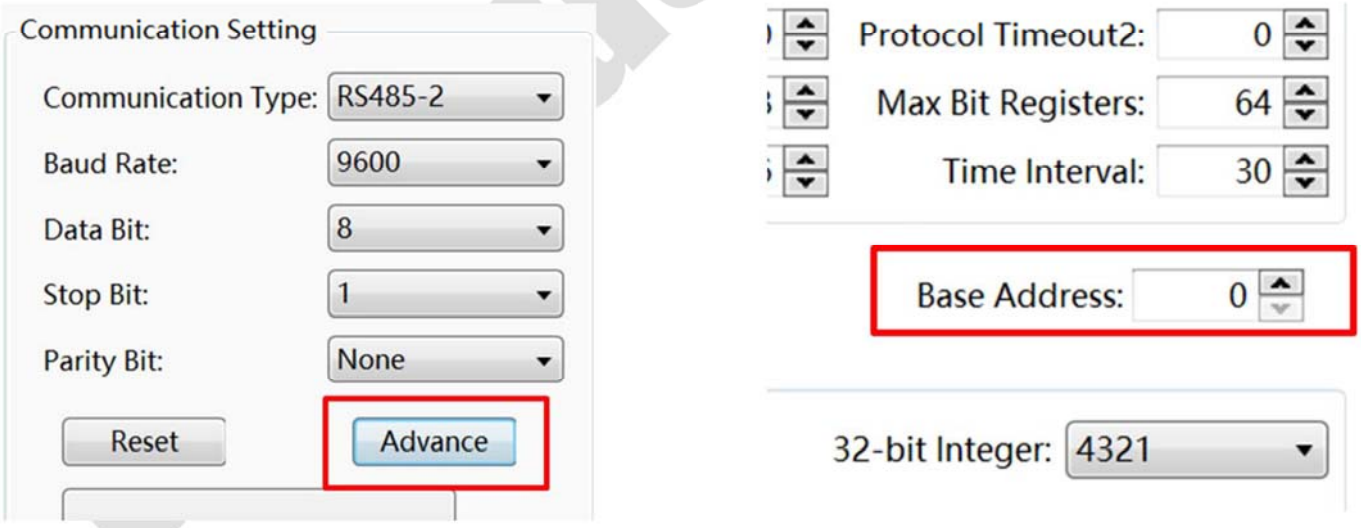
После загрузки проекта в контроллер, он будет в состоянии отвечать на Modbus RTU запросы с заданной скоростью и форматом данных (9600, 8,N,1 в нашем примере).
Режим Modbus ASCII контроллерами серии MX300 не поддерживается.

Контроллеры серии MX300 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:

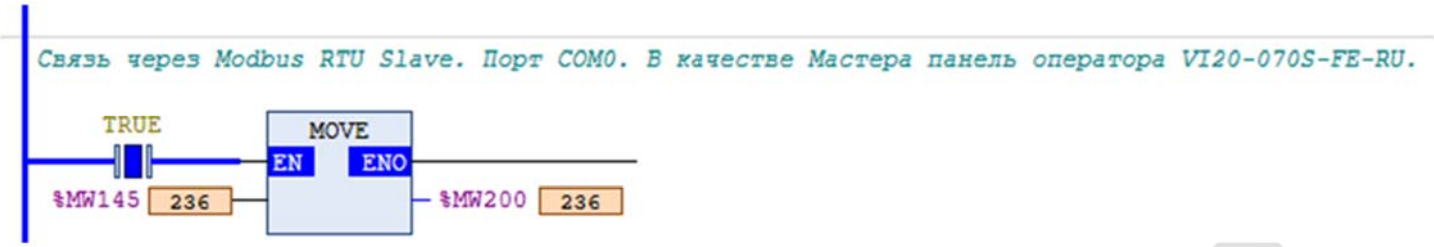


Задать протокол и во вкладке **Advance** убрать смещение адреса:



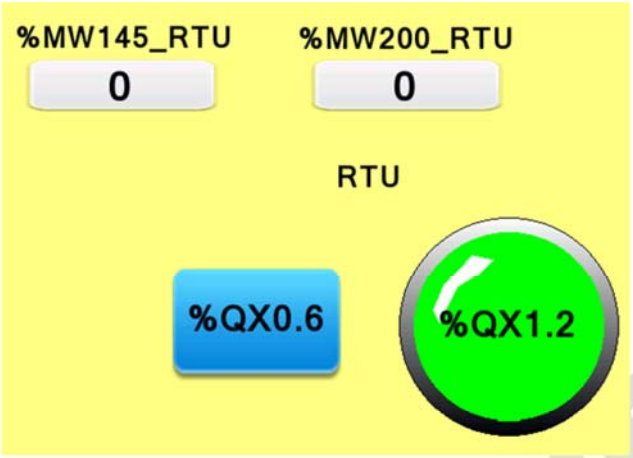
В программе контроллера создана простая программа для отображения данных, в которой задействованы следующие регистры:

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

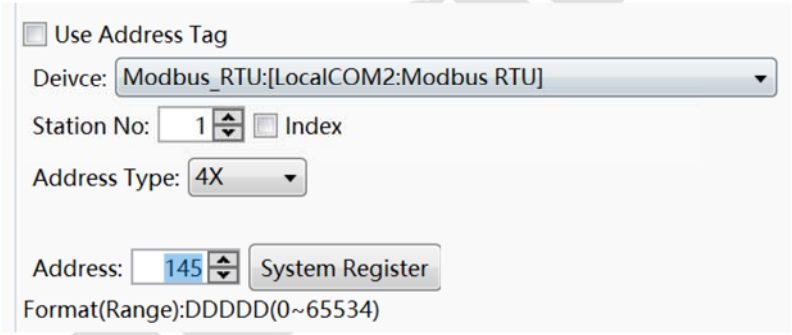


Регистр %MW145 - источник данных, регистр %MW200 - приёмник данных.

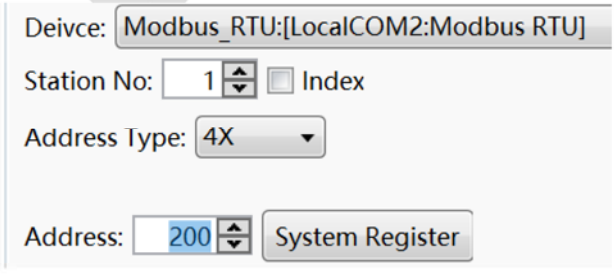
В панели оператора нарисован простой экран с четырьмя объектами соответственно: Input, Display, Button и Indicator.



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

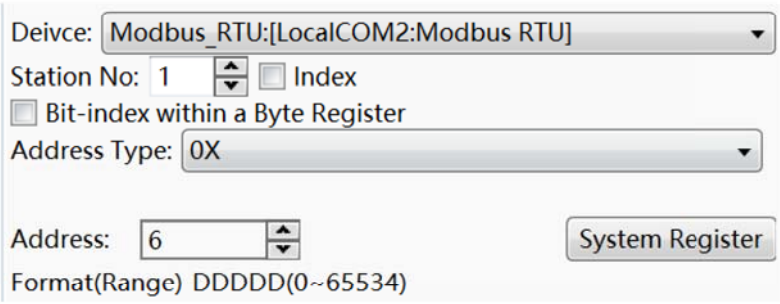


а к регистру %MW200 так

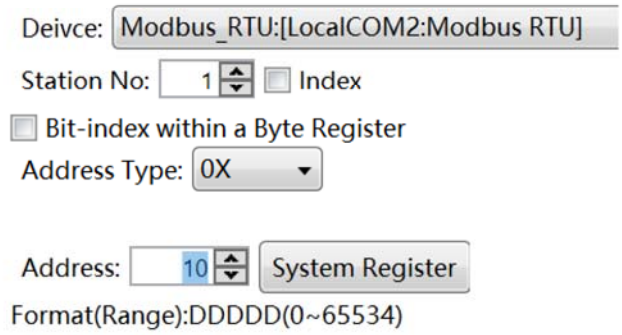


Таким образом, нужно просто указать номер регистра %MW****.

Обращение к булеву регистру %QX0.6 выглядит так:



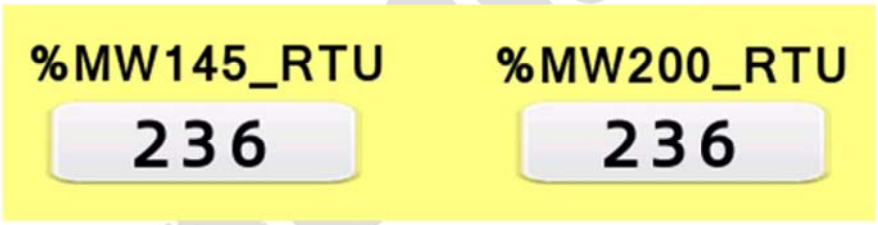
а к регистру %QX1.2 так:

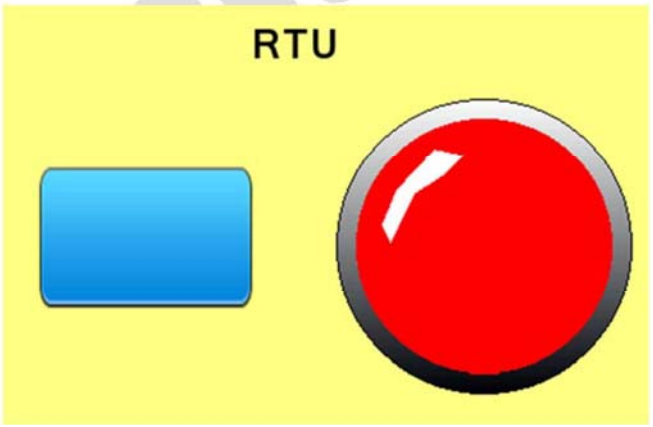


Т.е. идёт сплошная десятичная адресация. Каждый последующий регистр увеличивает адрес на 1.

%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:

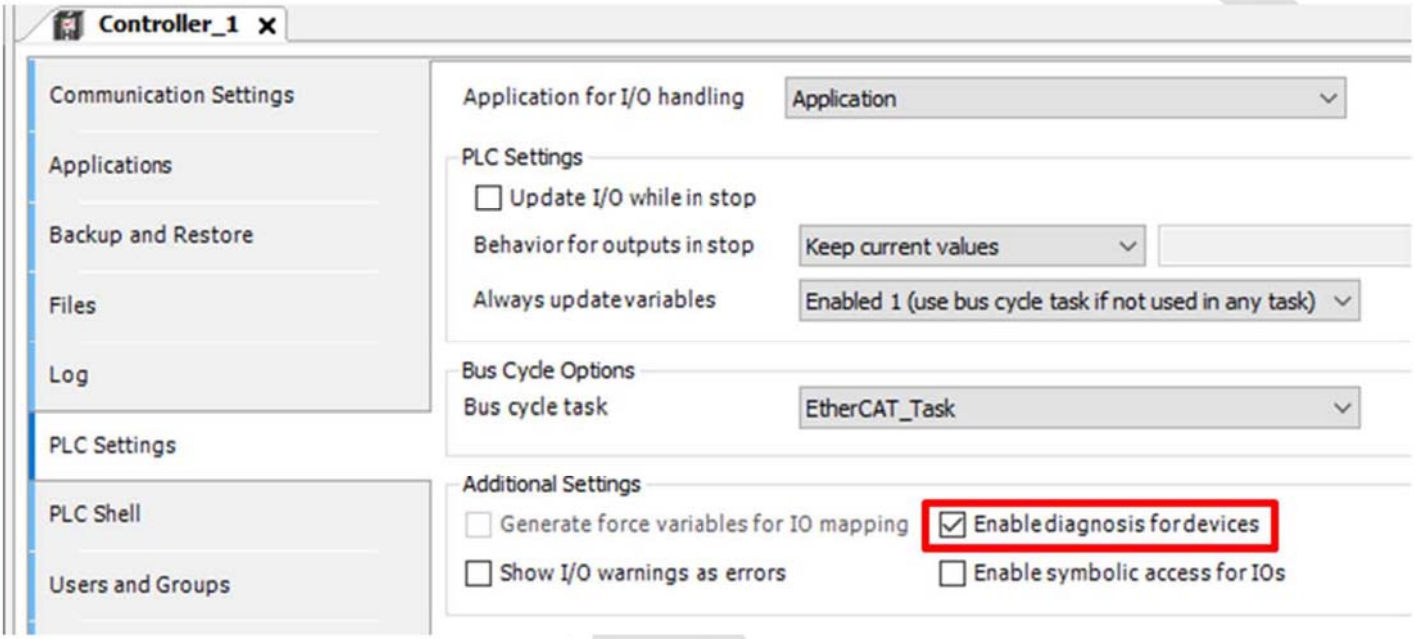




Связь по протоколу Modbus TCP Master

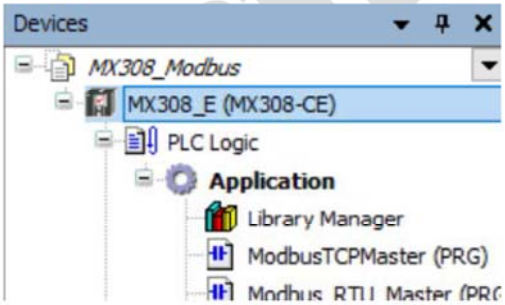
Контроллеры серии MX300 могут работать как в режиме Modbus TCP Client (Master), так и в режиме Modbus TCP Server (Slave), причём оба режима могут использоваться одновременно. В данной главе рассматривается организация связи контроллера с Ведомыми устройствами по протоколу Modbus TCP Client (Master).

При использовании Modbus TCP Client (Master) для удобства оценки состояния ведомых узлов рекомендуется во вкладке **Device** пункте **PLC Settings** установить флажок **Enable diagnosis for devices**:

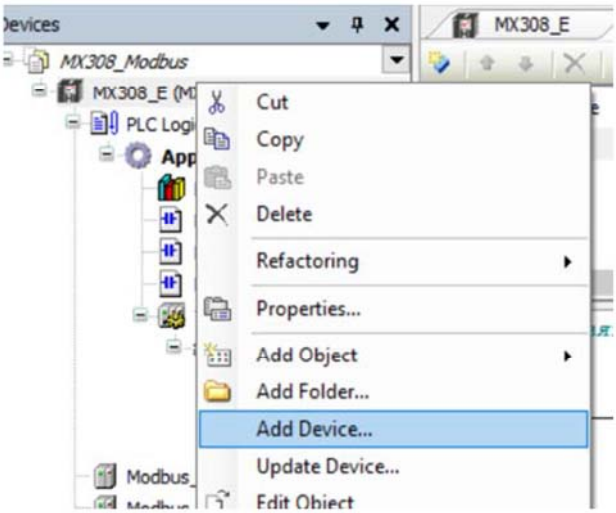


После установки флажка к проекту автоматически будет подключена библиотека **DED (CAA Device Diagnosis)**:

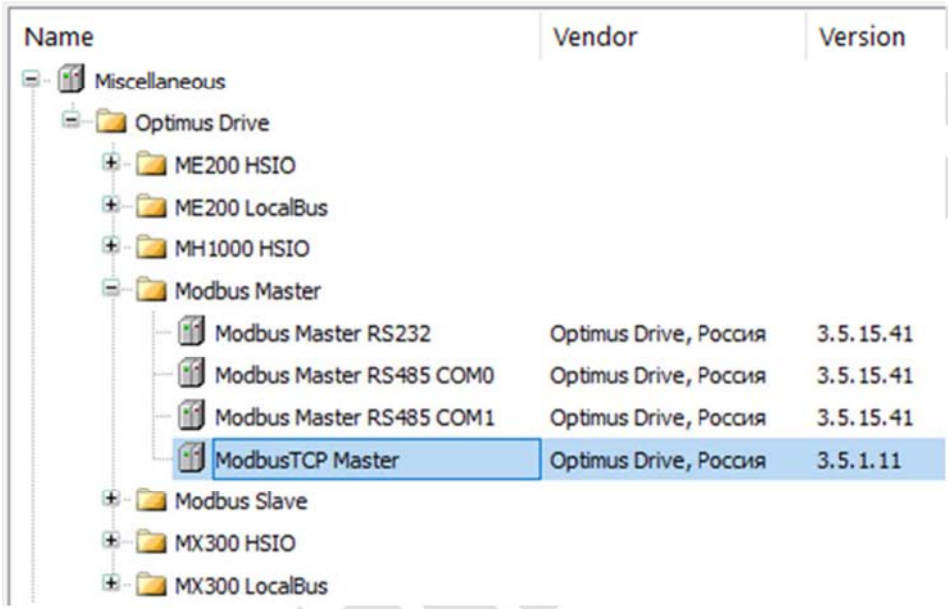
Для начала работы по протоколу Modbus TCP Client (Master) в проект необходимо добавить адаптер **Modbus TCP Master**. Для этого встаньте на пункт **Device** и щёлкните правой кнопкой мышки:



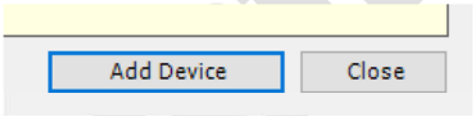
В появившемся меню выберите пункт **Add Device**:



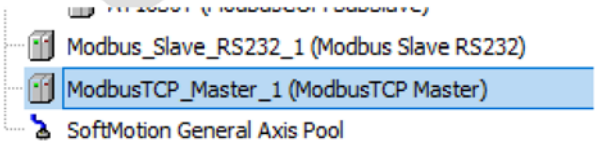
В открывшемся окне выберите пункт **Miscellaneous – Optimus Drive – Modbus Master – Modbus TCP Master**:



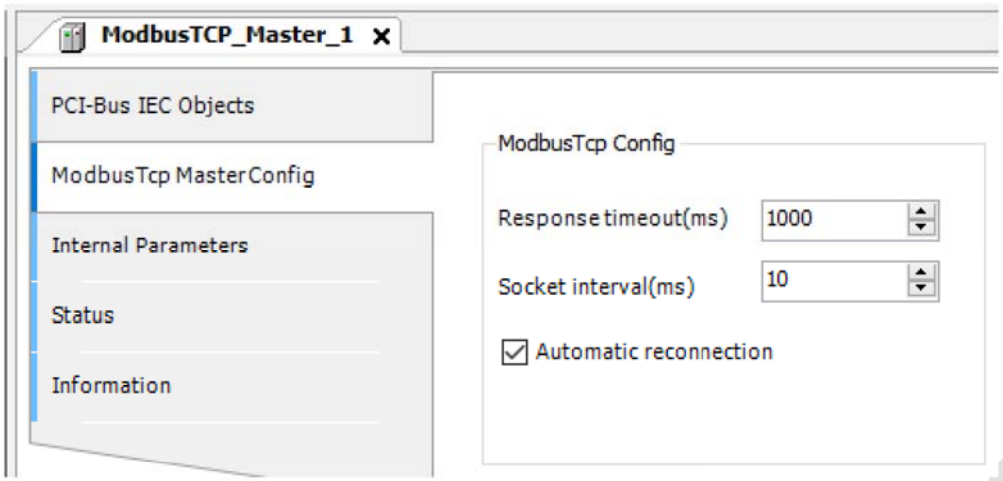
Нажмите кнопку **Add Device**:



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



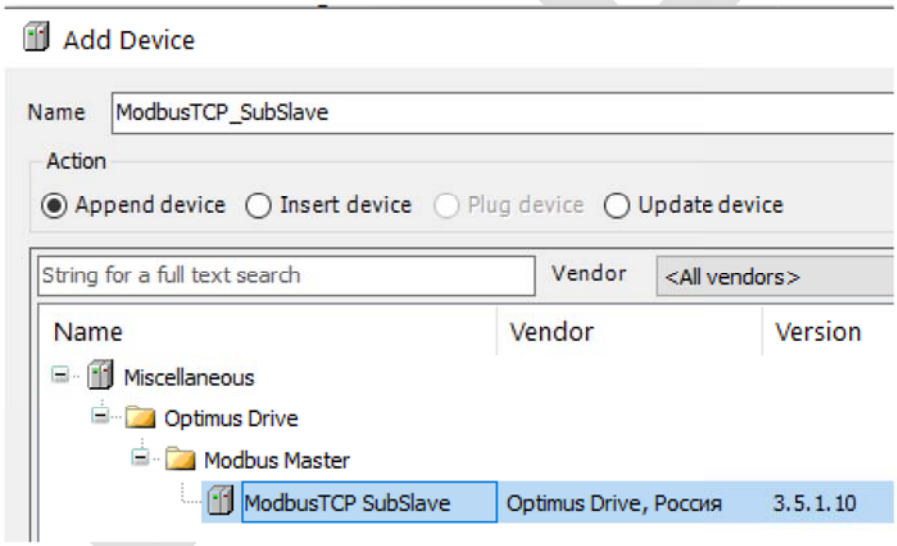
Щёлкните дважды левой кнопкой мышки на пункте **Modbus TCP Master**, откроется вкладка настроек параметров Мастера:



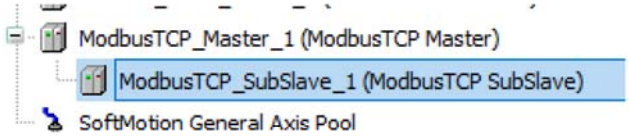
В разделе **Modbus TCP Master Config** необходимо задать:

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

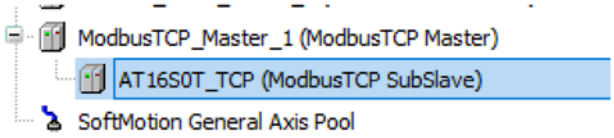
Контроллер готов к работе по протоколу Modbus TCP Client (Master) и осталось добавить линки, т.е. конкретные ведомые устройства, которые контроллер будет опрашивать. Для этого встаньте мышкой на пункт **Modbus TCP Master** и щёлкните правой кнопкой мышки и в отрывшемся меню выберите пункт **Add Device.**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



У Вас в древе проекта появится пункт с Ведомым устройством,

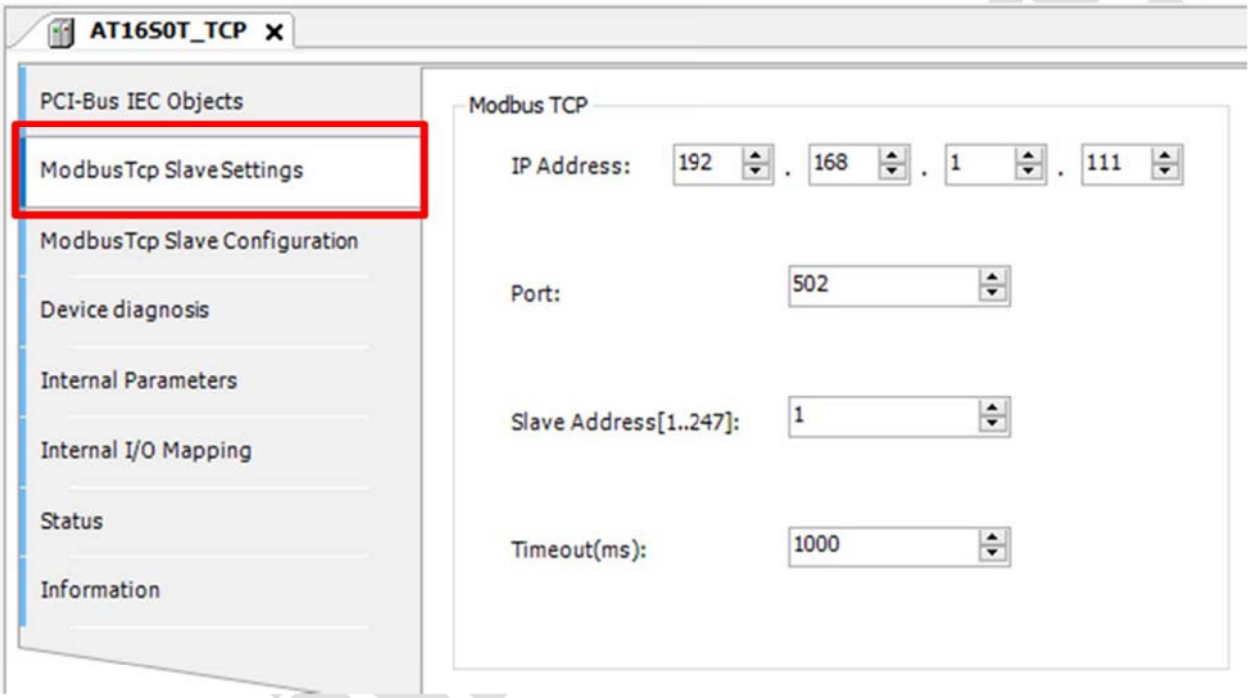


которому можно сразу переименовать на нужное название Ведомого. В нашем примере в качестве Ведомого будет использоваться контроллер Optimus Drive AT16S0T-RU

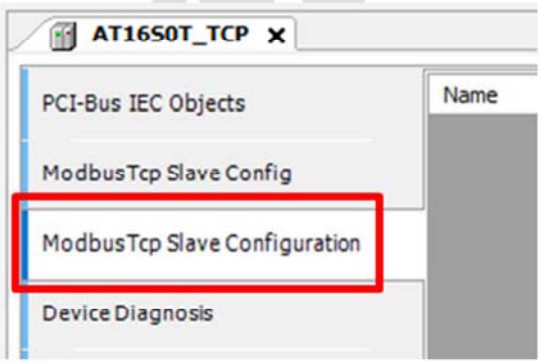


Имя узла - AT16S0T_TCP

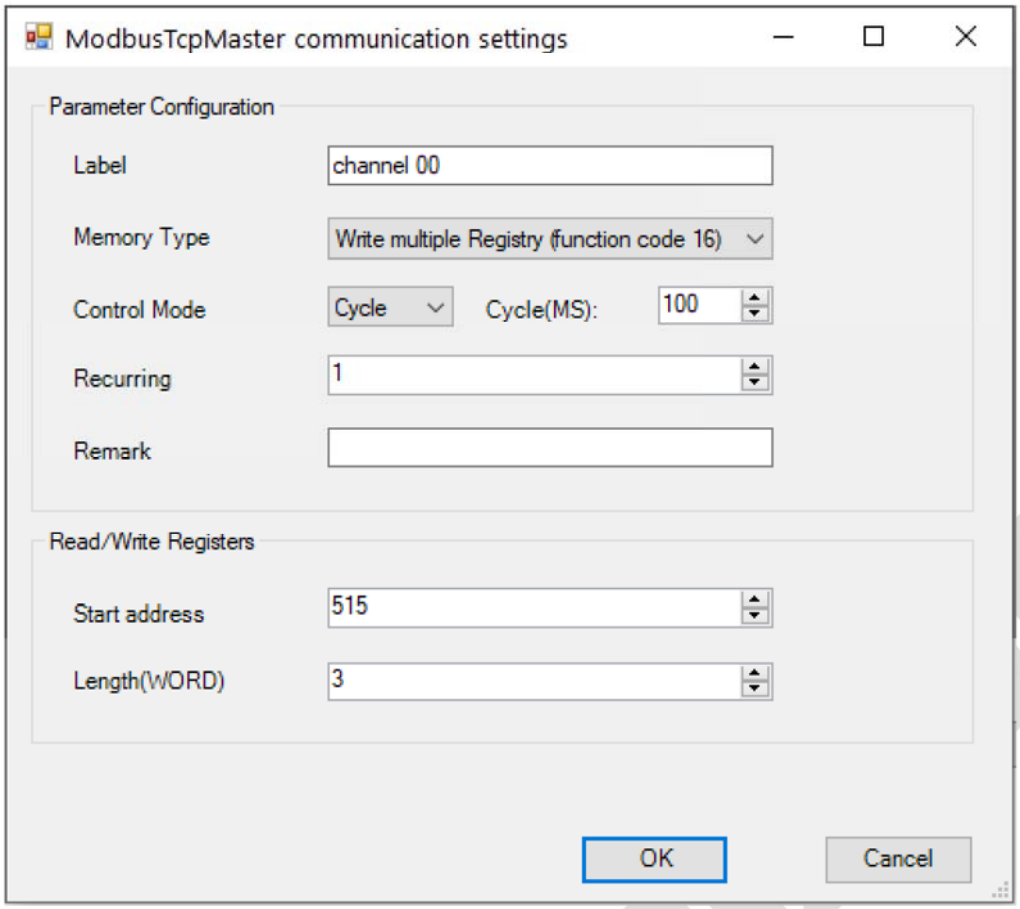
Щёлкните дважды левой кнопкой мышки на Ведомом, откроется вкладка настройки обмена с Ведомым. В разделе **Modbus TCP Slave Config** установите IP адрес Ведомого (в нашем примере 192.168.1.111), Modbus адрес Ведомого (Slave Address), порт (Port), Таймаут (Timeout):



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)

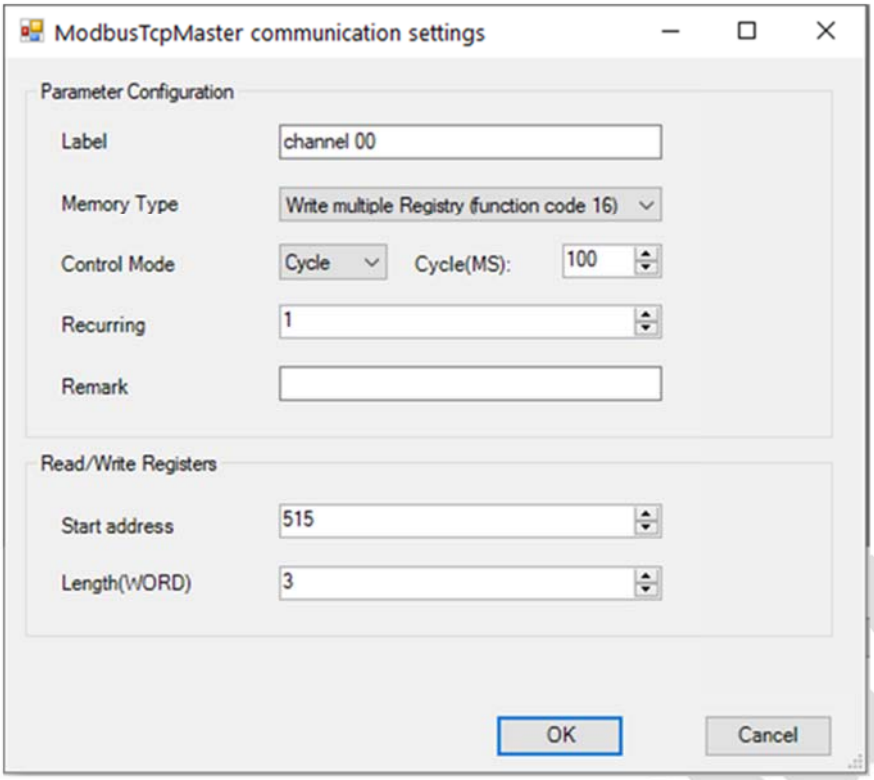
Read/Write Register – блок, в котором оформляется чтение/запись данных из Ведомого

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

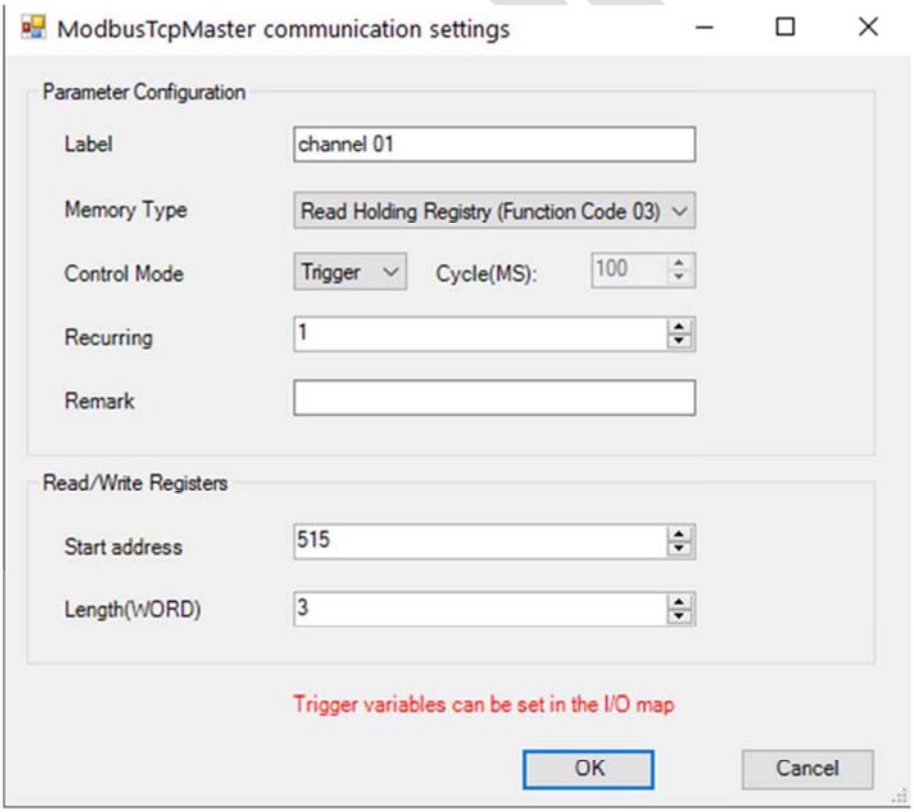
Length – Длина данных для чтения/записи

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.



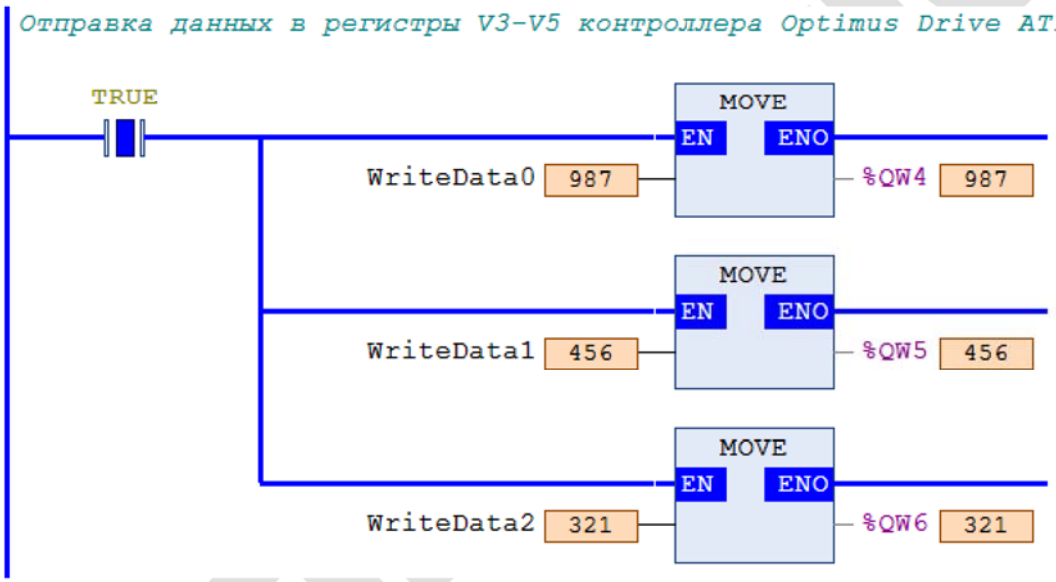
Чтение. Имя – Channel 01, Команда 3 (групповое чтение регистров), метод опроса – Trigger, из программы битом. Начальный регистр для чтения 515 (0x0203), количество последовательно считываемых регистров 3 штуки.



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**:

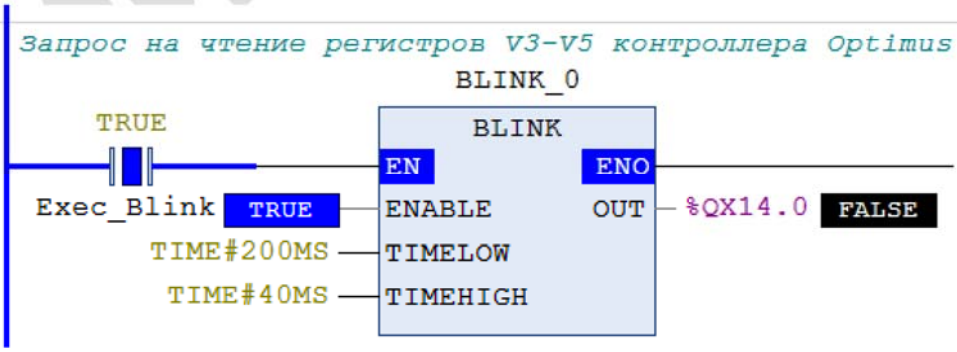
Variable	Mapping	Channel	Address	Type	Unit	Description
		ErrorCode	%IW6	WORD		ModbusTCP ErrorCode
		channel 00	%QW4	ARRAY [0..2] OF WORD		Write multiple Registry (function code 16)
		channel 00[0]	%QW4	WORD		Write #0515)
		channel 00[1]	%QW5	WORD		Write #0516)
		channel 00[2]	%QW6	WORD		Write #0517)
		channel 00	%IX14.0	BOOL		Processing complete state
		channel 01	%IW8	ARRAY [0..2] OF WORD		Read Holding Registry (Function Code 03)
		channel 01[0]	%IW8	WORD		Read #0515
		channel 01[1]	%IW9	WORD		Read #0516
		channel 01[2]	%IW10	WORD		Read #0517
		channel 01	%IX22.0	BOOL		Processing complete state
		channel 01	%QX14.0	BOOL		Trigger control bit

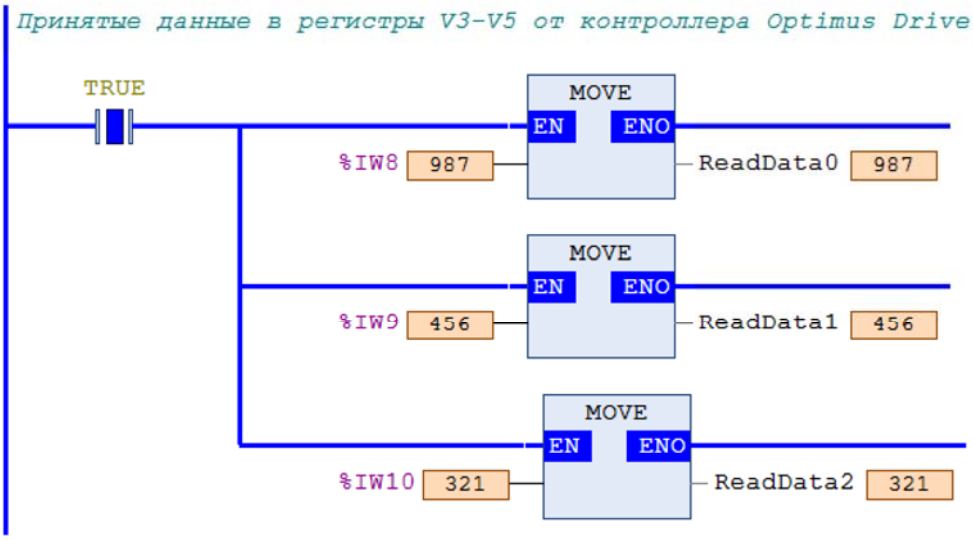
В нашем примере для записи выделены %QW4, %QW5 и %QW6. Для чтения выделены регистры %IW8, %IW9 и %IW10. Данные регистры можно напрямую использовать в программе для обмена данными с Ведомым:



Для запуска триггера удобно использовать команду Blink:

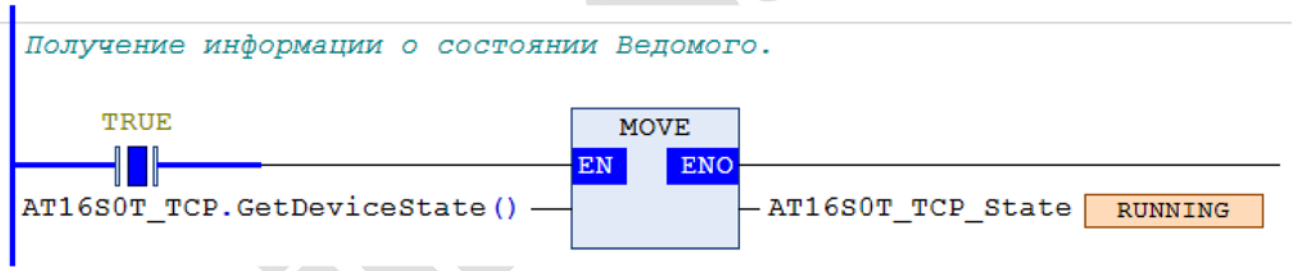
channel 01	%QX14.0	BOOL		Trigger control bit





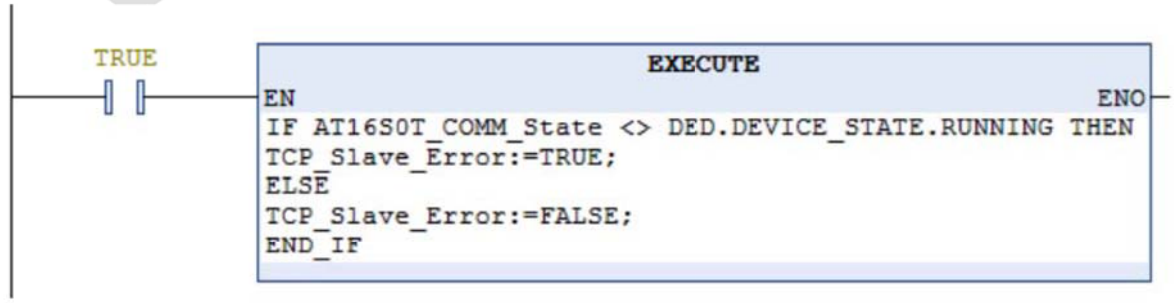
Ошибки связи можно проверять через свойство узла **GetDeviceState** и переменную типа **DED.DEVICE_STATE**

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:

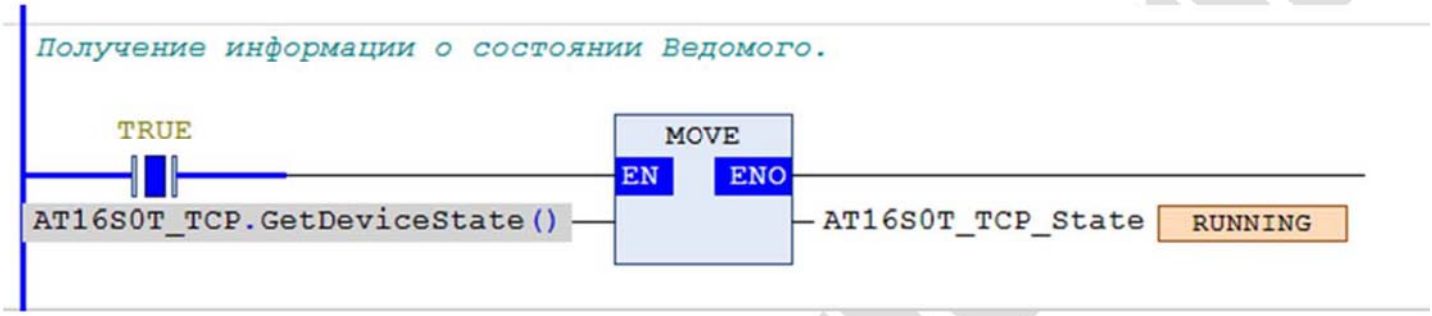
TCP_Slave_Error: BOOL;



Текст программы из блока Execute:

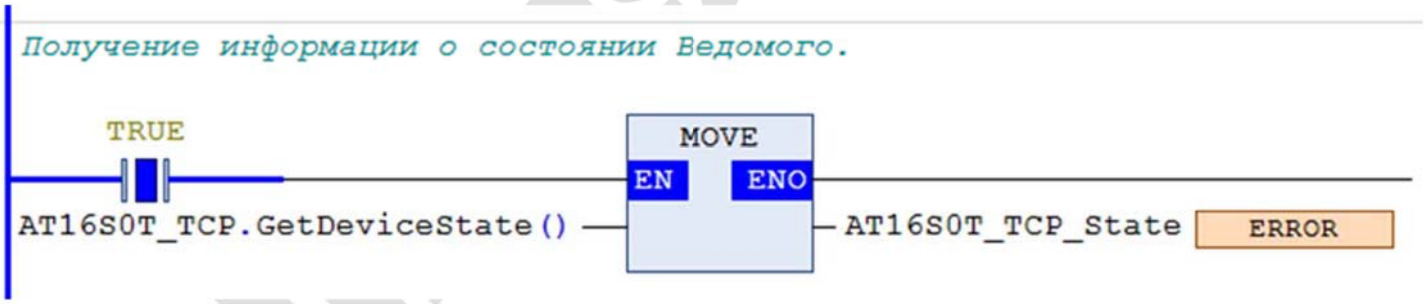
```
IF AT16S0T_COMM_State <> DED.DEVICE_STATE.RUNNING THEN
TCP_Slave_Error:=TRUE;  //включить флаг ошибки
ELSE
TCP_Slave_Error:=FALSE;  //выключить флаг ошибки
END_IF
```

Нормальное состояние ведомого:



Modbus_TCP.TCP_Slave_Error	Controller_1.Application	BOOL	FALSE
Modbus_TCP.AT16S0T_TCP_State	Controller_1.Application	DEVICE_STATE	RUNNING

Потеря связи с ведомым:

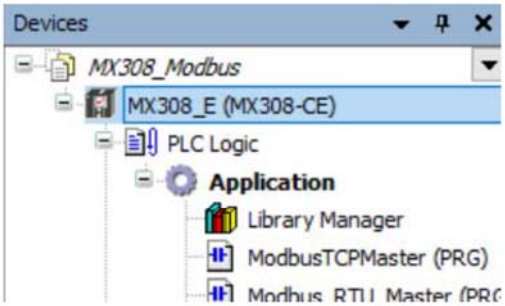


Modbus_TCP.TCP_Slave_Error	Controller_1.Application	BOOL	TRUE
Modbus_TCP.AT16S0T_TCP_State	Controller_1.Application	DEVICE_STATE	ERROR

Связь по протоколу Modbus TCP Slave

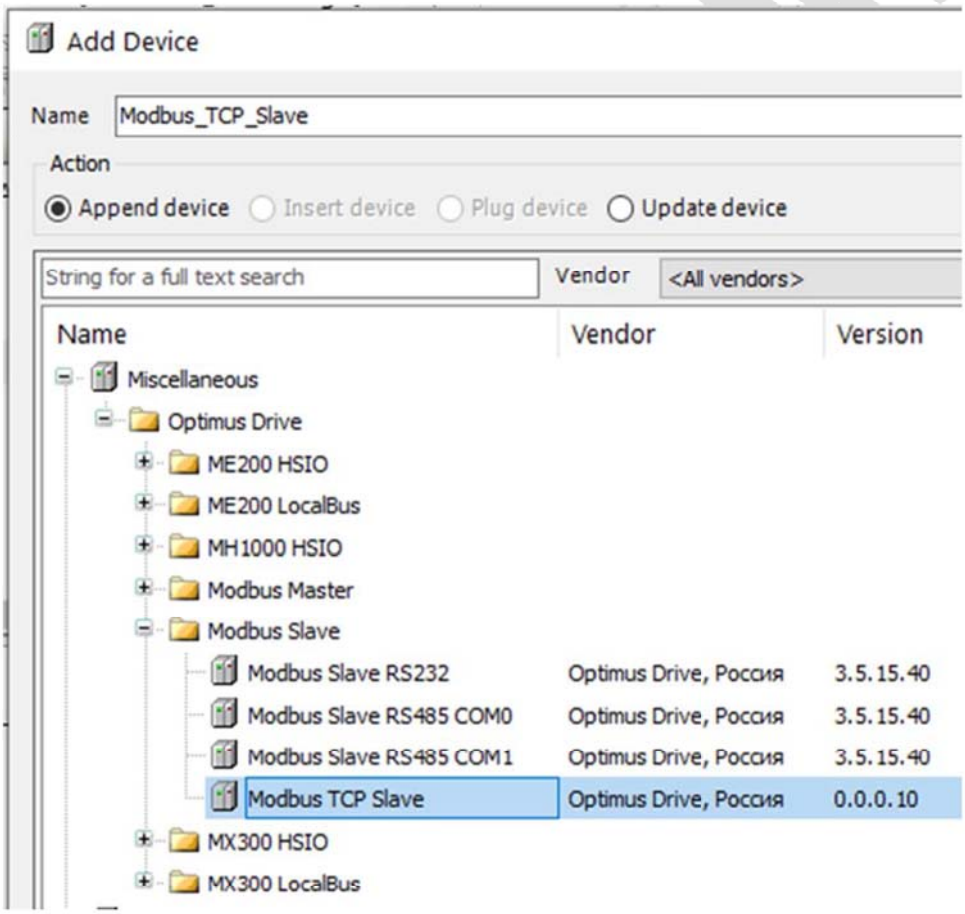
Контроллеры серии MX300 могут работать как в режиме Modbus TCP Client (Master), так и в режиме Modbus TCP Server (Slave), причём оба режима могут использоваться одновременно. В данной главе рассматривается организация связи с контроллером по протоколу Modbus TCP Server (Slave).

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

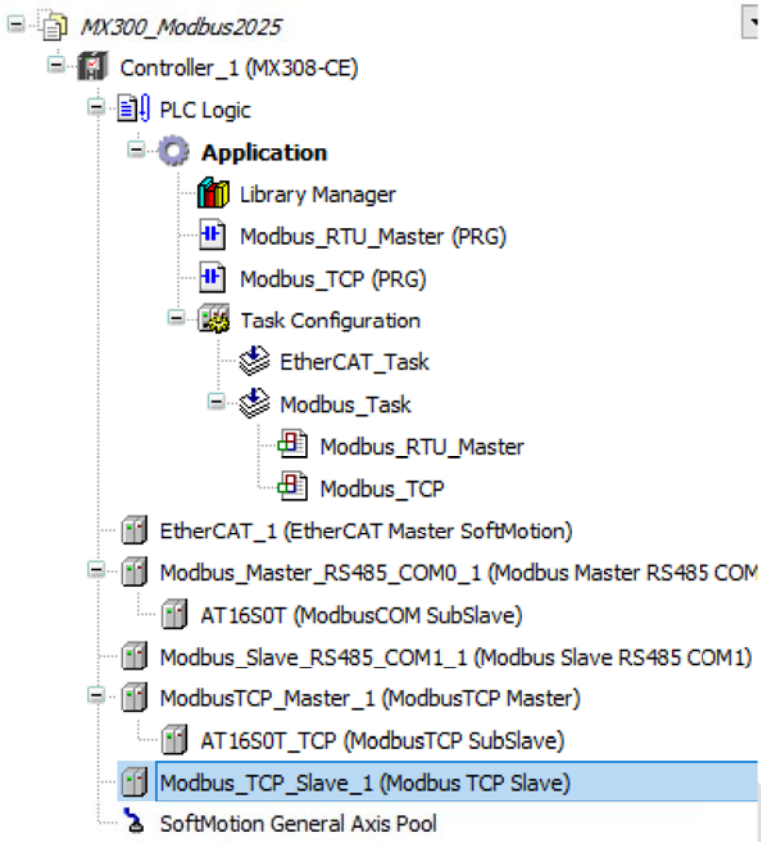


В появившемся меню выберите пункт **Add Device.**

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)



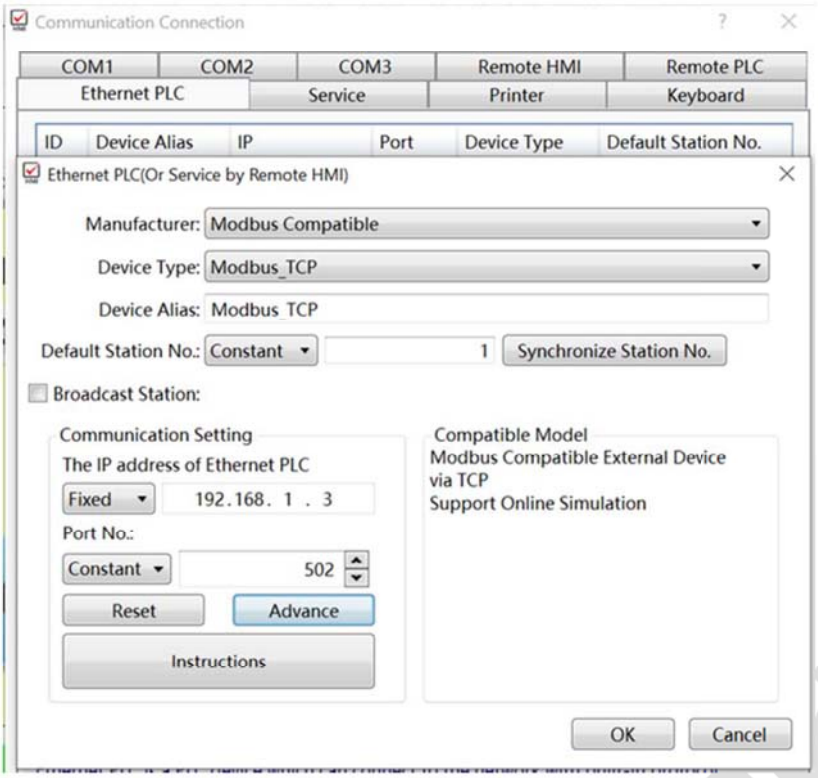
После добавления адаптера в проект и его загрузки, контроллер откроет доступ к своей памяти по протоколу Modbus TCP через порт Ethernet. Согласно спецификации протокола Modbus, он имеет возможность читать булевы и словные регистры без обозначения типов данных в них. Поэтому таблица адресов Modbus будет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

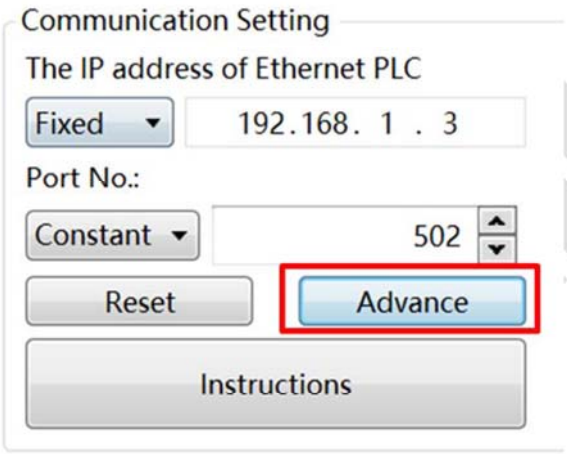
Регистры типа %MW0~%MW65535 являются стандартными словами 16 бит. Адресация к битам в словах не поддерживается. Поэтому в программе биты типа %MX0.0... можно использовать, но их состояние нужно передавать как словный регистр, а потом на стороне Мастера разбирать по битам.
Например, маркеры %MX0.0.. %MX0.7 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.

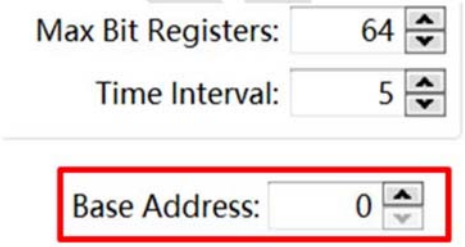
Контроллеры серии MX300 могут выступать в качестве Ведомого устройства для любого стандартного Мастера Modbus TCP. Например, можно рассмотреть в связь с панелью оператора Optimus Drive VI20-070S-FE-RU.

В панели необходимо выбрать стандартный Modbus TCP драйвер:



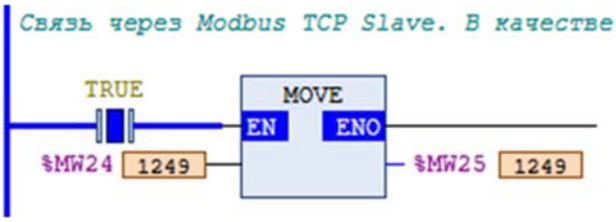
Указать IP адрес контроллера. В нашем примере 192.168.1.3 и во вкладке **Advance** убрать смещение адреса:





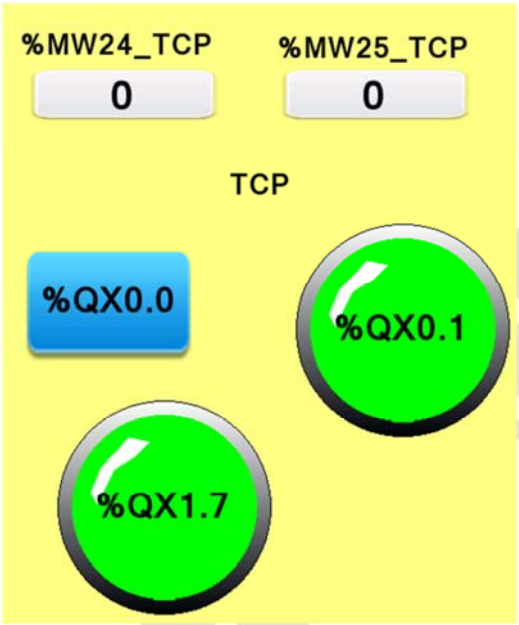
В программе контроллера создана простая программа для отображения данных, в которой задействованы следующие регистры:

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

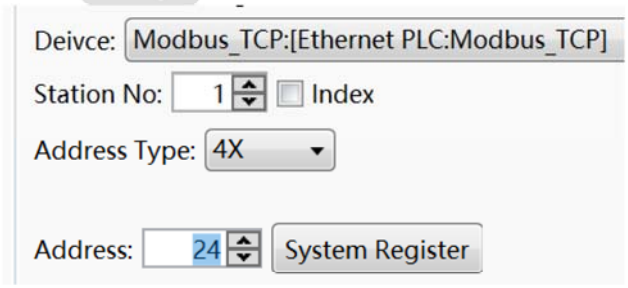


Регистр %MW24 - источник данных, регистр %MW25 - приёмник данных.

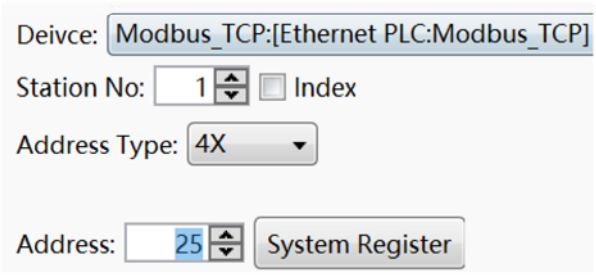
В панели оператора нарисован простой экран с пятью объектами соответственно: Input, Display, Button и 2 штуки Indicator.



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

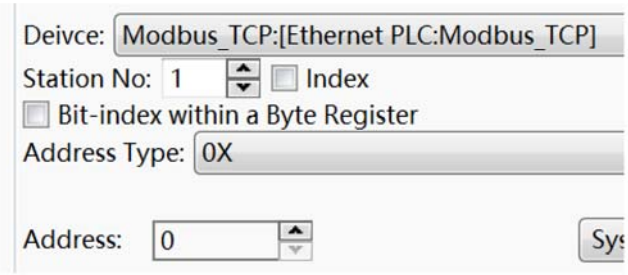


а к регистру %MW25 так

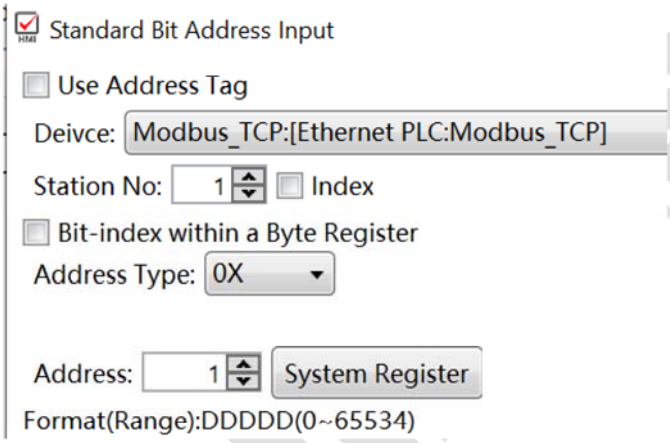


Таким образом, нужно просто указать номер регистра %MW****.

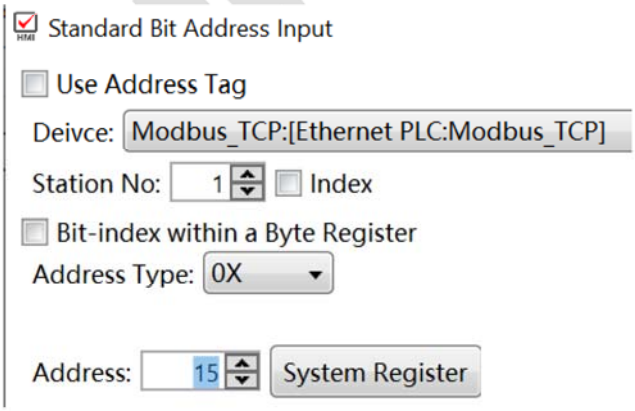
Обращение к булеву регистру %QX0.0 выглядит так:



к регистру %QX0.1 так:



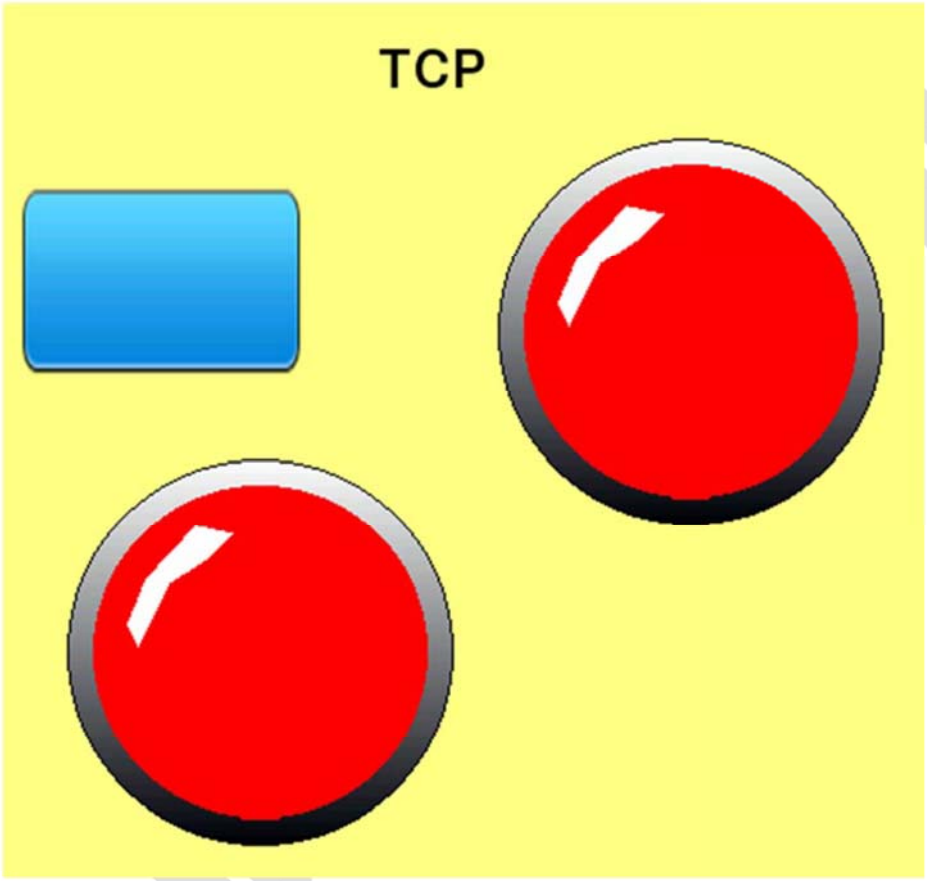
а к регистру %QX1.7 так:



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.

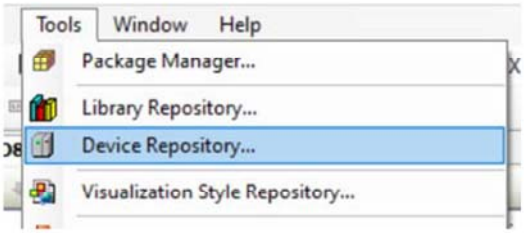
При начале опроса контроллера панель будет отображать состояние регистров:



Работа с преобразователем частоты серии Optimus Drive AD800 по сети EtherCAT

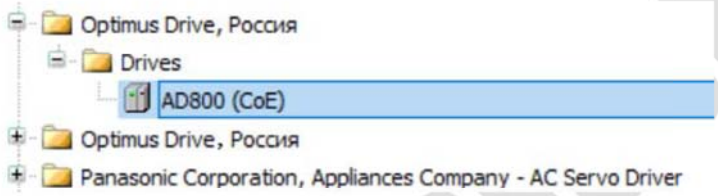
Преобразователи частоты (ПЧ) серии **AD800** имеют возможность расширяться функциональными платами. Для интеграции ПЧ в сеть EtherCAT используется плата **ET01**. Установка и работа с данной платой приведены в Руководстве по эксплуатации «**Плата расширения EtherCAT ET01**».

Для включения ПЧ в состав проекта контроллера серии MX300 необходимо наличие в проекте файла с описанием устройства, или как его ещё называют XML файла EtherCAT. Данный файл входит в состав пэкиджа для контроллера, но при необходимости загрузки другой версии файла нужно воспользоваться меню **Tools – Device Repository**:



Нажать кнопку **Install**, выбрать нужный файл и установить в проект.

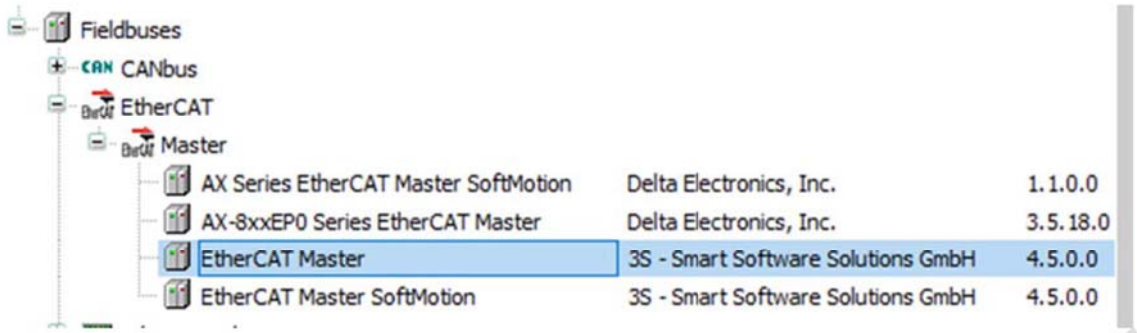
Файл с описанием будет лежать по пути **Fieldbuses – EtherCAT – Slave – Optimus Drive**:



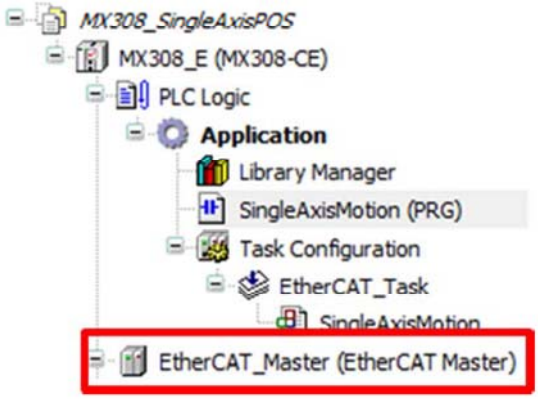
Параметры ПЧ AD800 для работы по EtherCAT:
P7-00 - Сброс на заводские установки (передёрнуть питание).
P0-10 = 3,
P0-11 = 20,
P0-17 = 0,
P0-18 = 2,
P0-29 = 1

Далее необходимо добавить в проект EtherCAT адаптер. Для этого щёлкните правой кнопкой на названии контроллера и в меню выберите пункт **Add Device**:

В открывшемся окне выберете пункт **EtherCAT - Master - EtherCAT Master**:

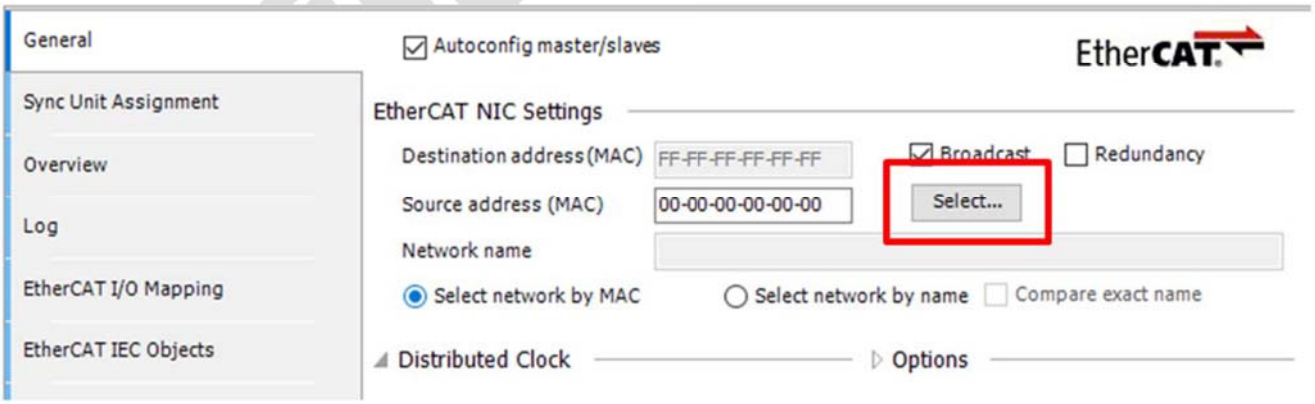


В древе проекта появится пункт:

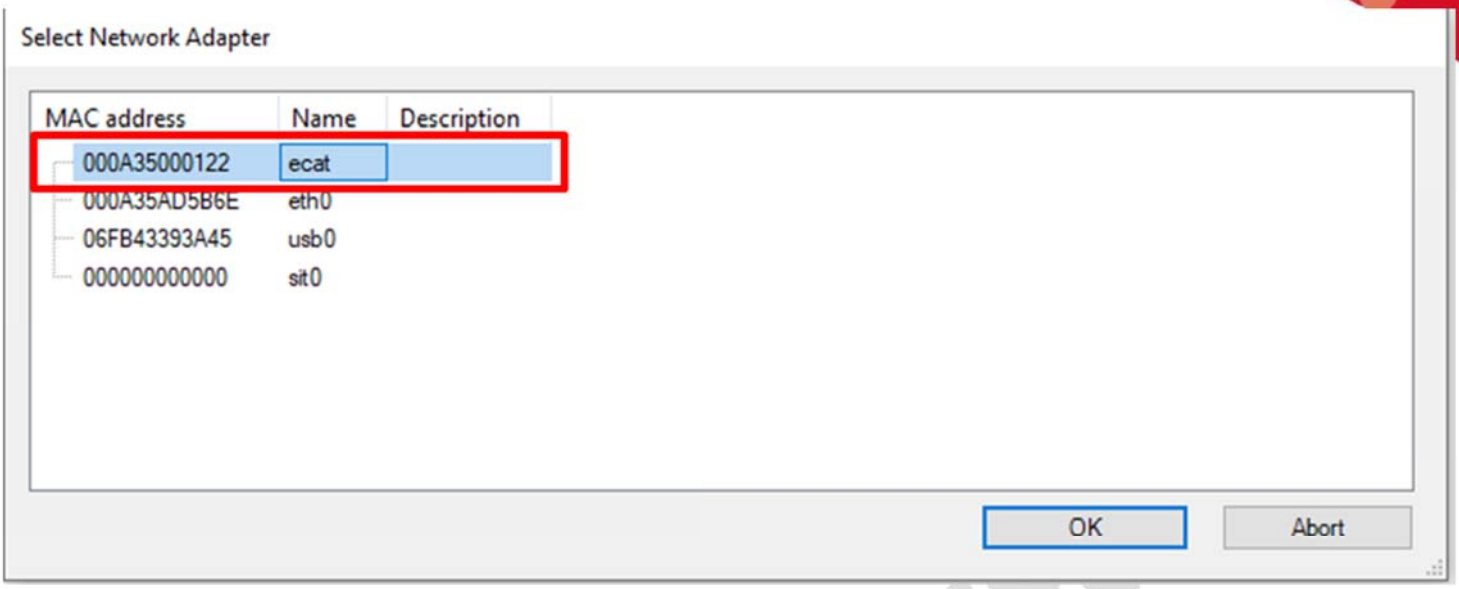


Далее необходимо назначить в проекте Мастера сети EtherCAT. В нашем примере это будет контроллер MX300, поэтому необходимо установить с ним связь (см. соответствующий раздел данного Руководства):

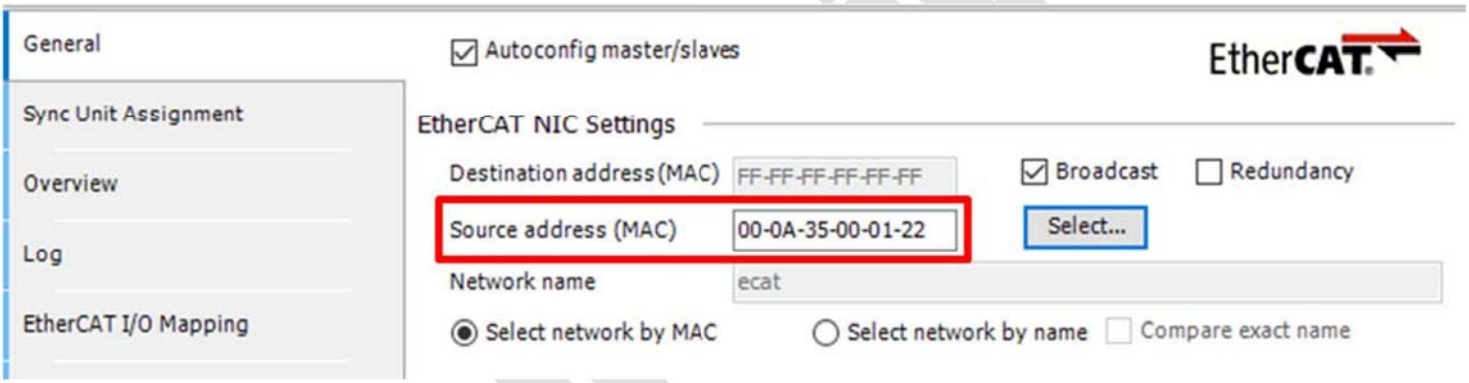
Далее щёлкните дважды левой кнопкой мышки на пункте **EtherCAT Master** и в открывшейся вкладке выберите пункт **General** и нажмите кнопку **Select**:



В открывшемся окне выберите пункт (MAC адрес у каждого контроллера свой):

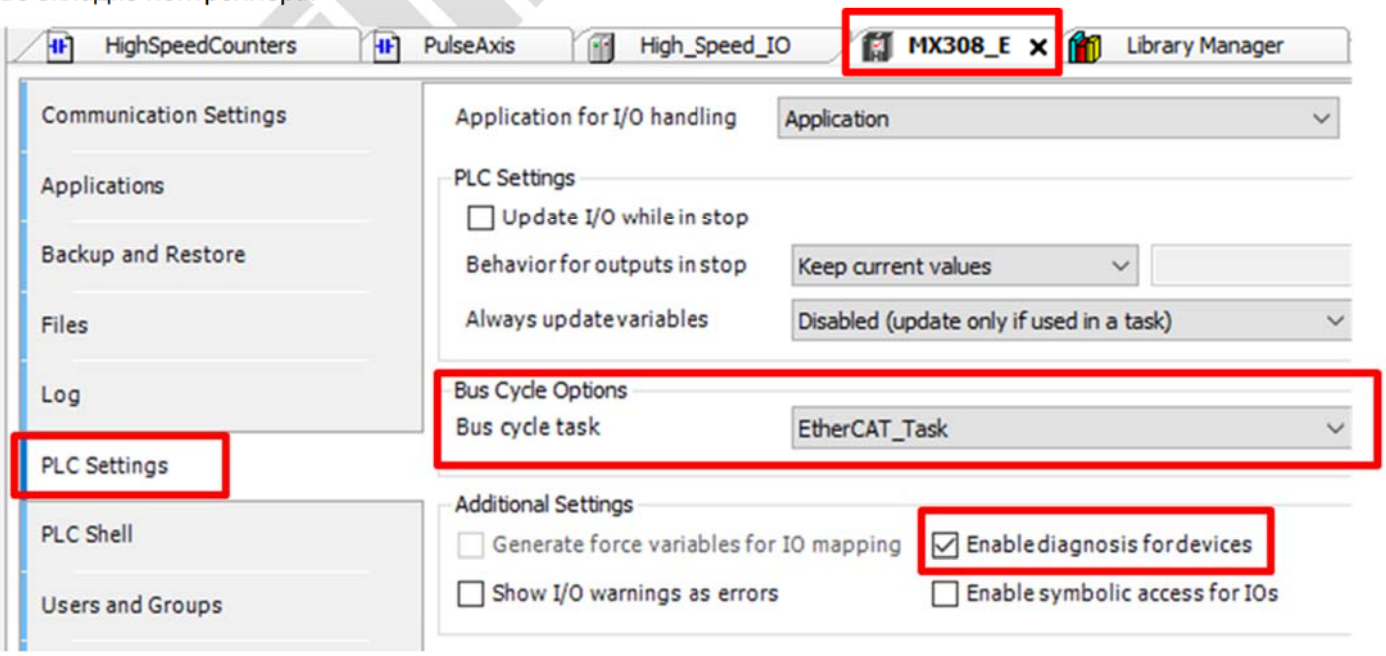


MAC адрес должен установиться в данном поле:

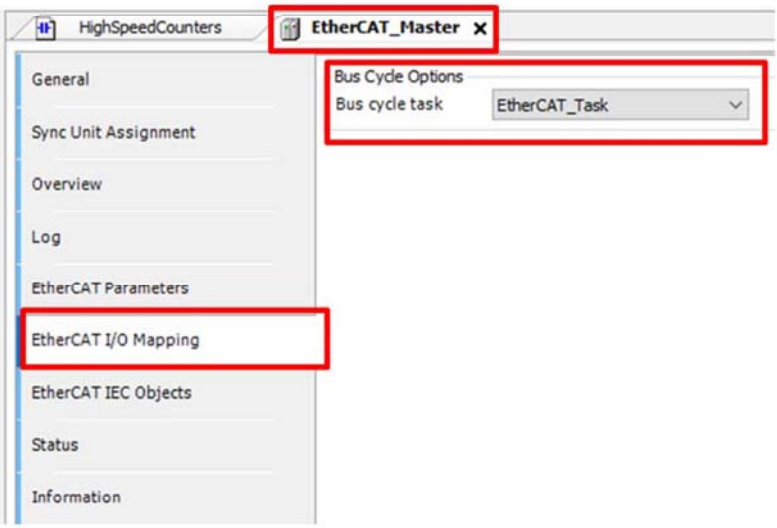


Далее необходимо активировать задачу EtherCAT_Task. Для этого сделайте следующие настройки:

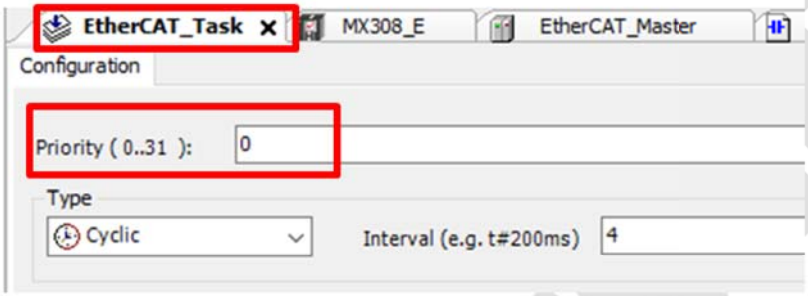
Во вкладке Контроллера:



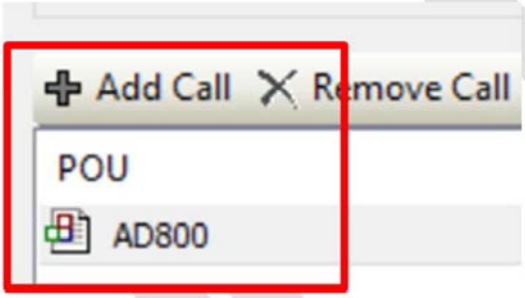
Во вкладке EtherCAT_Master:



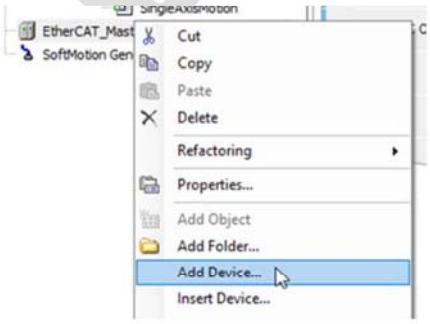
Во вкладке задачи выставить высший приоритет (0), привязать нужные POU:



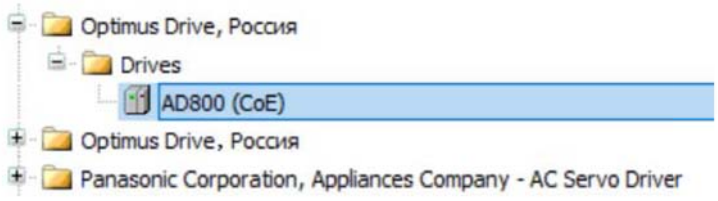
Для приводов AD800 рекомендуется такт шины не менее 4 мс. При большом количестве приводов на шине время опроса необходимо увеличить.



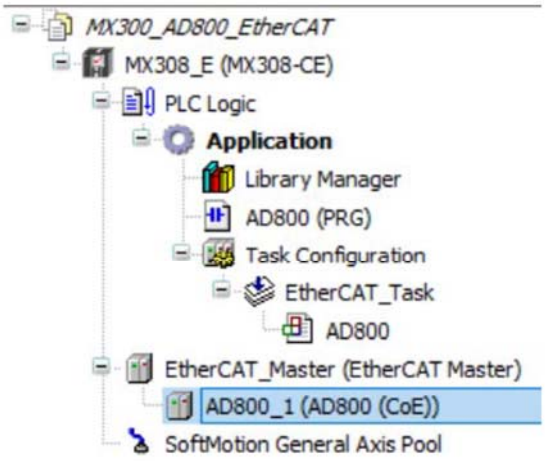
Далее необходимо добавить в проект сервопривод. Для этого щёлкните правой кнопкой мышки на пункте **EtherCAT Master** и в открывшемся окне выберите пункт **Add Device**:



В открывшемся окне выберите Преобразователь Частоты Optimus Drive AD800:
по пути **Fieldbuses – EtherCAT – Slave – Optimus Drive**:



В древе проекта появится пункт AD800:



На данном этапе мы добавили привод как узел сети EtherCAT и после загрузки проекта в контроллер привод станет доступен для управления.
Преобразователи частоты добавляются в проект как обычные устройства, т.е. управляются напрямую через PDO пакеты. Параметрам ПЧ будут поставлены в соответствие регистры контроллера, записывая нужные значения в которые, можно будет управлять приводом.

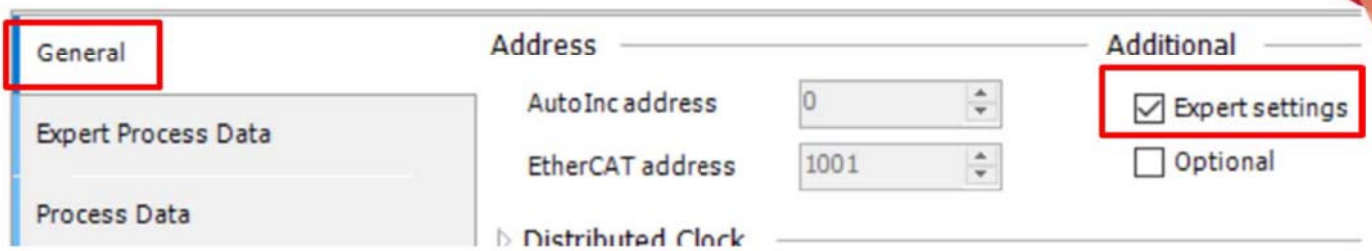
Щёлкните дважды левой кнопкой мышки на названии узла (AD800) и в открывшейся вкладке выберите пункт **Process Data**:

По умолчанию назначаются два регистра на отправку и два на приём данных от привода.

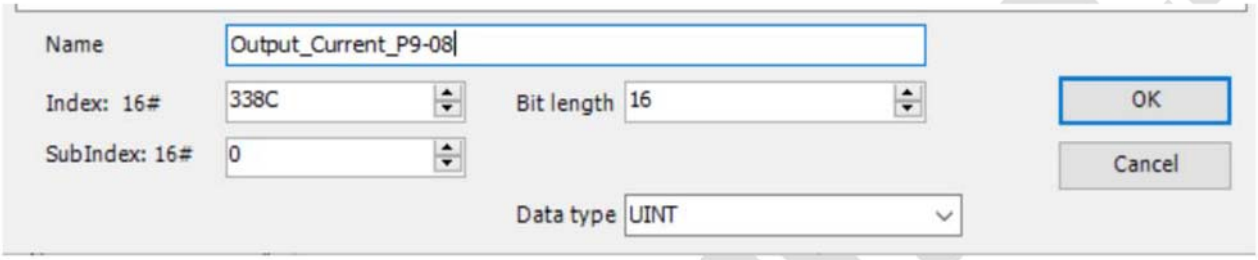
В разделе **Outputs**, т.е. данные в сторону привода, будет два регистра – **ControlWord** (команды приводу) и **Set_Value** (задание скорости в Герцах).

В разделе **Inputs**, т.е. данные от привода в сторону контроллера, будет также два регистра – **Status_Word** (слово состояния привода) и **Output_Frequency** (выходная частота в Герцах).

При необходимости, в PDO пакет можно добавить ещё параметры. Для этого необходимо включить экспертный режим:

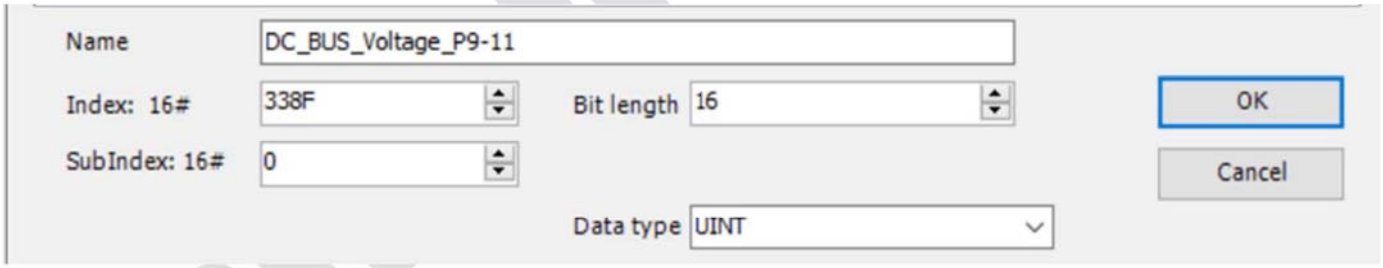


Далее необходимо щёлкнуть мышкой на пункте **Expert Process Data** и выбрать нужную группу, например **Inputs**, и нажать кнопку **Insert**. В открывшемся окне нужно ввести нужный адрес регистра, который Вы хотите добавить. Адрес привязан к параметрам ПЧ и вычисляется путём сложения 0x3000 + номер параметра в HEX. Мониторинговые параметры находятся в группе 9. Например, выходной ток находится в параметр P9-08. Переводим число 908 в HEX, получаем 0x38C, складываем с 0x3000 и получаем число 0x338C и вводим в форму:

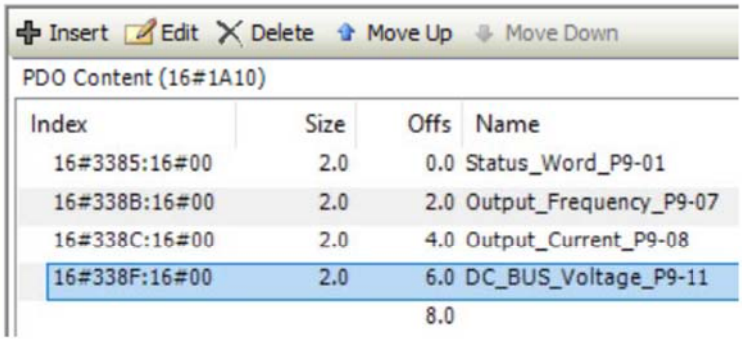


Для данного параметра тип данных нужно установить UNIT. Адрес (Index) и название (Name) вводятся строго в латинской раскладке клавиатуры.

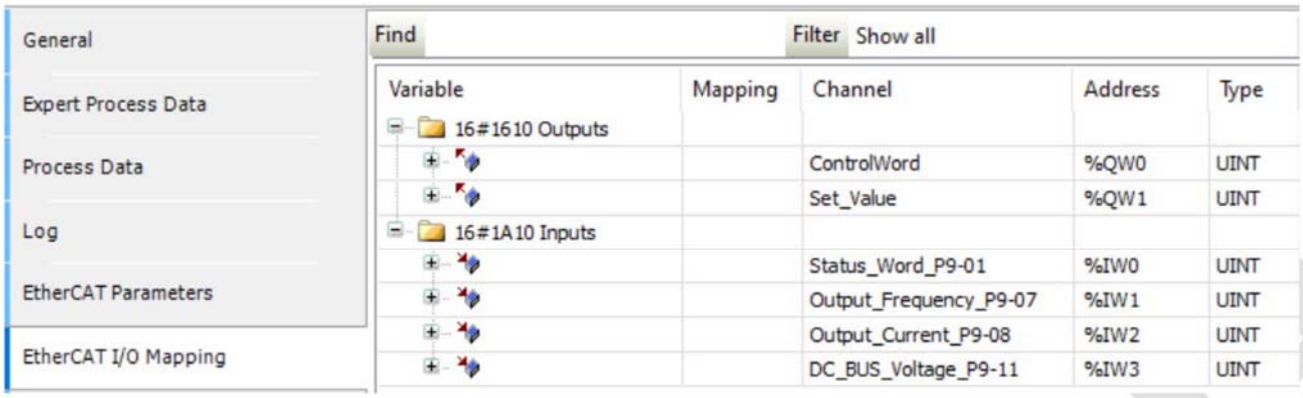
Ещё для примера добавим напряжение на шине DC. Данный параметр находится в P9-11. Переводим число 911 в HEX, получаем 0x38F, складываем с 0x3000 и получаем число 0x338F, нажимаем **Insert** и в открывшемся окне вводим в формате UINT:



В итоге будет список параметров:

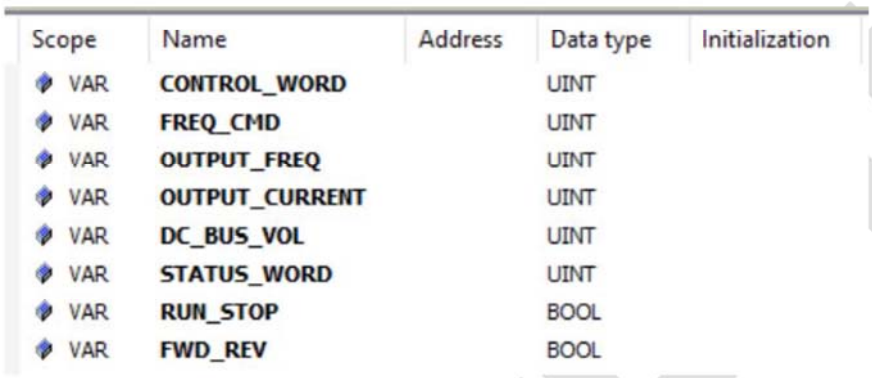


Далее в пункте EtherCAT I/O Mapping можно посмотреть регистры контроллера, через которые будет осуществляться управление приводом:

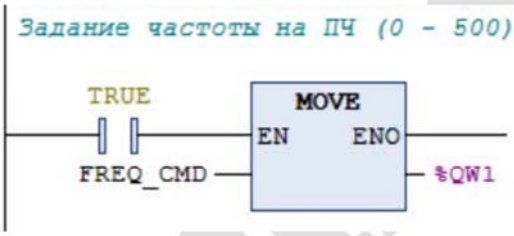


Поверх них можно назначить теги или использовать в программе напрямую.

А можно объявить в программе переменные и через них работать с регистрами, отвечающими за привод:



И отправлять данные в соответствующий регистр, например задание частоты:

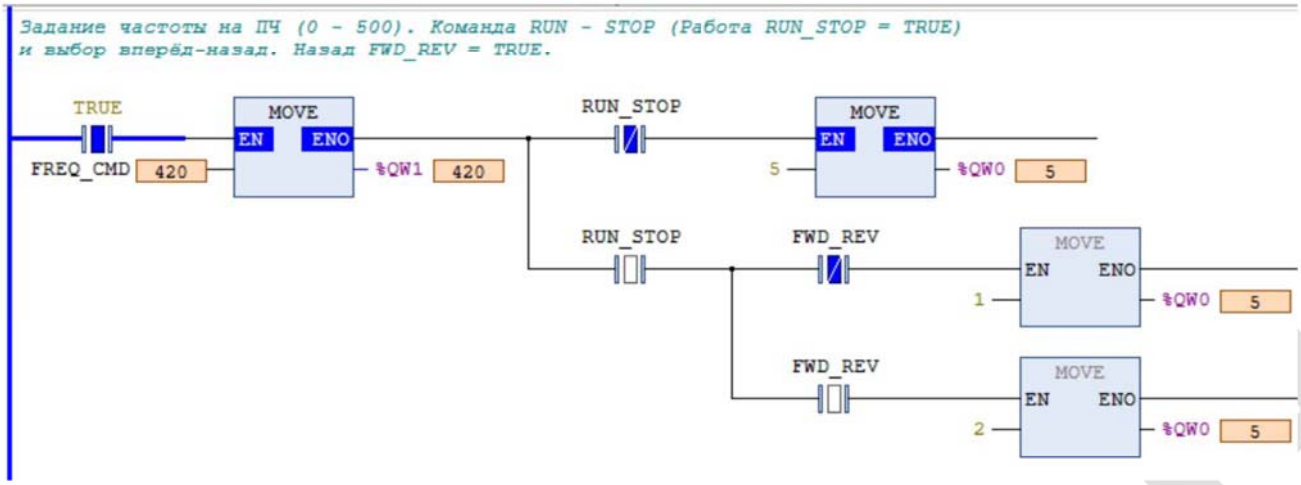


Для управления приводом в контрольное слово нужно записать следующие значения:

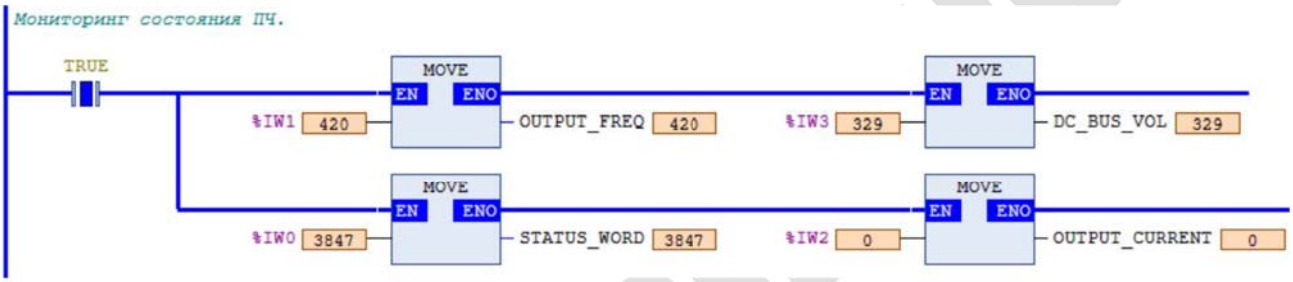
5 – Стоп
1 – Вращение вперёд
2 – Вращение назад

(другие команды можно посмотреть в Руководстве по эксплуатации платы ET01).

Для удобства работы команды можно описать в программе через булевы переменные, которые будут отправлять нужное число в командный регистр, например такой код:



Мониторинг состояния ПЧ можно осуществлять также через переменные:



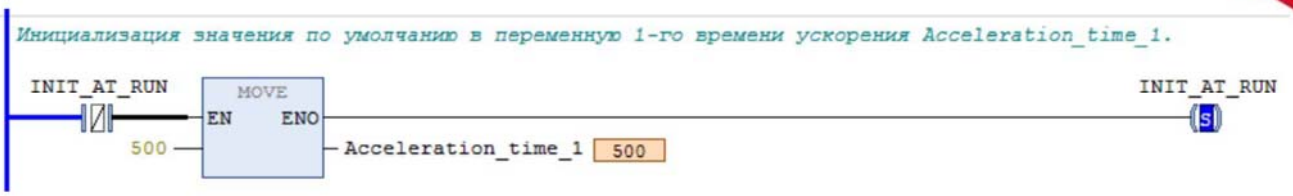
Помимо циклического опроса через пакет типа PDO, привод позволяет опрашивать себя посредством запросов типа SDO (запрос-ответ). Это полезно для конфигурирования привода, которое делается один раз и далее не должно занимать трафик. С этой целью используются команды **ETC_CO_SdoWrite** и **ETC_CO_SdoRead**.

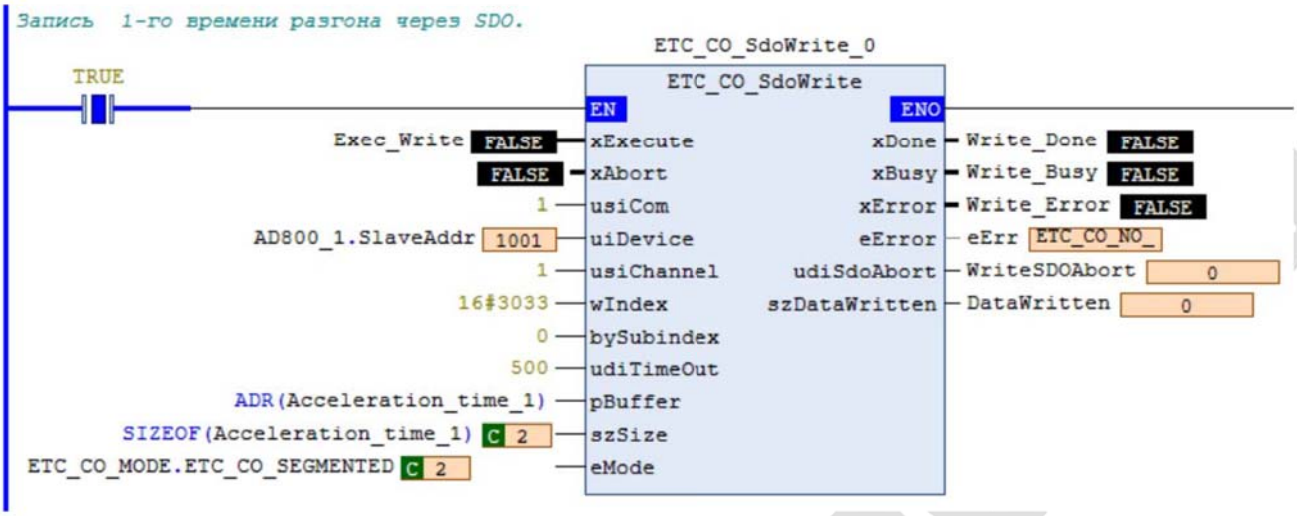
Для примера запишем в привод и для контроля прочитаем 1-е время разгона, это параметр P0-51. Адрес для EtherCAT вычисляется также как для пакетов PDO (см. выше по тексту).

0x3000 + 0x0033 = 0x3033 это EtherCAT адрес параметра P0-51

Блок записи значения времени разгона в привод:

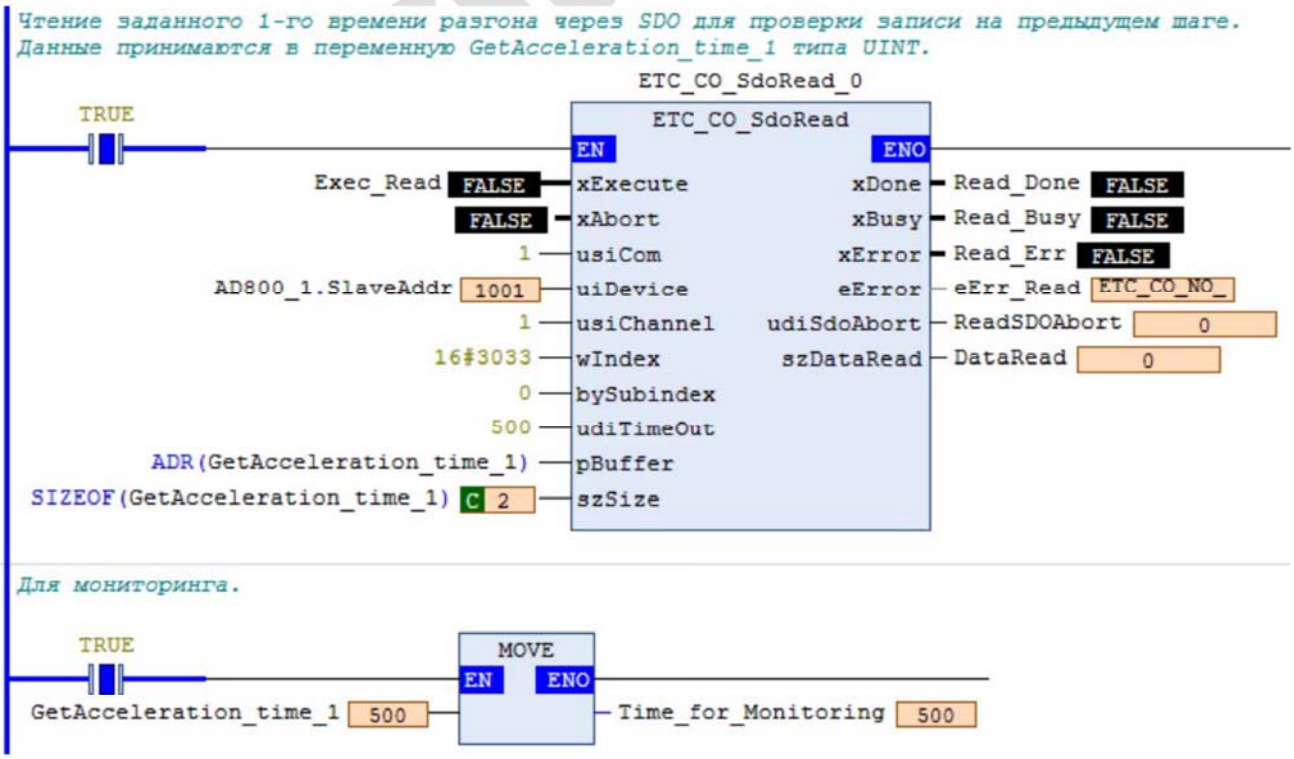
Scope	Name	Address	Data type	Initialization
VAR	**INIT_AT_RUN**		BOOL	
VAR	**ETC_CO_SdoWrite_0**		ETC_CO_SdoWrite	
VAR	**Acceleration_time_1**		UINT	
VAR	**Exec_Write**		BOOL	
VAR	**Write_Done**		BOOL	
VAR	**Write_Busy**		BOOL	
VAR	**Write_Error**		BOOL	
VAR	**eErr**		ETC_CO_ERROR	
VAR	**WriteSDOAbort**		UDINT	
VAR	**DataWritten**		UDINT	





Блок чтения значения времени разгона из привода:

VAR	**ETC_CO_SdoRead_0**	ETC_CO_SdoRead
VAR	**Exec_Read**	BOOL
VAR	**Read_Done**	BOOL
VAR	**Read_Busy**	BOOL
VAR	**Read_Err**	BOOL
VAR	**eErr_Read**	ETC_CO_ERROR
VAR	**ReadSDOAbort**	UDINT
VAR	**DataRead**	UDINT
VAR	**GetAcceleration_time_1**	UINT
VAR	**Time_for_Monitoring**	UINT



Связь по протоколу Ethernet/IP Scanner (Master)

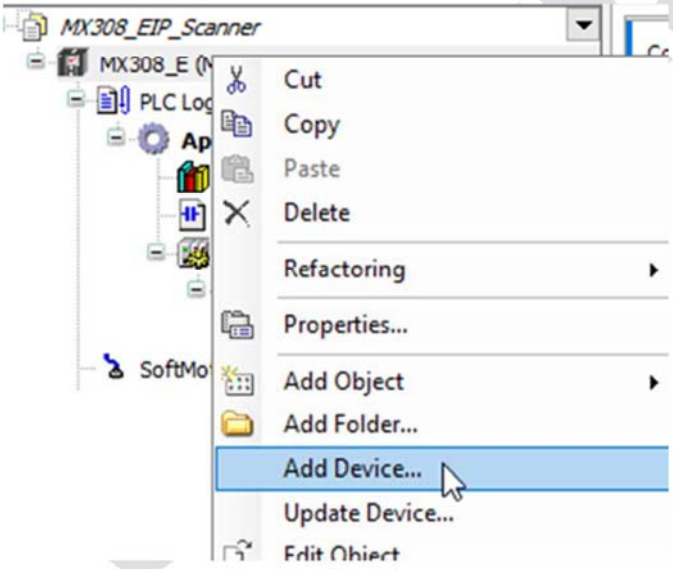
Протокол **Ethernet/IP** (IP = Industrial Protocol) является одним из наиболее распространённых промышленных протоколов в мире и поддерживается сотнями производителей промышленного оборудования. **Ethernet/IP** (сокращённо **EIP**) использует модель **CIP** (Common Industrial Protocol) поверх стандартного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

Контроллеры серии MX300 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).

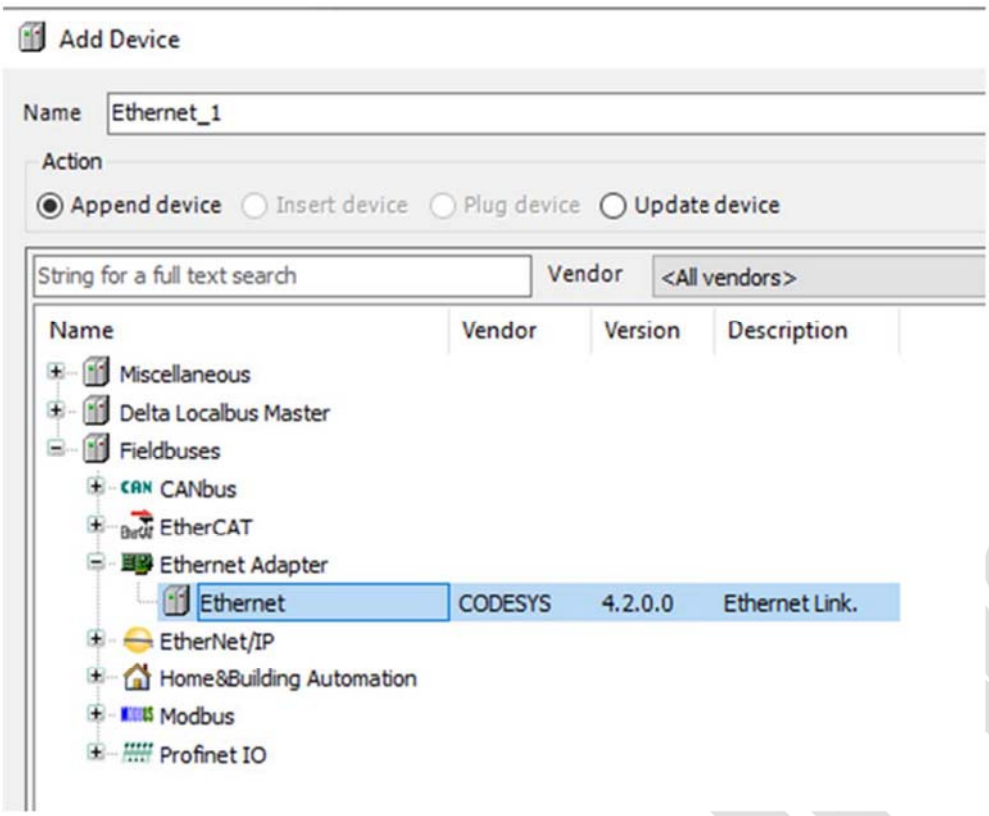
Для использования Ethernet/IP к проекту должны быть подключены следующие библиотеки: IoDrvEthernet, IoDrvEtherNetIP, EtherNetIP Services и DED.

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

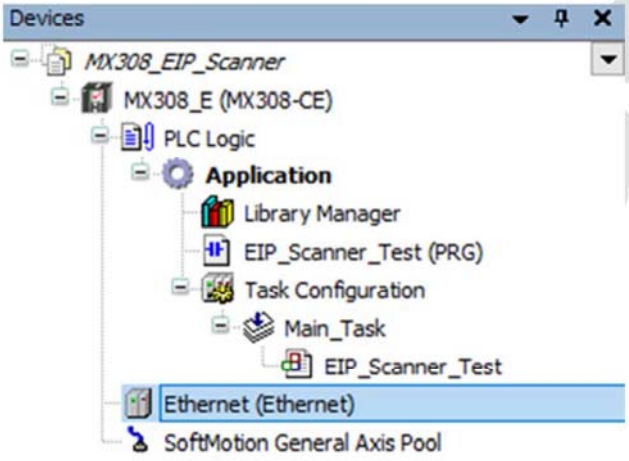
Щёлкните в древе правой кнопкой мышки на пункте **Device** (MX308_E) и в отрывшемся меню выберите пункт **Add Device**



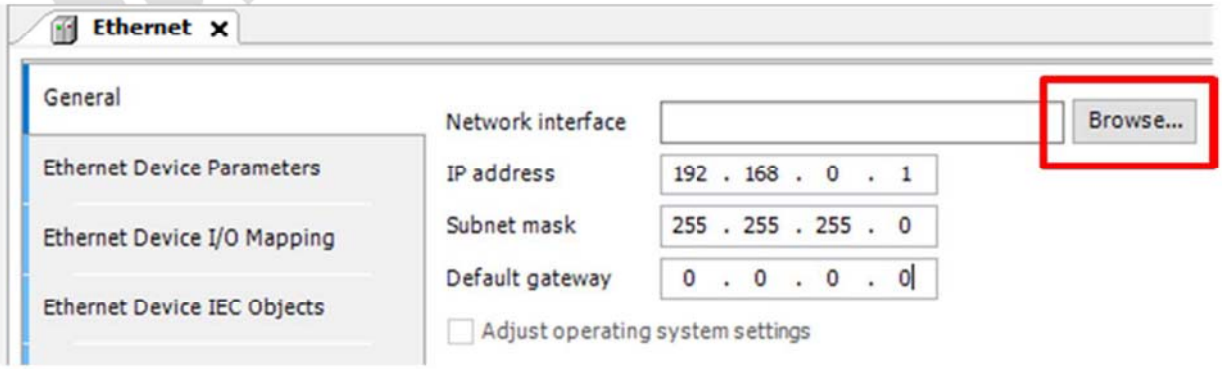
В открывшемся окне выберите **Fieldbuses – Ethernet**:



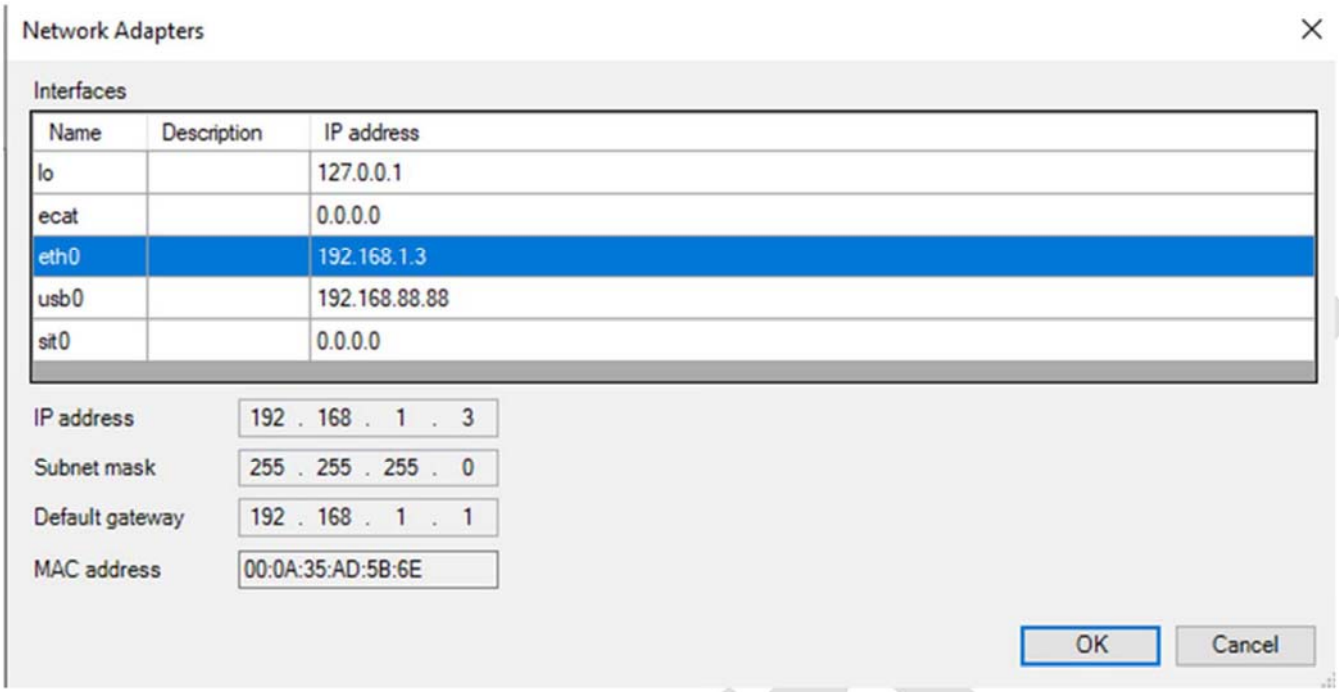
В древе проекта появится пункт **Ethernet**:



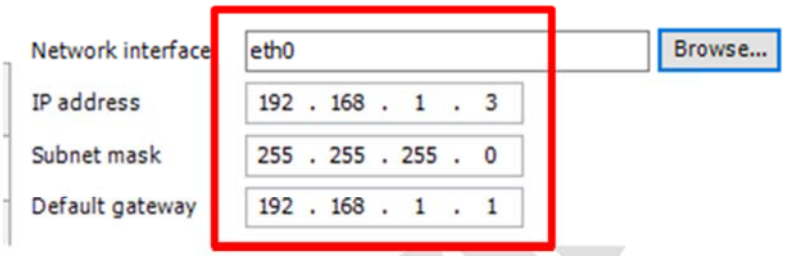
Щёлкните на нём дважды левой кнопкой мышки и в открывшейся вкладке выберите пункт **General**:



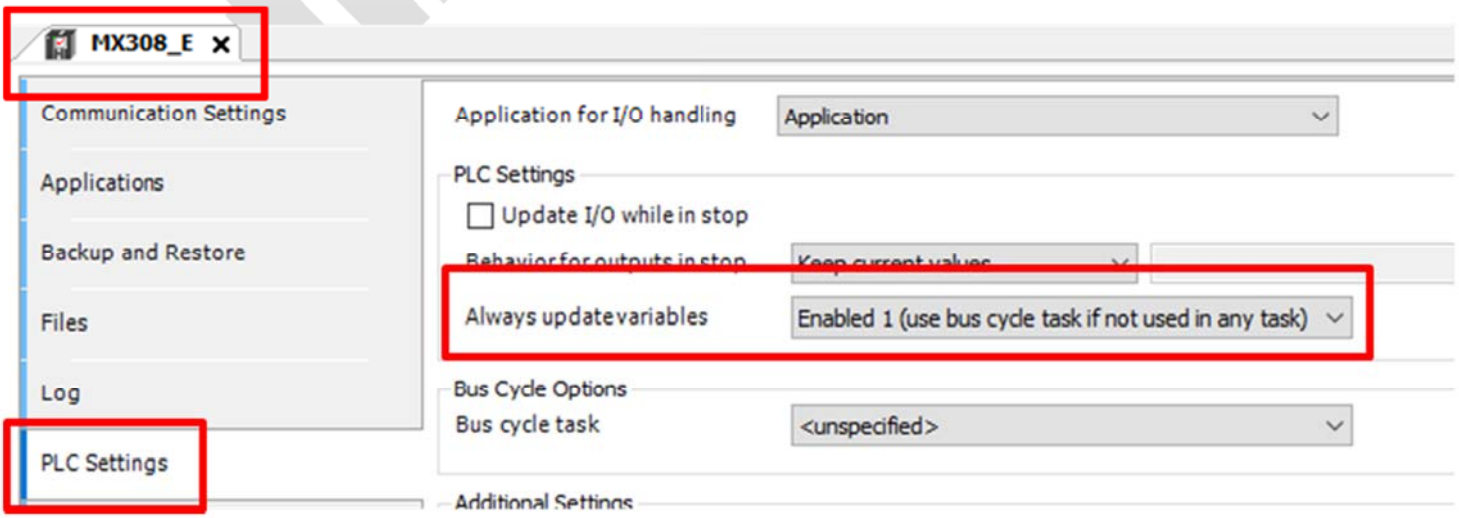
Нажмите кнопку **Browse** и в открывшемся окне выберите IP адрес контроллера:



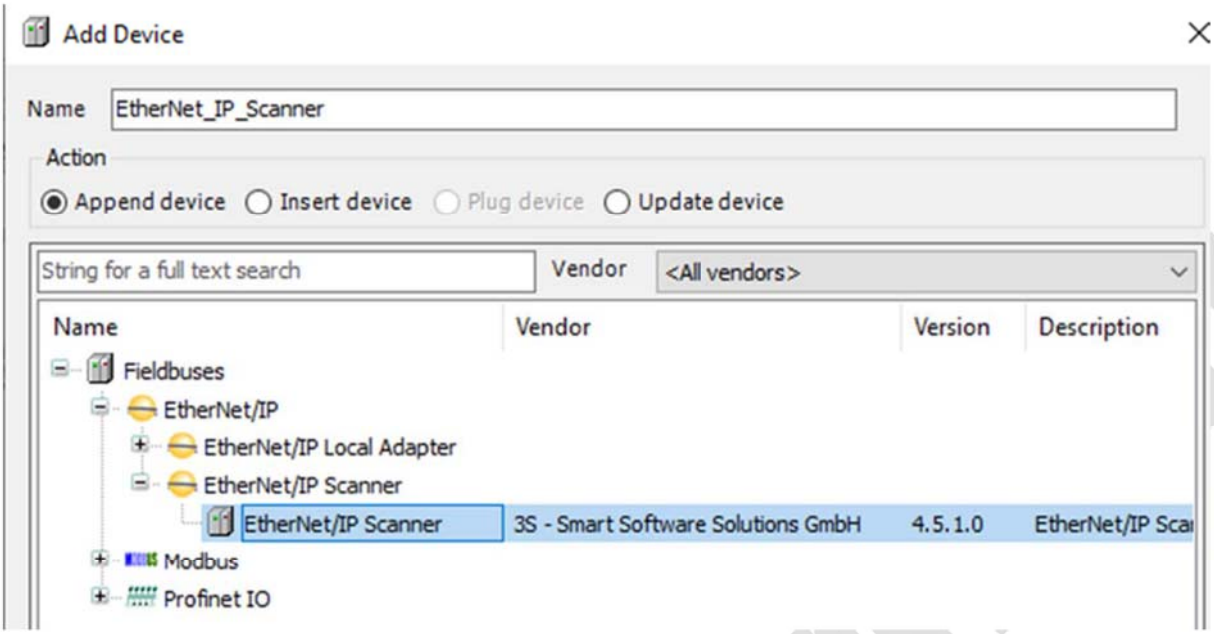
После чего Адаптер будет привязан к Ethernet порту с конкретным IP адресом:



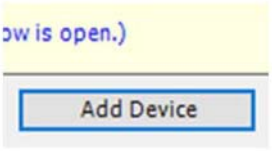
Далее двойным щелчком левой кнопки мыши откройте вкладку **Device** и в пункте **PLC Settings** разрешите обновление переменных. Данный шаг позволяет видеть изменение переменных и регистров без написания в теле программы, т.е. в таблице Watch и в таблицах мэпинга.



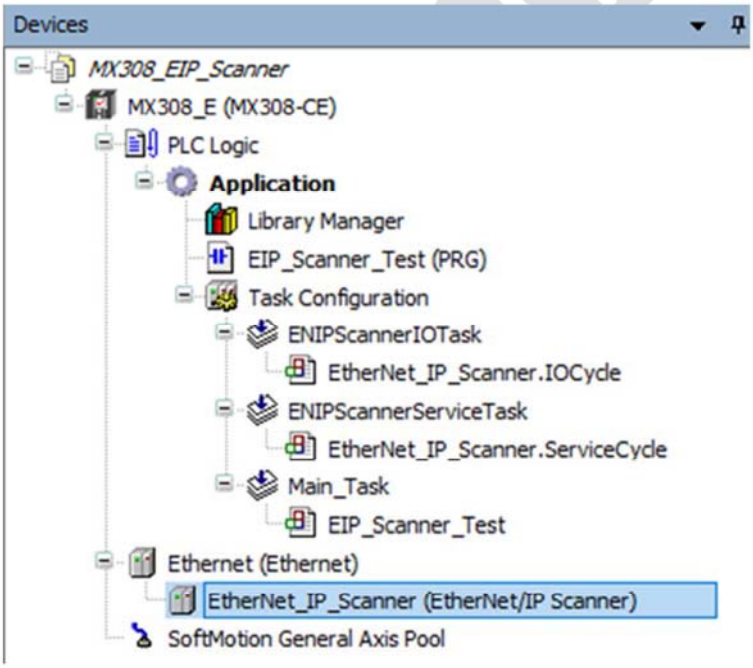
Щёлкните в древе правой кнопкой мышки на пункте **Ethernet** и в отрывшемся меню выберите пункт **Add Device**. Далее в открывшемся окне выберите пункт **Fieldbuses – Ethernet/IP Scanner**:



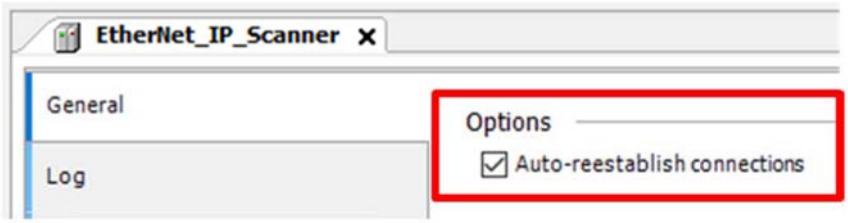
и нажмите **Add Device**:



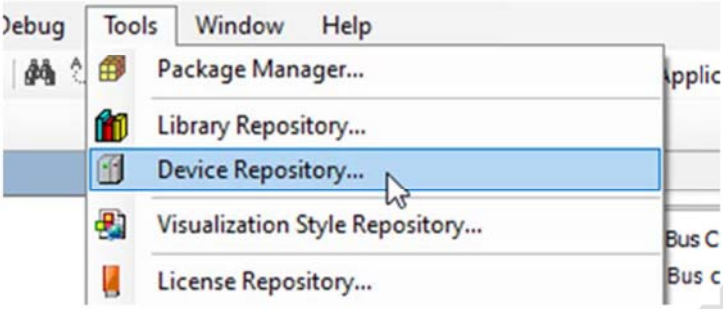
В древе проекта появится пункт **Ethernet/IP Scanner**:



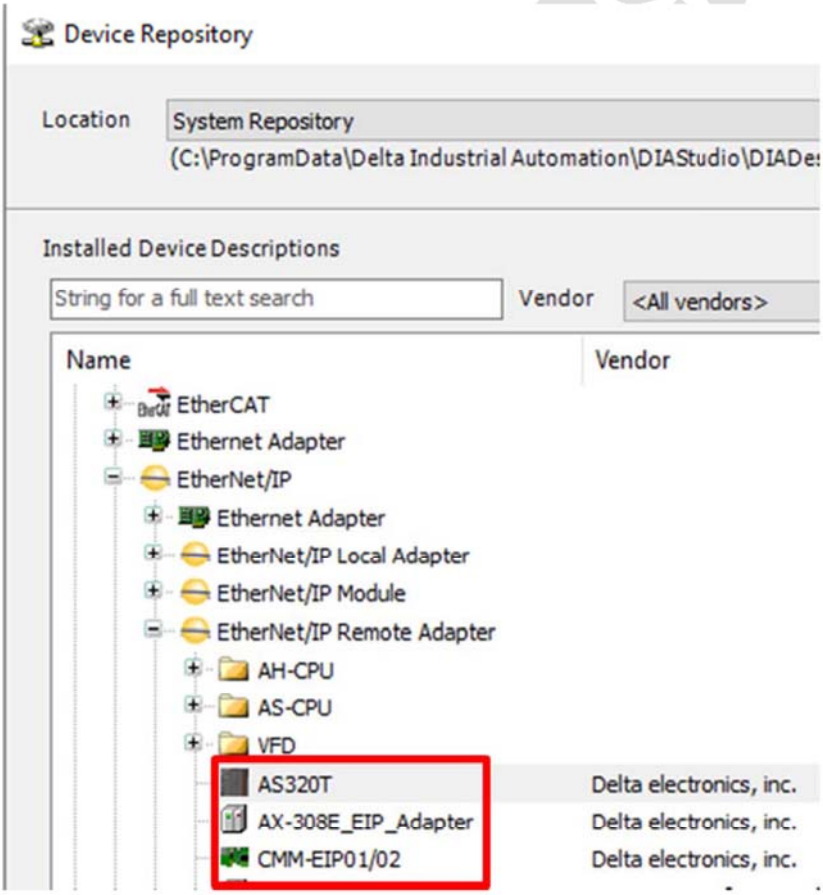
Щёлкните двойным щелчком левой кнопки мышки на пункте **Ethernet/IP Scanner** и в открывшейся вкладке выберите пункт General. Установите флажок автоматического восстановления соединения:



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

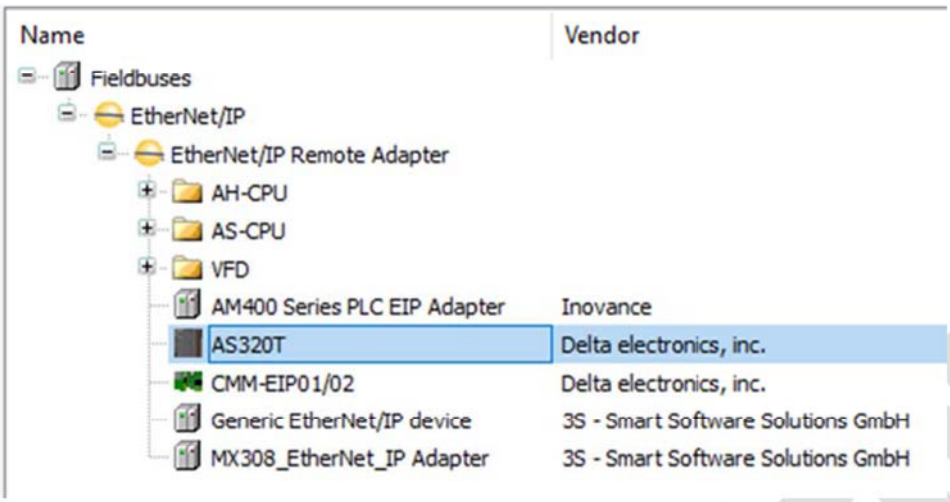


Ниже мы рассмотрим организацию связи с контроллером Delta AS320T и преобразователем частоты Delta MS300 посредством платы CMM-EIP02 по протоколу Ethernet/IP. Предполагается, что на этом этапе будут установлены соответствующие EDS файлы данных устройств:

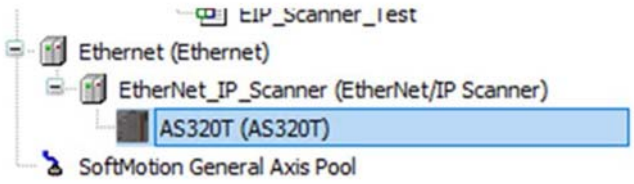


Пример настройки связи с контроллером Delta AS320T по протоколу Ethernet/IP

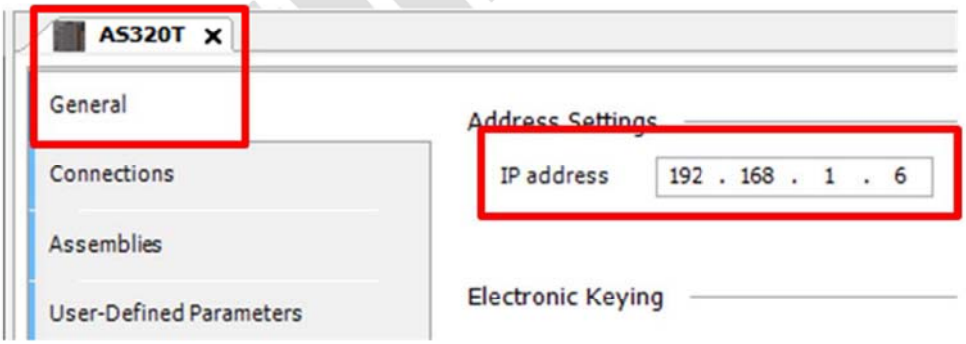
Для начала работы необходимо добавить Ведомое устройство в проект. Для этого щёлкните правой кнопкой мышки в древе проекта на пункте **Ethernet/IP Scanner** и в открывшемся меню выберите пункт **Add Device**. В открывшемся окне выберите из **Device Repository** устройство **AS320T**:



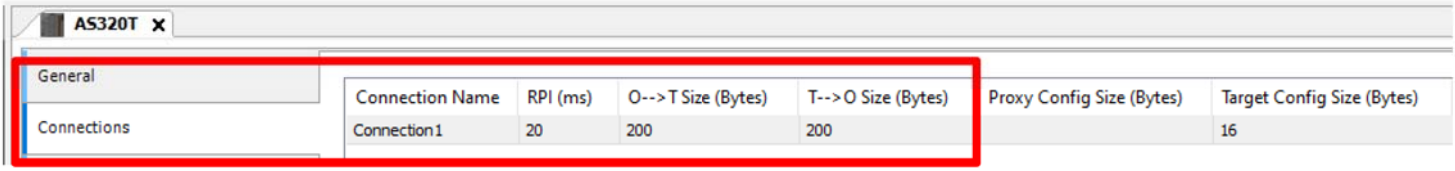
В древе проекта появится устройство:



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:



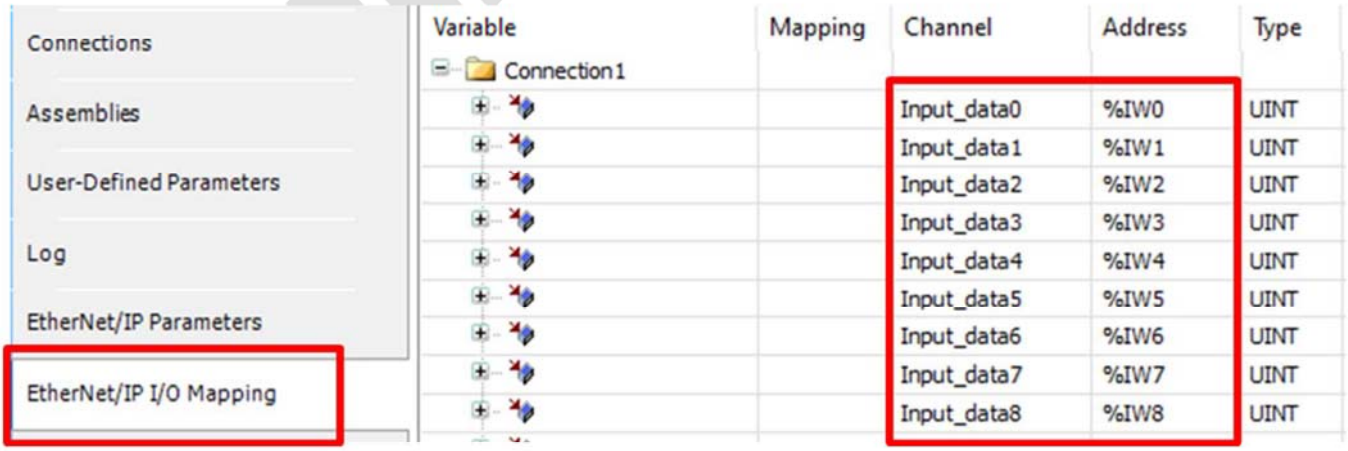
200 байтов от Ведомого (O → T) и 200 к Ведомому (T → O)

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	Application	Type	Value
%IW0	MX308_E.Application	WORD	789
%IW99	MX308_E.Application	WORD	1234
%QW0	MX308_E.Application	WORD	0
%QW99	MX308_E.Application	WORD	0
AS320T.eState	MX308_E.Application	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

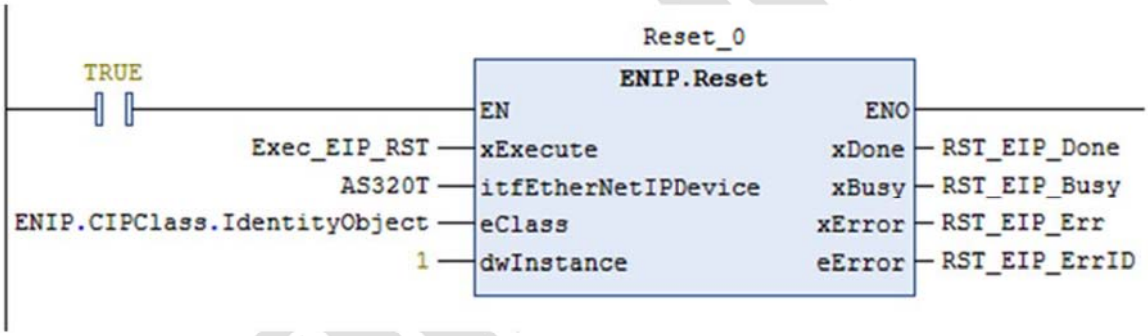
Watch 1

Expression	Application	Type	Value
%IW0	MX308_E.Application	WORD	0
%IW99	MX308_E.Application	WORD	0
%QW0	MX308_E.Application	WORD	45
%QW99	MX308_E.Application	WORD	555
AS320T.eState	MX308_E.Application	ADAPTERSTATE	RUNNING

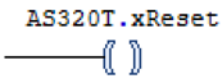
которые автоматически попадают в регистры D0 и D99 Ведомого:

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**:



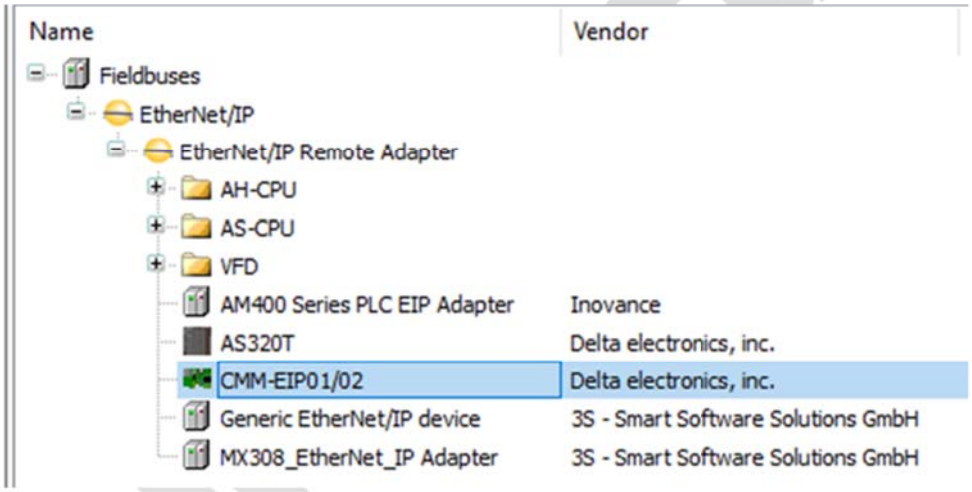
Пример настройки связи с ПЧ Delta MS300 по протоколу Ethernet/IP

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.

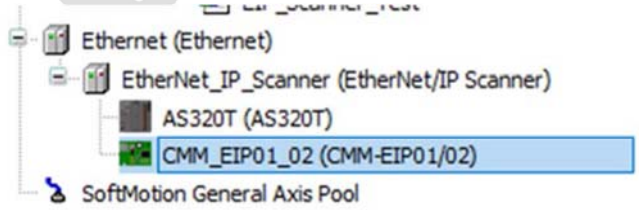
Также, необходимо убедиться, что код продукта в EDS файле на данную плату стоит 17157:

```
[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**:

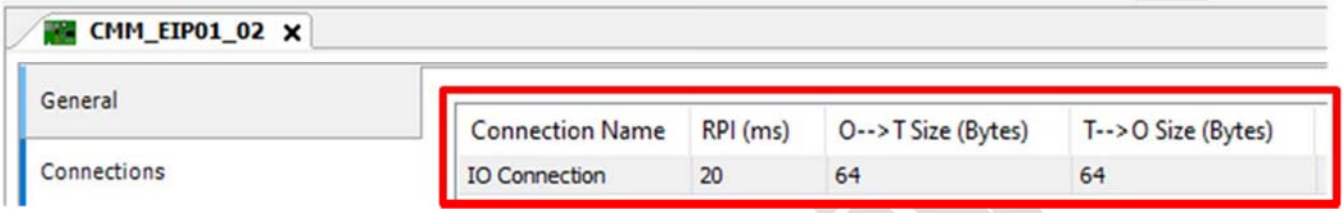


В древе проекта появится устройство:



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 (MX308) %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	

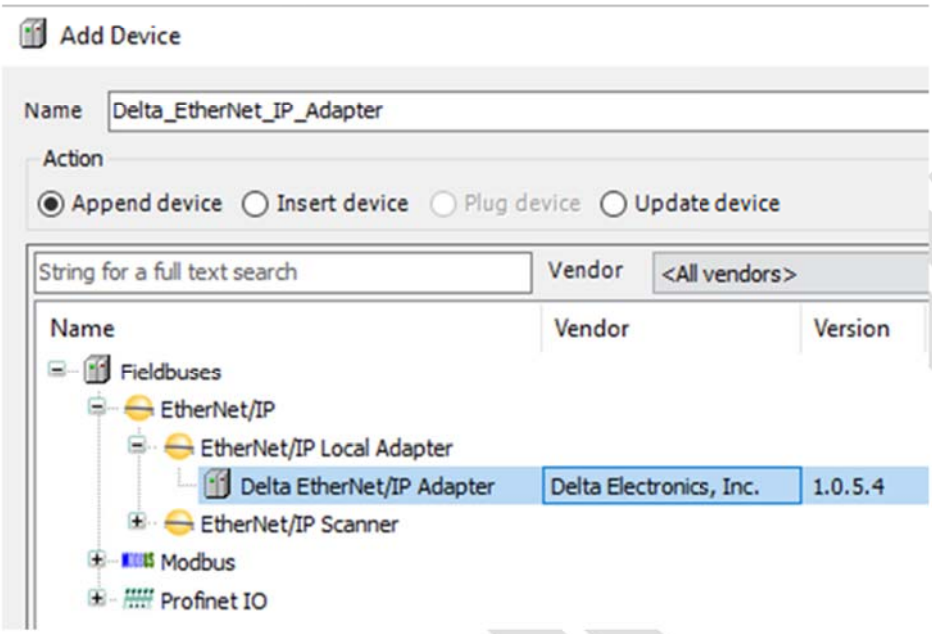
Далее регистрам можно присвоить переменные и использовать в программе контроллера для управления приводом:

CMM_EIP01_02.eState	MX308_E.Application	ADAPTERSTATE	RUNNING
CTRL_WORD	MX308_E.Application	UINT	129
FREQ_CMD	MX308_E.Application	UINT	2500
FREQ_SET	MX308_E.Application	UINT	2500
OUT_FREQ	MX308_E.Application	UINT	1154
DC_BUS	MX308_E.Application	UINT	3060
STATUS_WORD	MX308_E.Application	UINT	13571

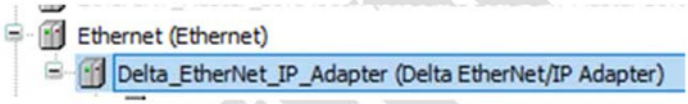
Пример настройки связи с другим контроллером CODESYS по протоколу Ethernet/IP

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 MX308, а в качестве Адаптера (Ведомого) контроллер Delta AX-308E.

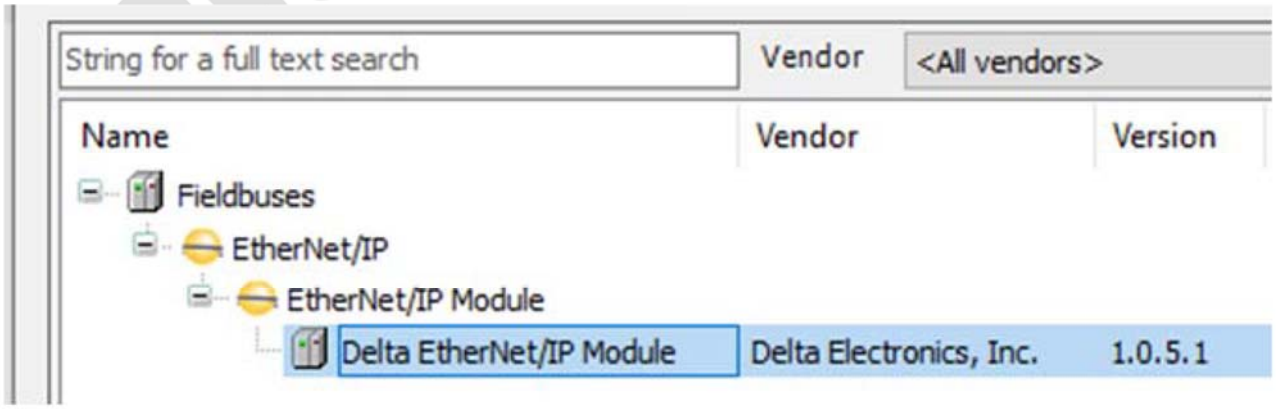
Создайте проект для контроллера Delta AX-308E. При необходимости воспользуйтесь Руководством на данный контроллер.
Добавьте в проект адаптер Ethernet. Выделите мышкой этот пункт в древе проекта и нажмите правую кнопку мышки. В появившемся меню выберите пункт **Add Device**. В открывшемся окне выберите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



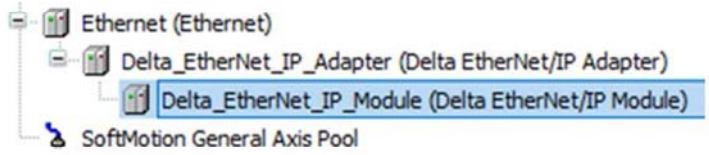
В древе проекта появится пункт:



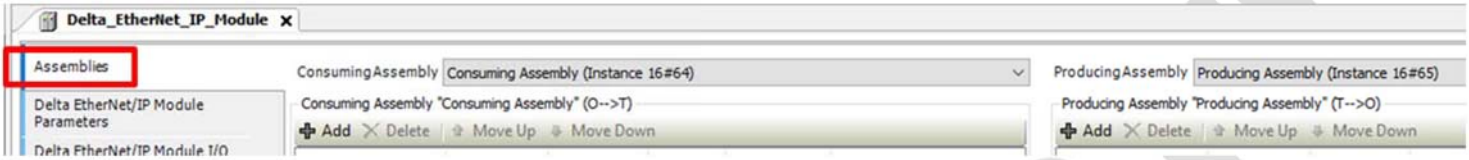
Встаньте мышкой на этот пункт и щёлкните правой кнопкой. В появившемся меню выберите пункт **Add Device**. В открывшемся окне выберите **Delta Ethernet/IP Module**:



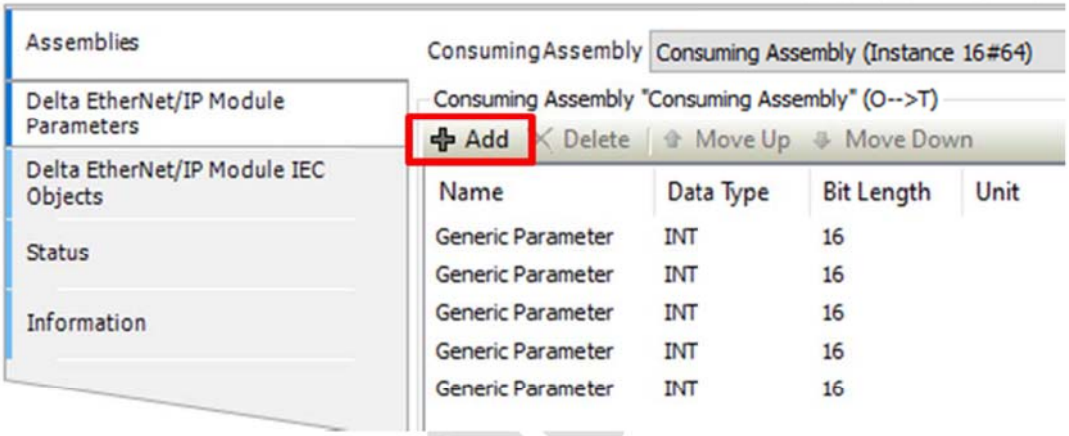
В древе проекта появится пункт:



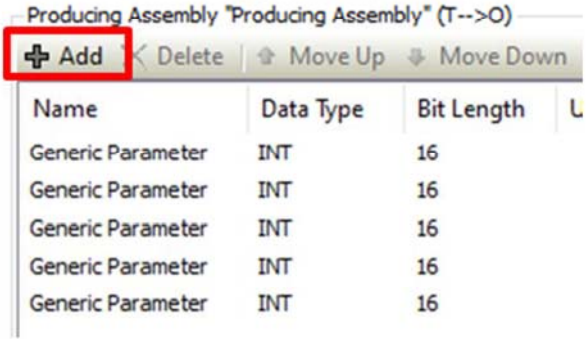
Щёлкните на этом пункте дважды левой кнопкой мышки и в отрывшейся вкладке выберите раздел **Assemblies**:



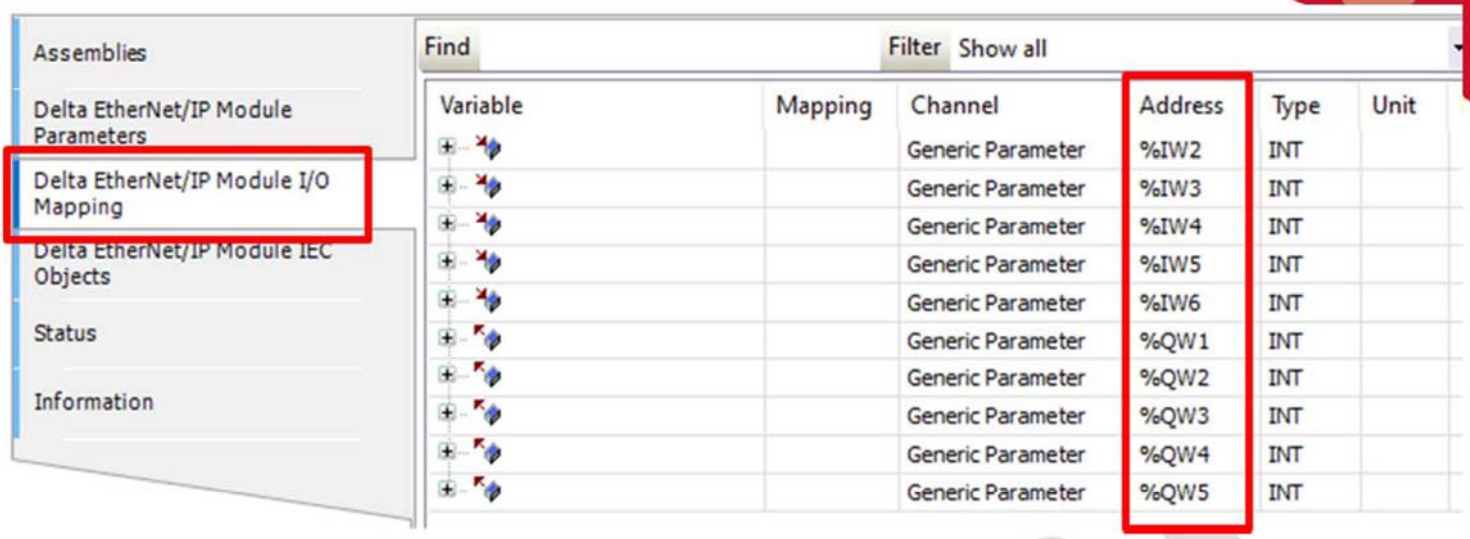
В данном разделе создаются пакеты для отправки и приёма от Мастера. Для добавления регистров в пакет на приём от Мастера нажимайте кнопку **Add** в поле **Consuming Assembly**:



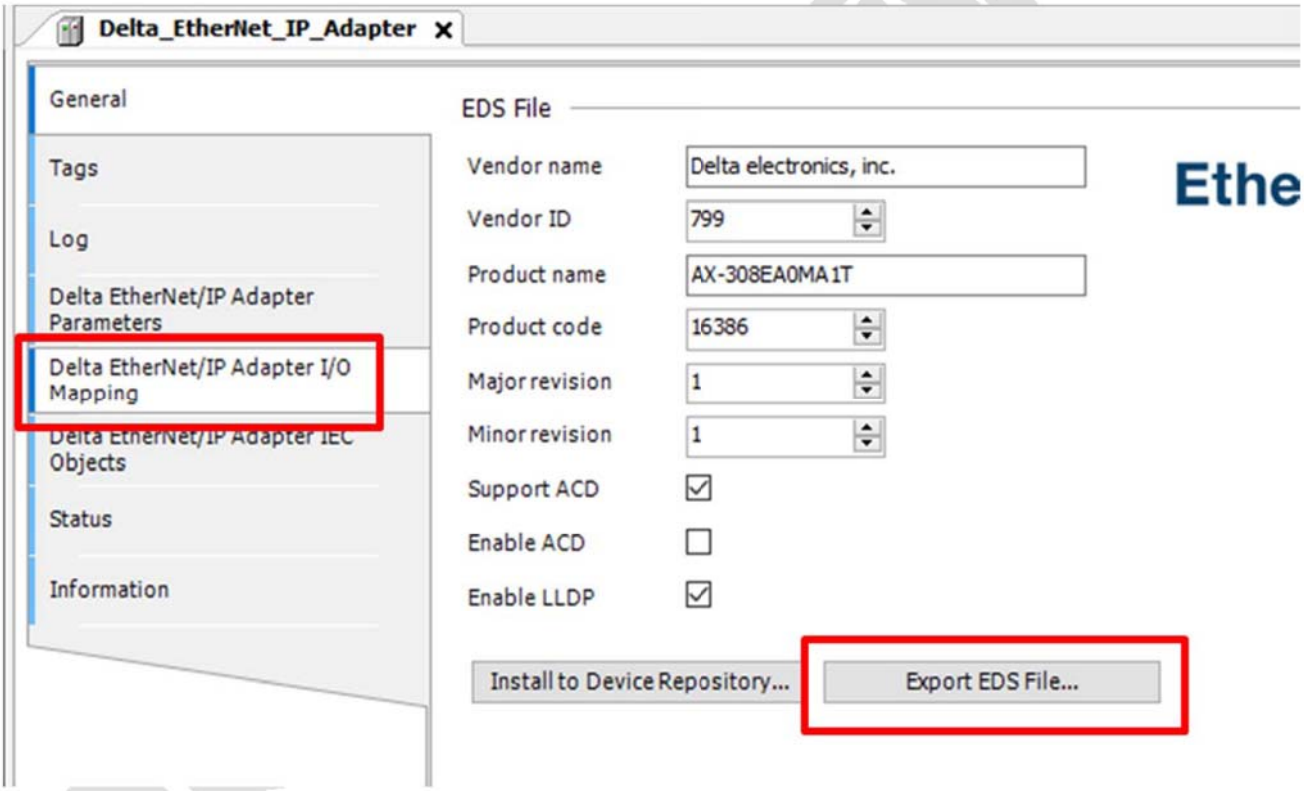
Для добавления регистров в пакет на отправку Мастеру нажимайте кнопку **Add** в поле **Producing Assembly**:



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.

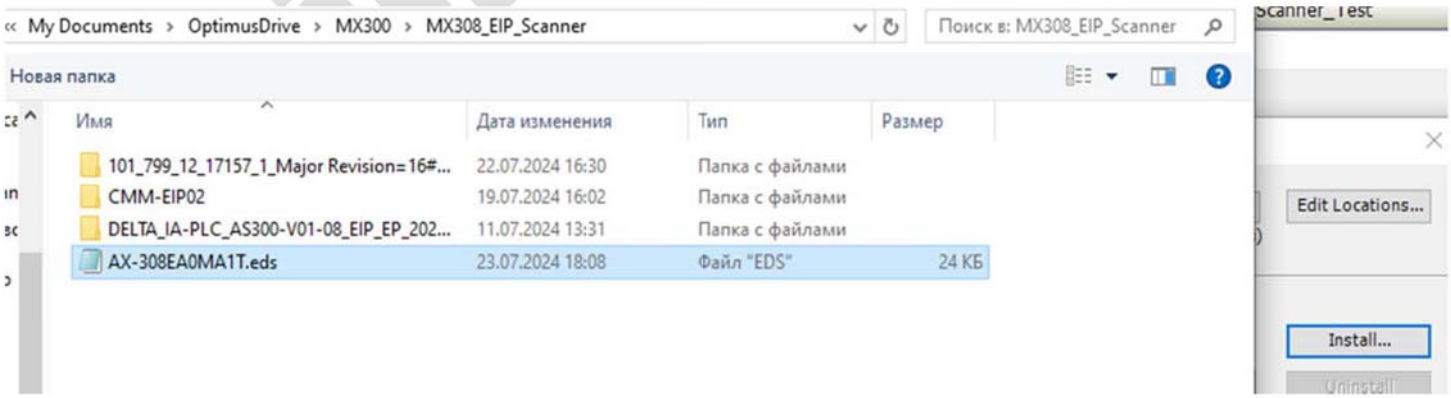
Для изменения поля откройте файл в Блокноте (Notepad) и измените название поле **ProdName**

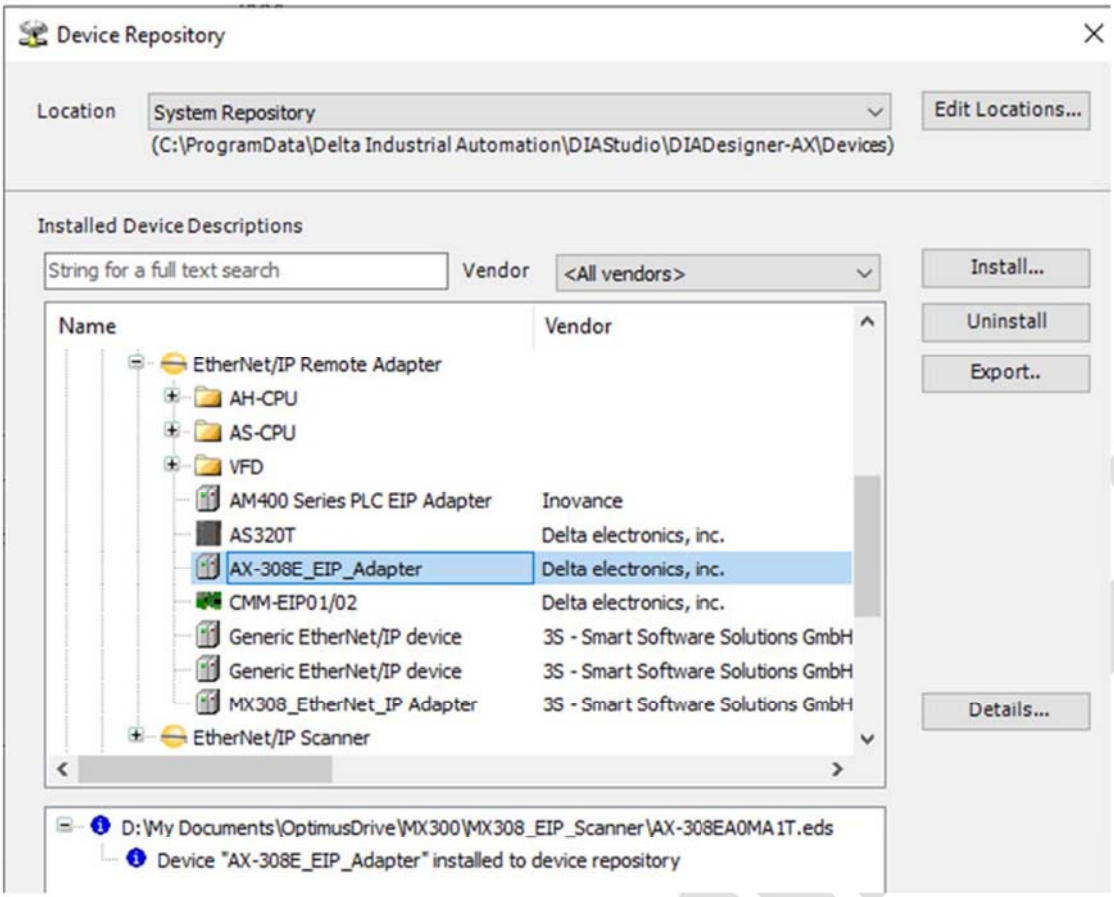
```
[Device]
        VendCode = 799;
        VendName = "Delta electronics, inc.";
        ProdType = 12;
        ProdTypeStr = "Communications Adapter";
        ProdCode = 16386;
        MajRev = 1;
        MinRev = 1;
        ProdName = "AX-308EA0MA1T";
```

например на такое:

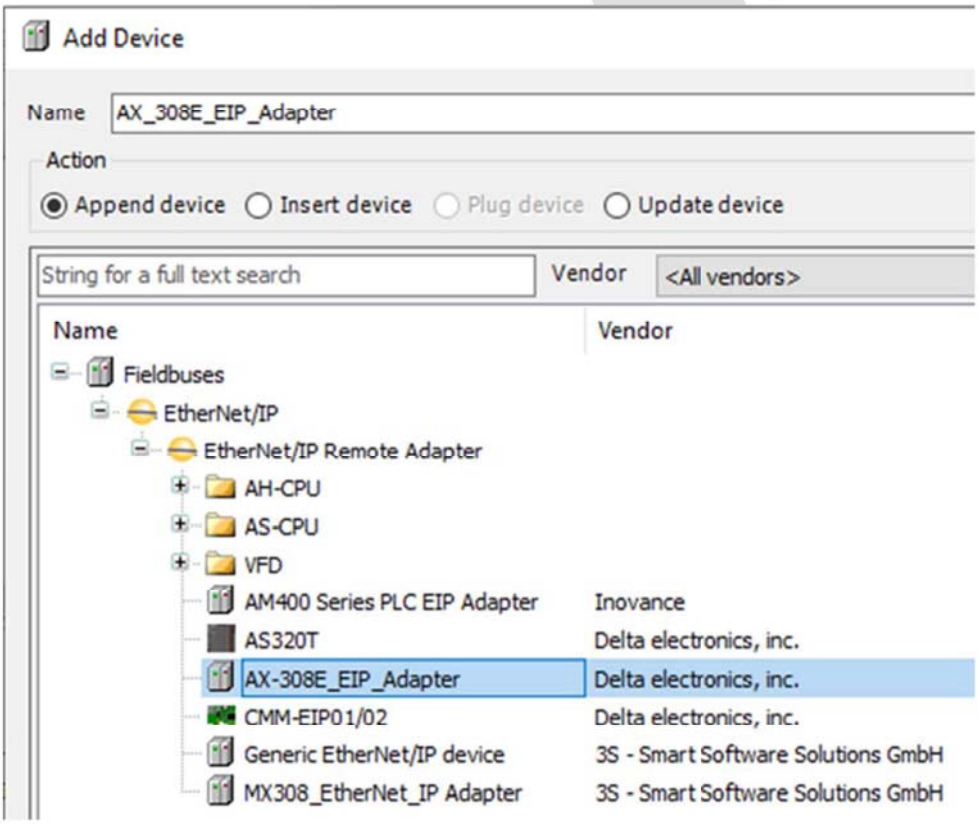
```
[Device]
        VendCode = 799;
        VendName = "Delta electronics, inc.";
        ProdType = 12;
        ProdTypeStr = "Communications Adapter";
        ProdCode = 16386;
        MajRev = 1;
        MinRev = 1;
        ProdName = "AX-308E_EIP_Adapter";
```

Далее файл необходимо импортировать в репозитарий устройств Мастера (MX300). Т.е. там появится устройство типа Adapter Ethernet/IP. Данная процедура, как и создание проекта для контроллеров MX300, описаны в соответствующих Главах настоящего Руководства.

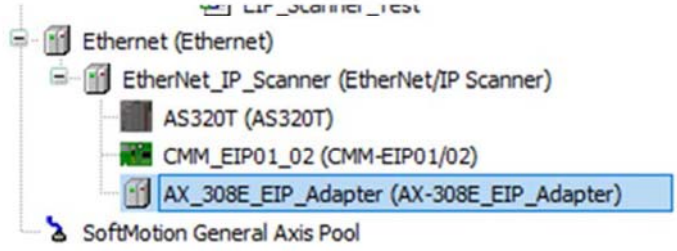




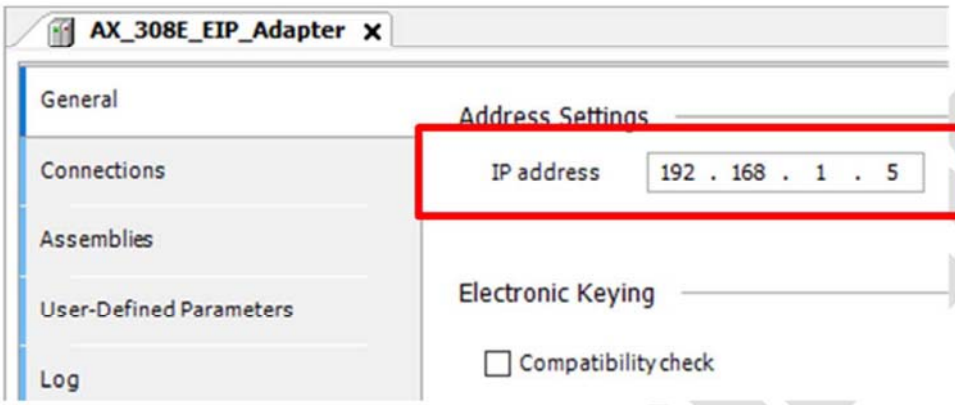
Добавьте адаптер в проект Мастера:



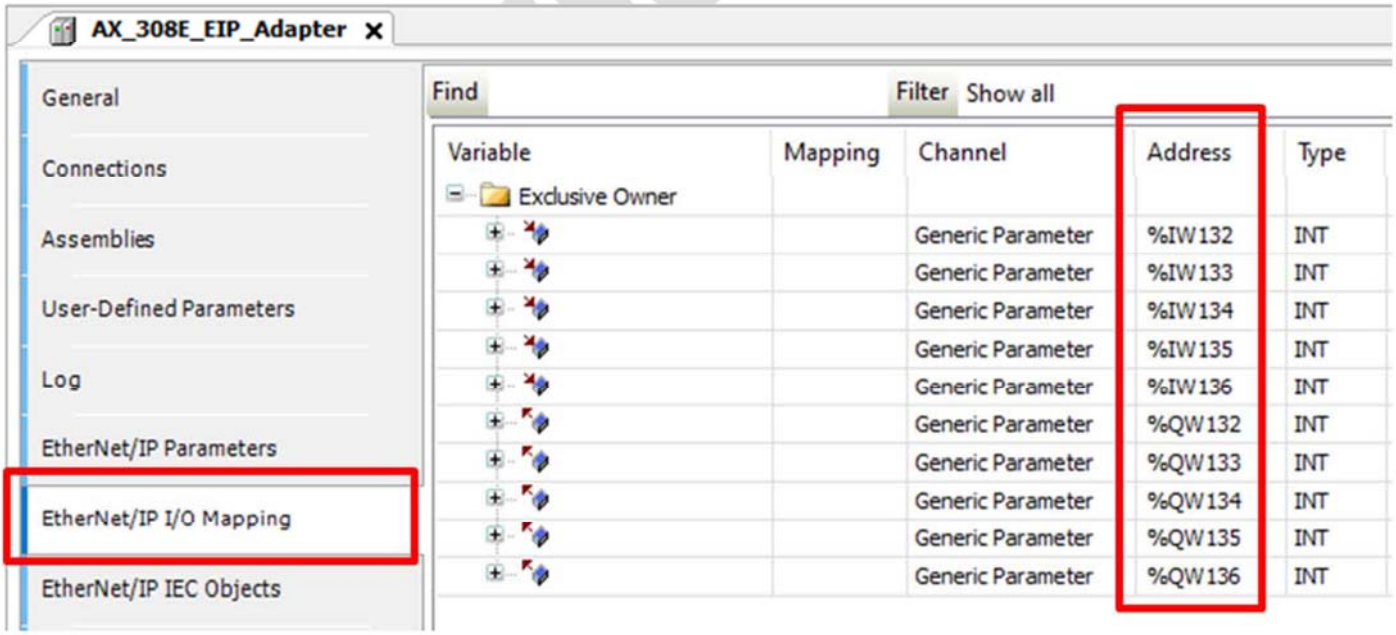
В древе проекта появится устройство:



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 MX308) в регистре %QW132 задать число 45:

%IW132	MX308_E.Application	WORD	0
%QW132	MX308_E.Application	WORD	45
AX_308E_EIP_Adapter.eState	MX308_E.Application	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 (MX308) %IW132 появится число 555:

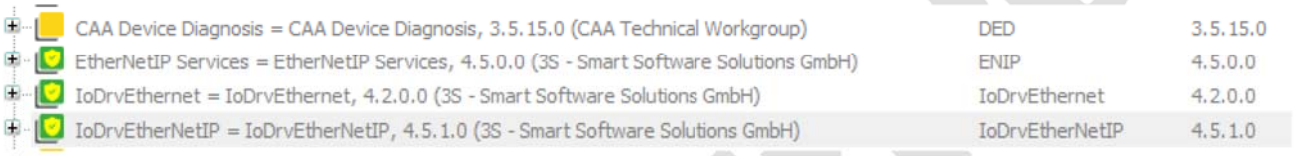
%IW132	MX308_E.Application	WORD	555
%QW132	MX308_E.Application	WORD	0
AX_308E_EIP_Adapter.eState	MX308_E.Application	ADAPTERSTATE	RUNNING

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**.

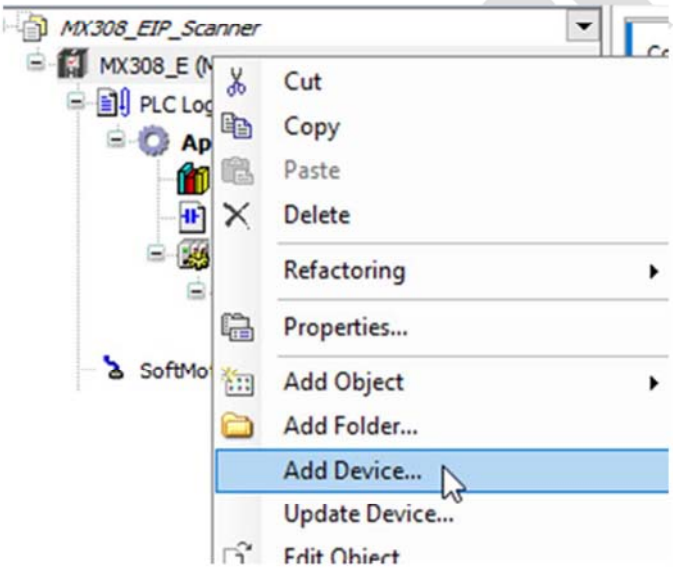
Контроллеры серии MX300 могут работать как в режиме **Ethernet/IP Scanner** (Master), так и в режиме **Ethernet/IP Adapter** (Slave). В данной главе рассматривается организация связи контроллера MX300 с Мастером в режиме Ethernet/IP Adapter (Slave).

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

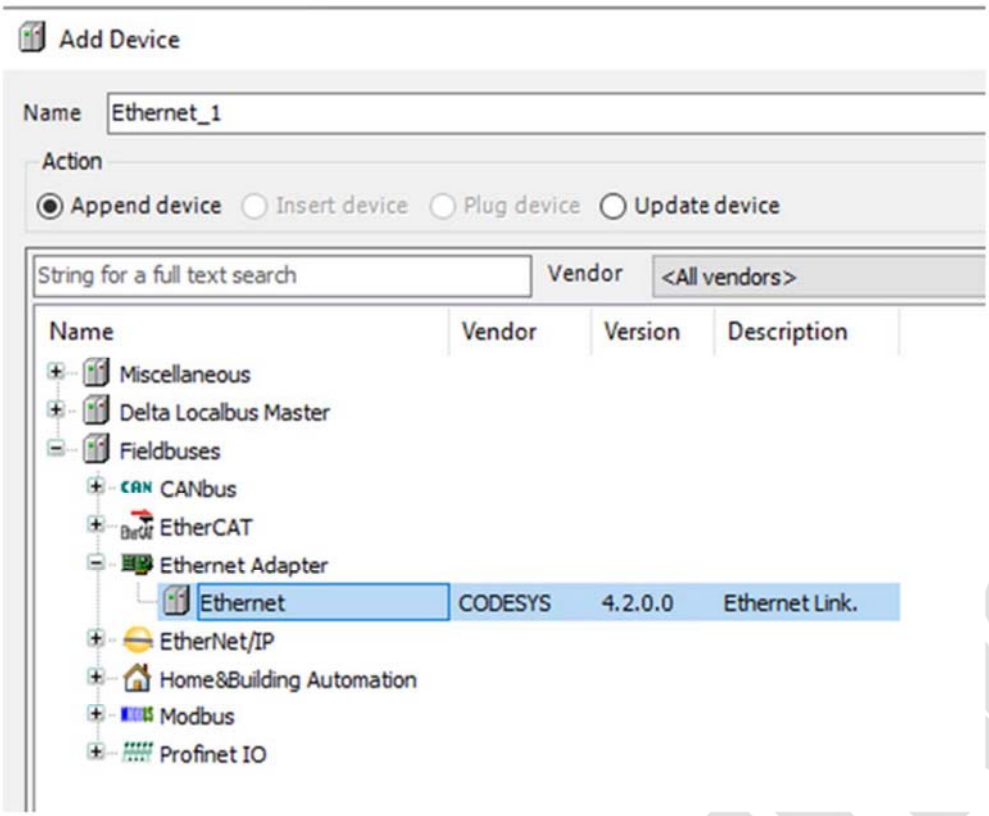


Настройка связи по протоколу Ethernet/IP начинается с добавления к проекту адаптера Ethernet и привязка его к определённому порту.

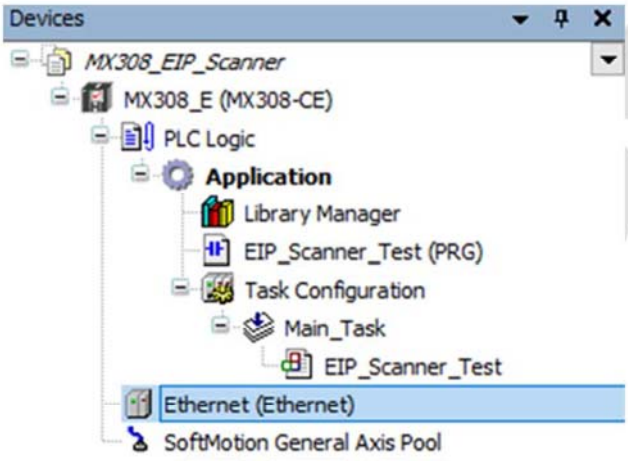
Щёлкните в древе правой кнопкой мышки на пункте **Device** (MX308_E) и в отрывшемся меню выберите пункт **Add Device**



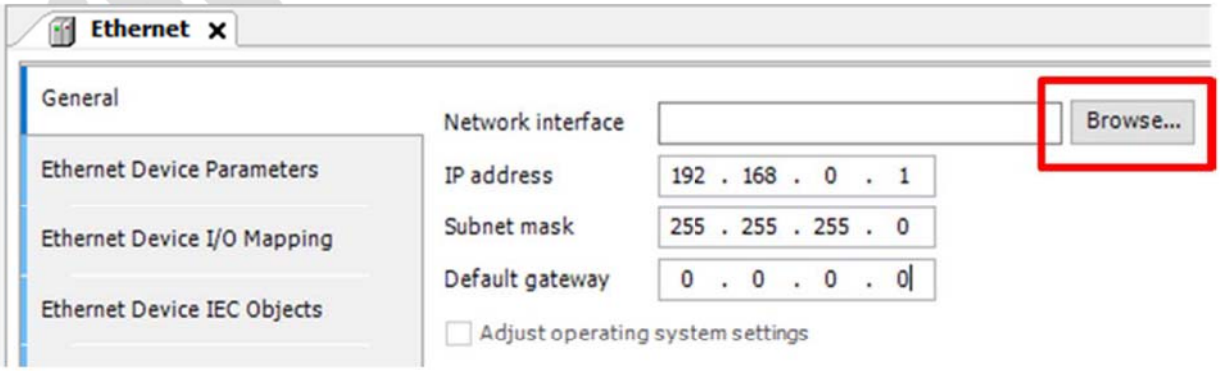
В открывшемся окне выберите **Fieldbuses – Ethernet**:



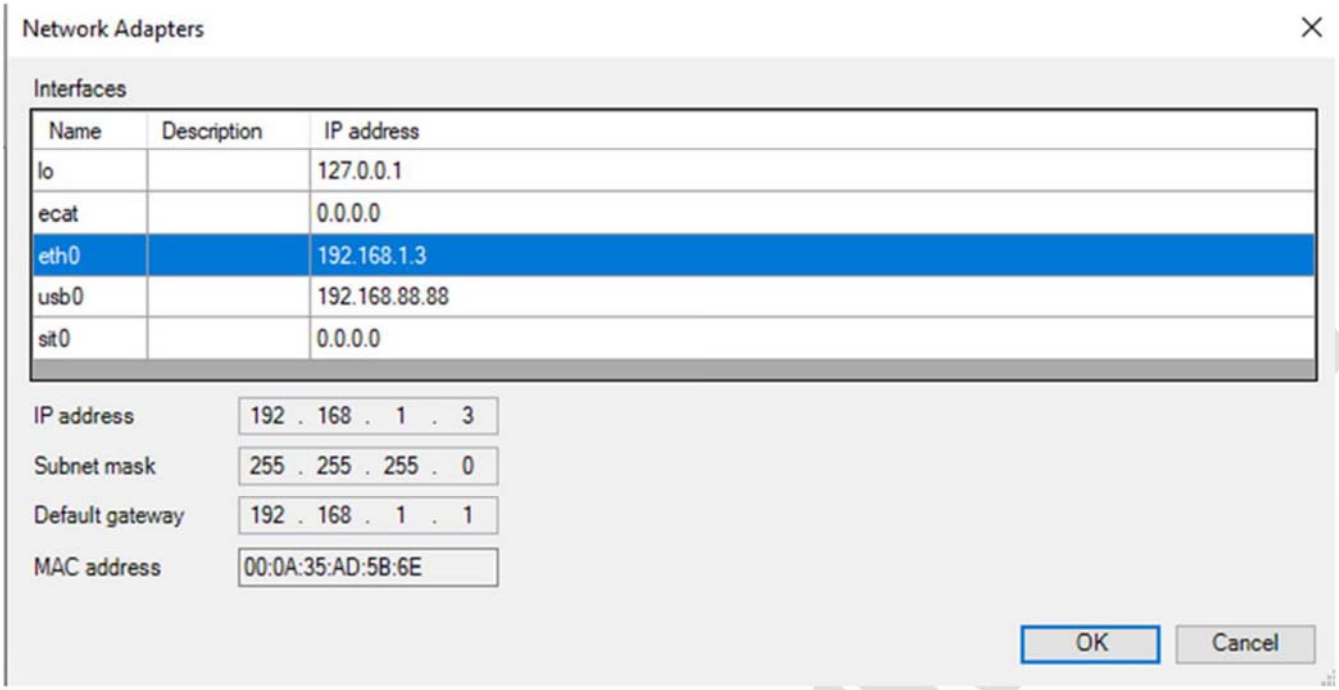
В древе проекта появится пункт **Ethernet**:



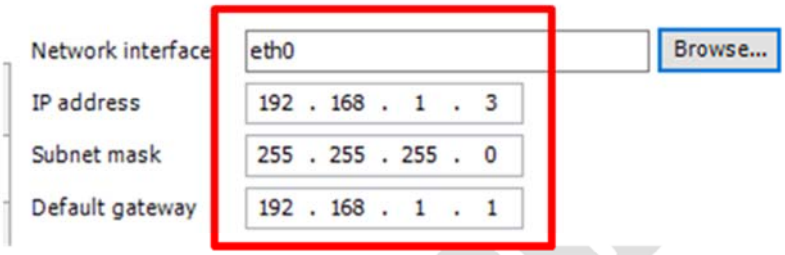
Щёлкните на нём дважды левой кнопкой мышки и в открывшейся вкладке выберите пункт **General**:



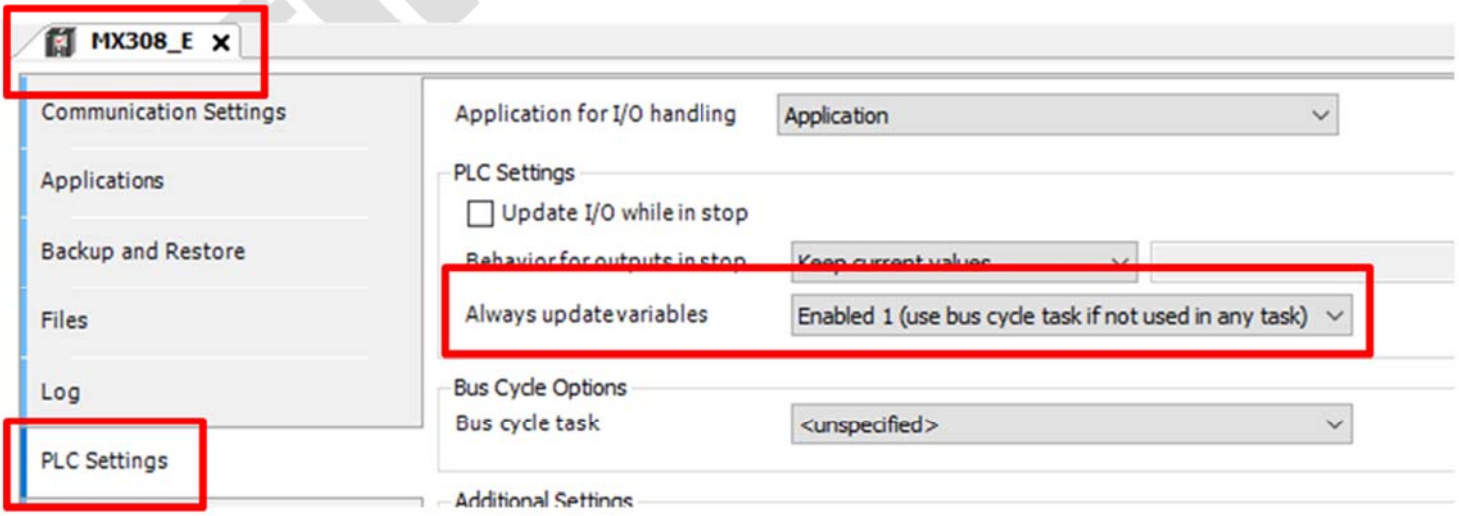
Нажмите кнопку **Browse** и в открывшемся окне выберите IP адрес контроллера:



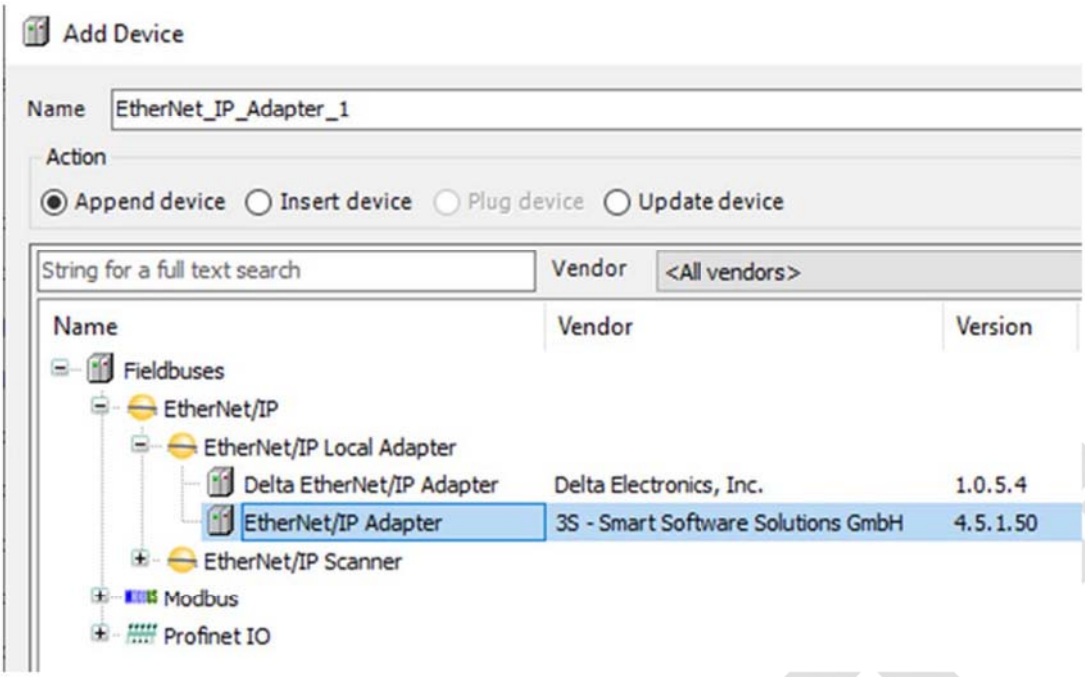
После чего Адаптер будет привязан к Ethernet порту с конкретным IP адресом:



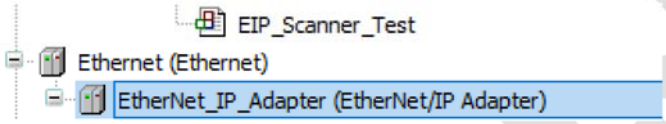
Далее двойным щелчком левой кнопки мыши откройте вкладку **Device** и в пункте **PLC Settings** разрешите обновление переменных. Данный шаг позволяет видеть изменение переменных и регистров без написания в теле программы, т.е. в таблице Watch и в таблицах мэпинга.



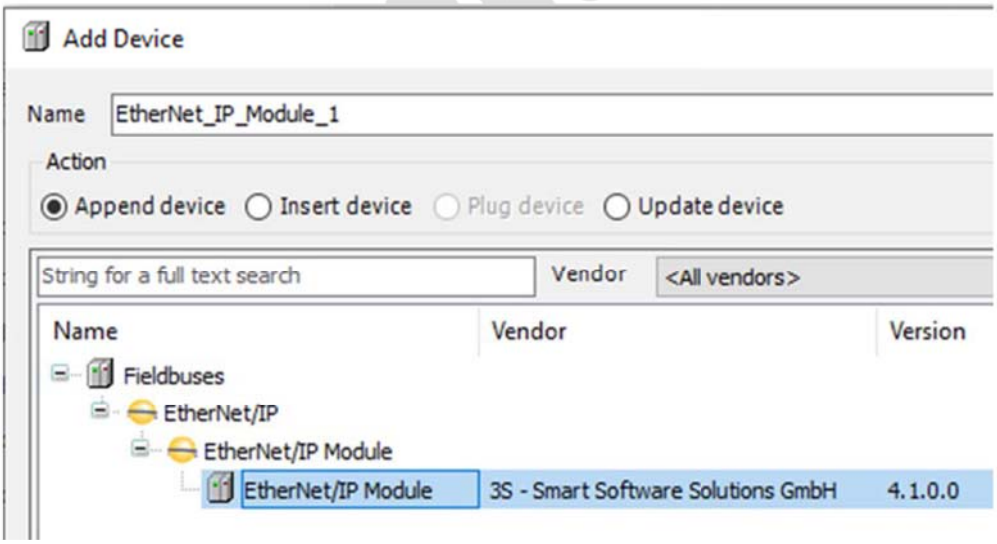
Щёлкните в древе правой кнопкой мышки на пункте **Ethernet** и в отрывшемся меню выберите пункт **Add Device**. Далее в открывшемся окне выберите пункт **Fieldbuses – Ethernet/IP Adapter**:



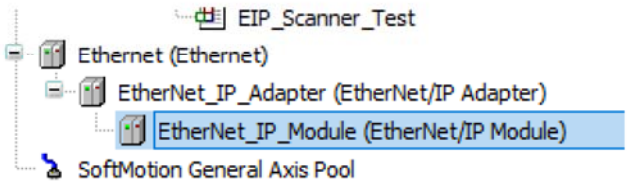
В древе проекта появится устройство:



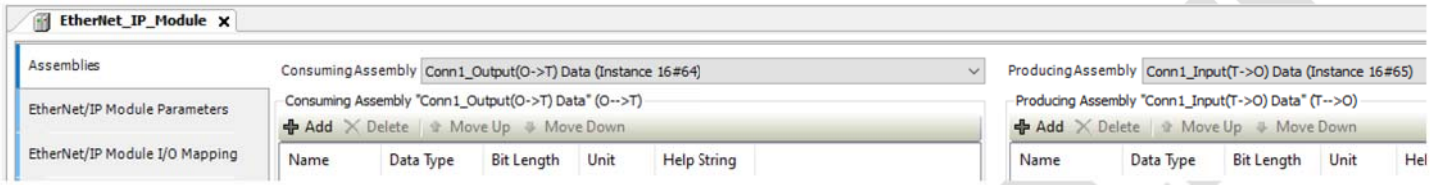
Щёлкните правой кнопкой мышки на данном пункте и добавьте устройство типа **Ethernet/IP Module**:



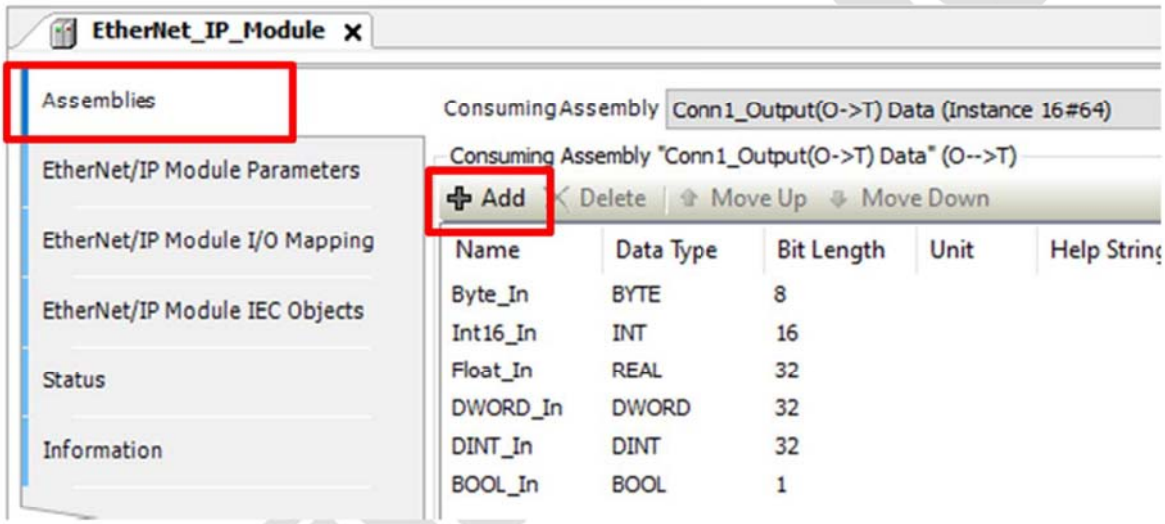
В древе проекта появится устройство:



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

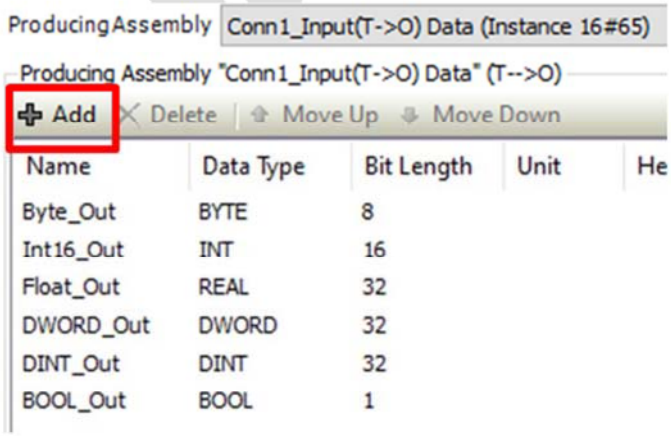


В данном разделе создаются пакеты для отправки и приёма от Мастера. Для добавления регистров в пакет на приём от Мастера нажимайте кнопку **Add** в поле **Consuming Assembly**:

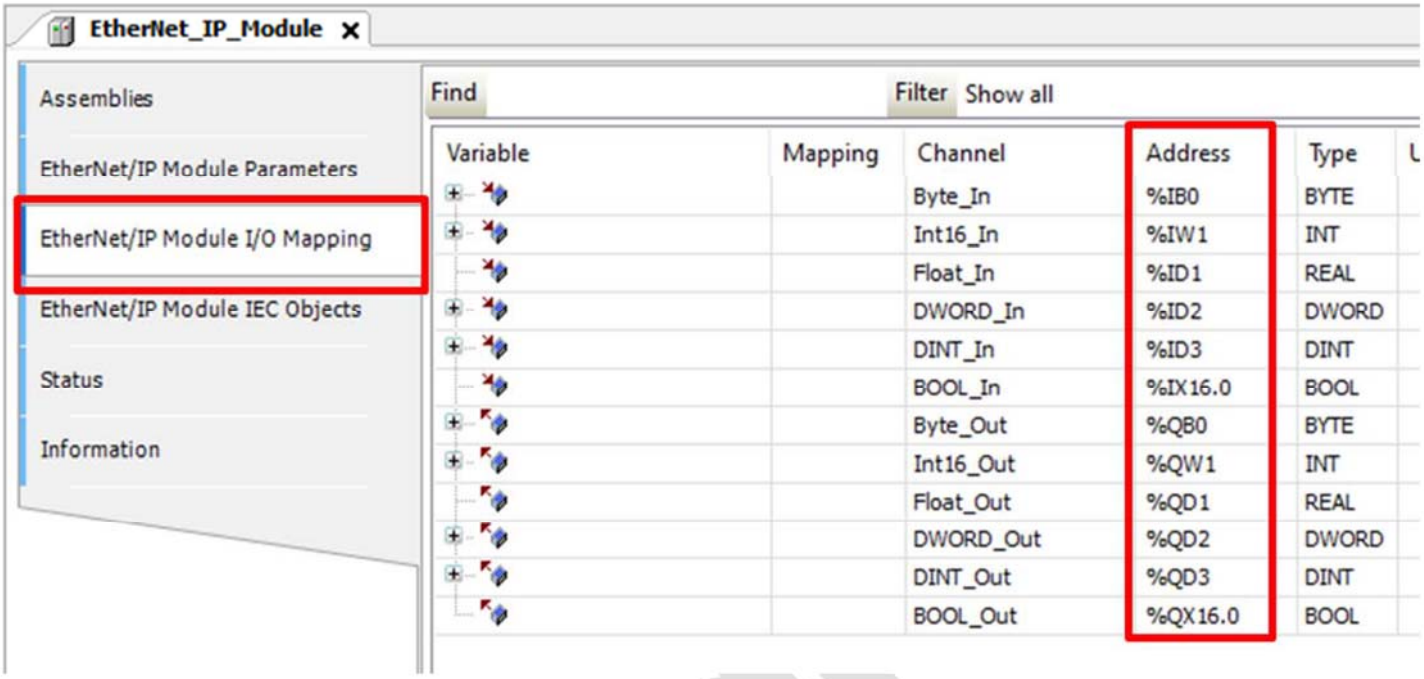


Теги можно создавать сразу с названием и любого из поддерживаемых типов данных.

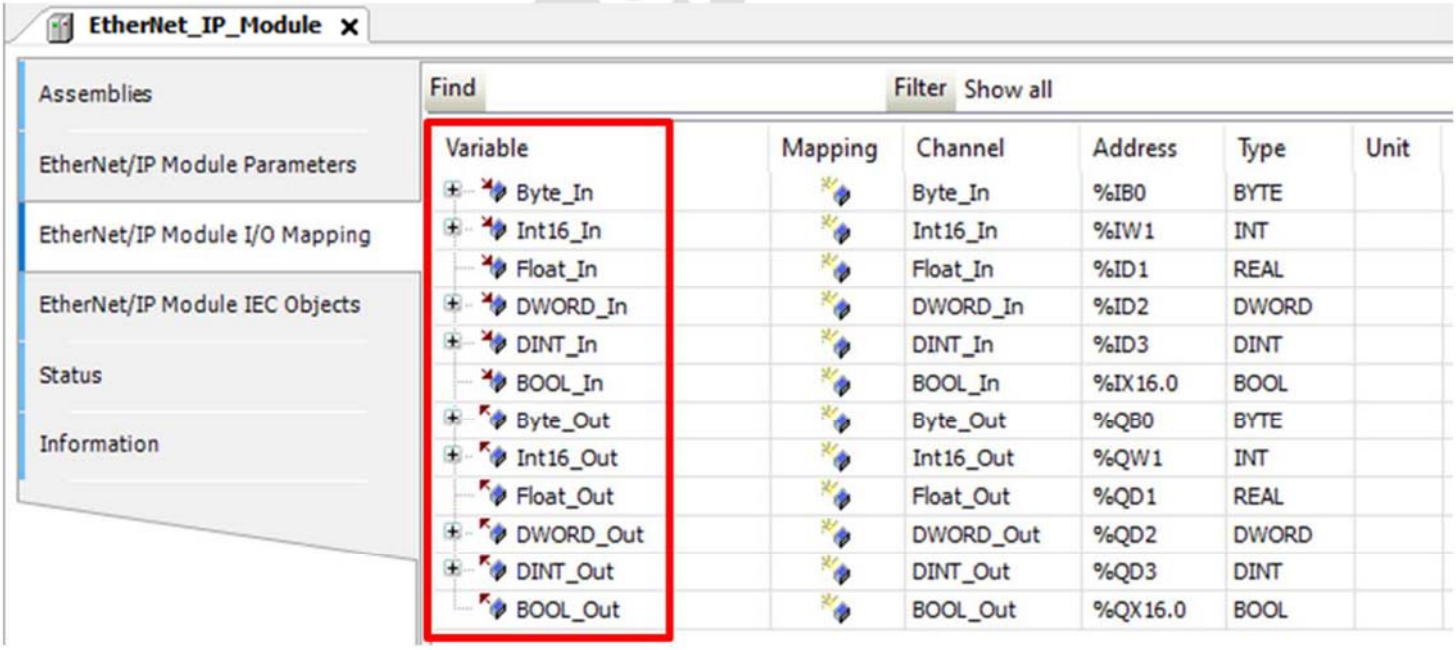
Для добавления регистров в пакет на отправку Мастеру нажимайте кнопку **Add** в поле **Producing Assembly**:



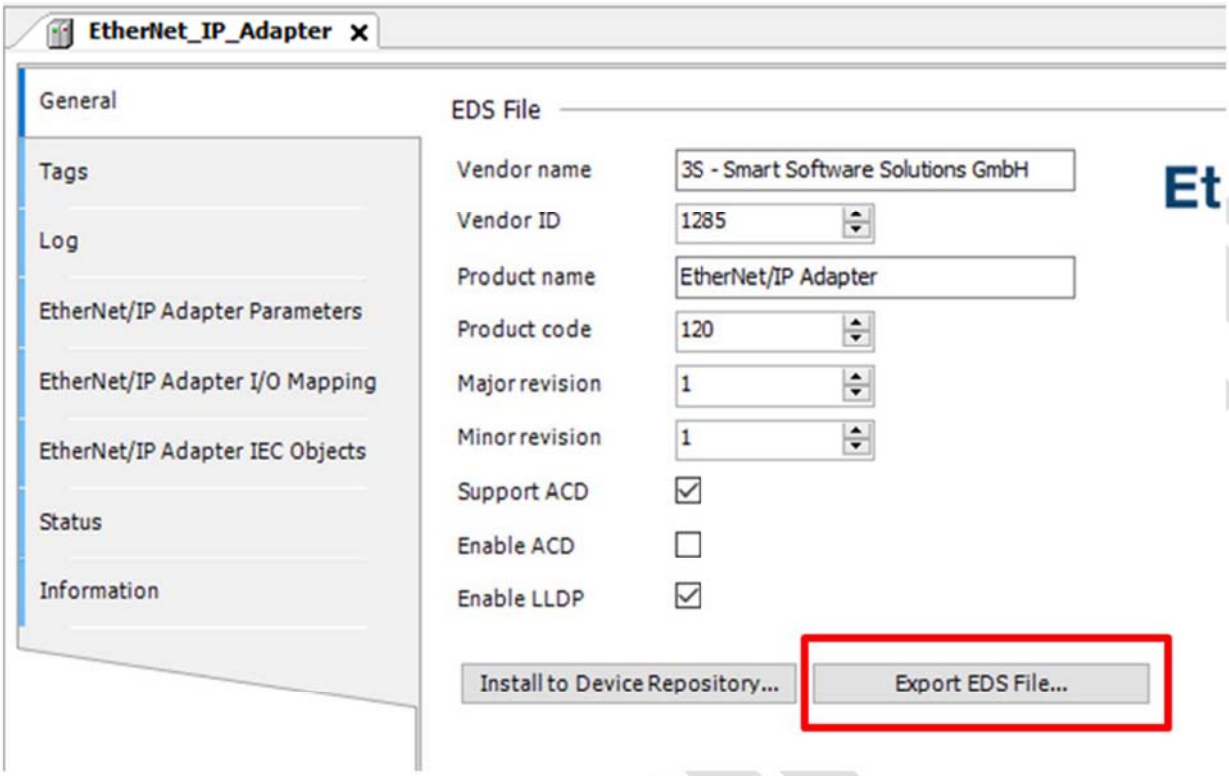
Общий список регистров можно увидеть в разделе **Ethernet/IP Module I/O Mapping**. Здесь же можно присвоить регистрам теги. Регистры типа %I** будут содержать данные от Мастера (Consuming Tags), а из регистров типа %Q** данные будут передаваться Мастеру (Producing Tags).



Для удобства лучше сразу присвоить тегам имена, которые можно будет использовать непосредственно в программе как Ведомого устройства (MX300):



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.

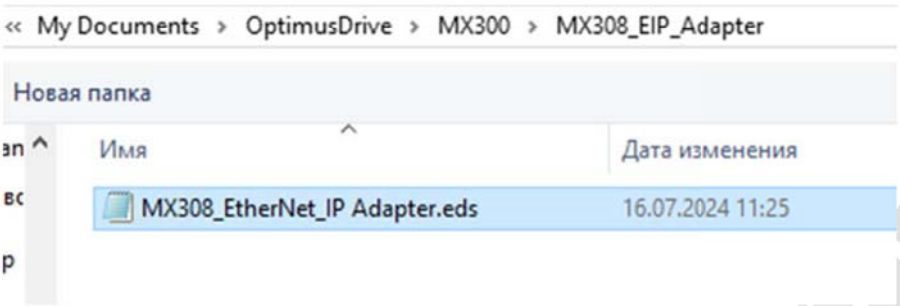
Для изменения поля откройте файл в Блокноте (Notepad) и измените название поле **ProdName**

```
[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:

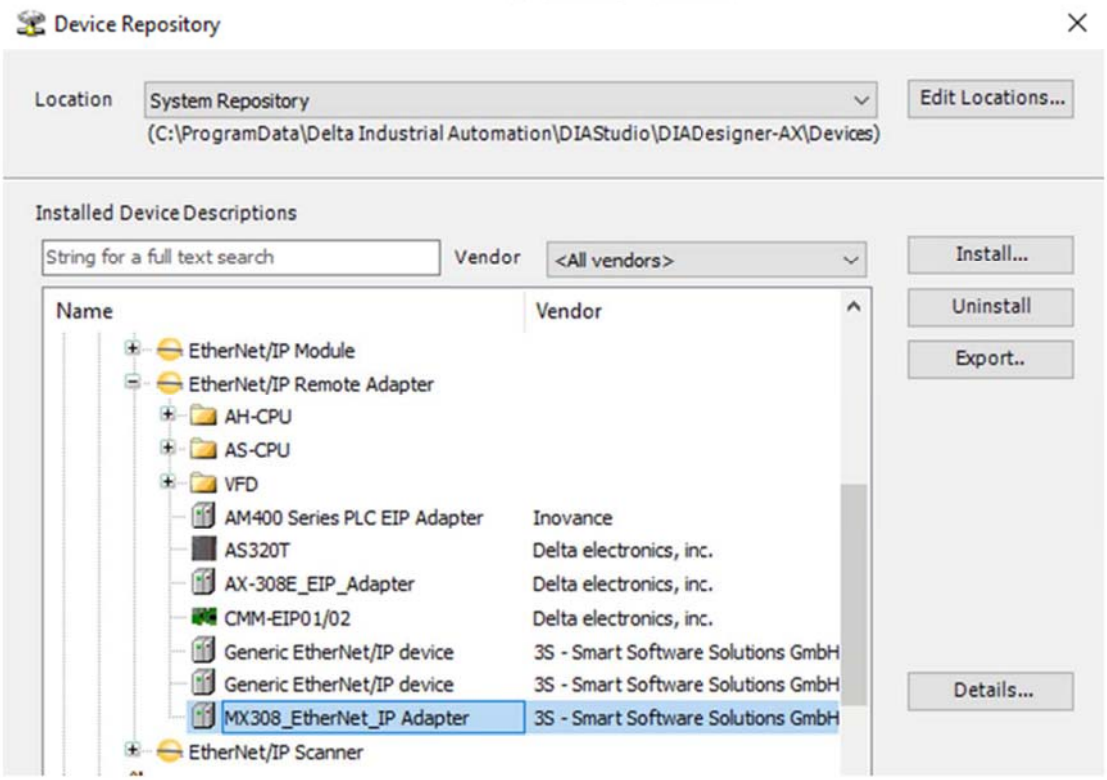
```
[Device]
        VendCode = 1285;
        VendName = "3S - Smart Software Solutions GmbH";
        ProdType = 12;
        ProdTypeStr = "Communications Adapter";
        ProdCode = 120;
        MajRev = 1;
        MinRev = 1;
        ProdName = "MX308_EtherNet_IP Adapter";
```

И можно изменить имя файла:

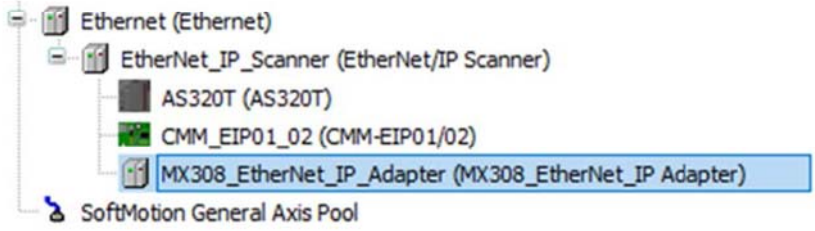


*Данный файл можно импортировать в любой **Ethernet/IP Scanner (Master).***

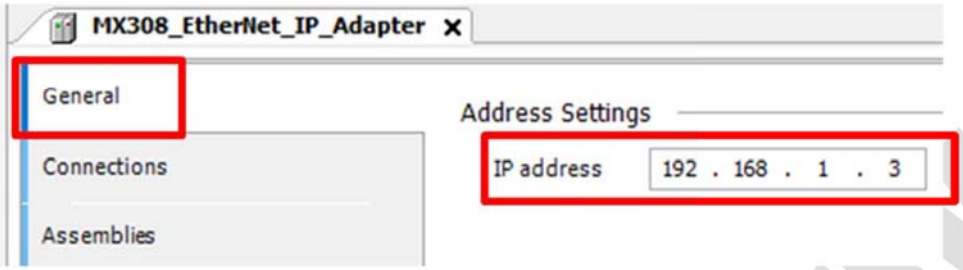
В качестве **Ethernet/IP Scanner (Master)** рассмотрим контроллер Delta AX-308E.
Создайте проект. При необходимости используйте документацию на данный контроллер.
Импортируйте в репозиторий устройств полученный на предыдущем шаге XML файл с тегами от контроллера MX308 (Adapter).



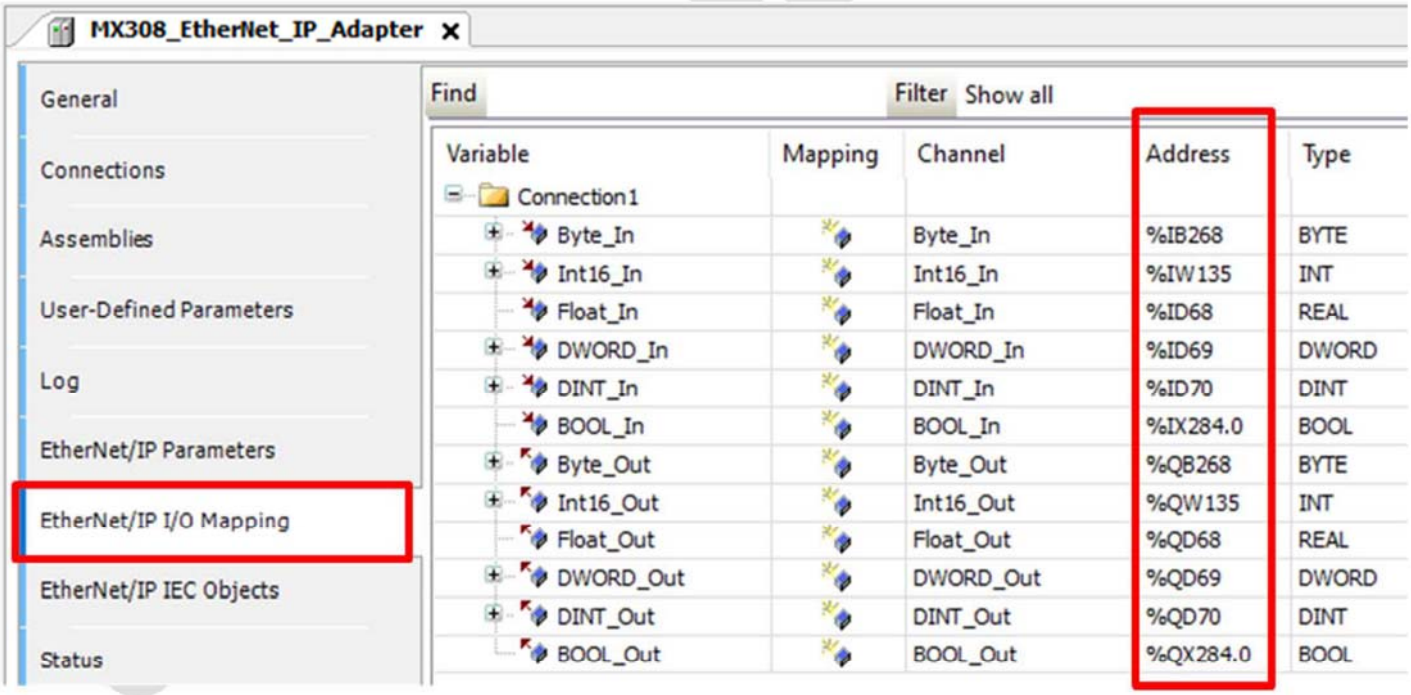
Добавьте устройство в проект:



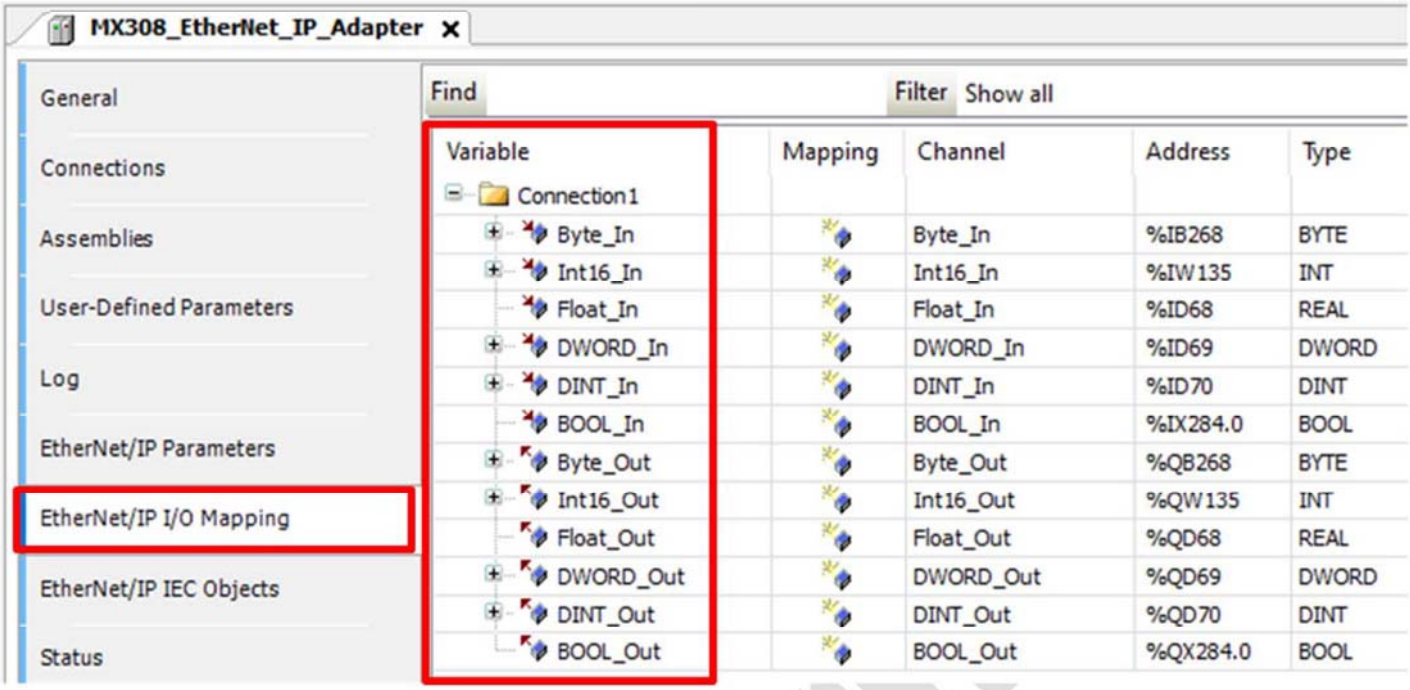
Щёлкните на нём двойным щелчком левой кнопкой мышки и в отрывшейся вкладке выберите пункт **General**, где необходимо установить IP адрес Адаптера:



Система автоматически раздаст адреса тегам, которые содержались в импортированном XML файле Адаптера (контроллера MX300). 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:



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

MX308_EtherNet_IP_Adapter.eState

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

Watch 1

Expression	Application	Type	Value
EtherNet_IP_Module.eState	MX308_E.Application	MODULESTATE	RUNNING
IoConfig_Globals_Mapping.Byte_In	MX308_E.Application	BYTE	255
IoConfig_Globals_Mapping.Int16_In	MX308_E.Application	INT	32767
IoConfig_Globals_Mapping.Float_In	MX308_E.Application	REAL	1234.567
IoConfig_Globals_Mapping.DWORD_In	MX308_E.Application	DWORD	4022333445
IoConfig_Globals_Mapping.DINT_In	MX308_E.Application	DINT	1022333445
IoConfig_Globals_Mapping.BOOL_In	MX308_E.Application	BIT	TRUE
IoConfig_Globals_Mapping.Byte_Out	MX308_E.Application	BYTE	0
IoConfig_Globals_Mapping.Int16_Out	MX308_E.Application	INT	0
IoConfig_Globals_Mapping.Float_Out	MX308_E.Application	REAL	0
IoConfig_Globals_Mapping.DWORD_Out	MX308_E.Application	DWORD	0
IoConfig_Globals_Mapping.DINT_Out	MX308_E.Application	DINT	0
IoConfig_Globals_Mapping.BOOL_Out	MX308_E.Application	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 задать значения для исходящих тегов (Produced Tags):

Watch 1

Expression	Application	Type	Value
EtherNet_IP_Module.eState	MX308_E.Application	MODULESTATE	RUNNING
IoConfig_Globals_Mapping.Byte_In	MX308_E.Application	BYTE	0
IoConfig_Globals_Mapping.Int16_In	MX308_E.Application	INT	0
IoConfig_Globals_Mapping.Float_In	MX308_E.Application	REAL	0
IoConfig_Globals_Mapping.DWORD_In	MX308_E.Application	DWORD	0
IoConfig_Globals_Mapping.DINT_In	MX308_E.Application	DINT	0
IoConfig_Globals_Mapping.BOOL_In	MX308_E.Application	BIT	FALSE
IoConfig_Globals_Mapping.Byte_Out	MX308_E.Application	BYTE	127
IoConfig_Globals_Mapping.Int16_Out	MX308_E.Application	INT	-32768
IoConfig_Globals_Mapping.Float_Out	MX308_E.Application	REAL	-9876.321
IoConfig_Globals_Mapping.DWORD_Out	MX308_E.Application	DWORD	4000222333
IoConfig_Globals_Mapping.DINT_Out	MX308_E.Application	DINT	-20123555
IoConfig_Globals_Mapping.BOOL_Out	MX308_E.Application	BIT	TRUE

то данные автоматически попадут в регистры Мастера:

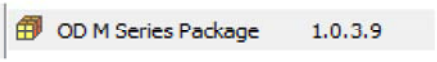
Watch 1

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

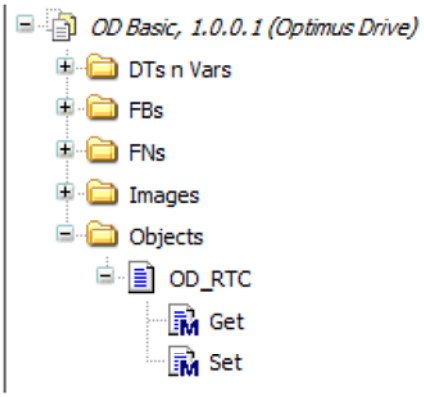
Контроллеры семейства MX300 имеют встроенные часы реального времени (RTC). Для поддержки работы часов используется специальная модификация батарейки CR2032 (с выводом проводов), установленная в отсеке в корпусе контроллера.

В составе базового пэкиджа с описанием устройства

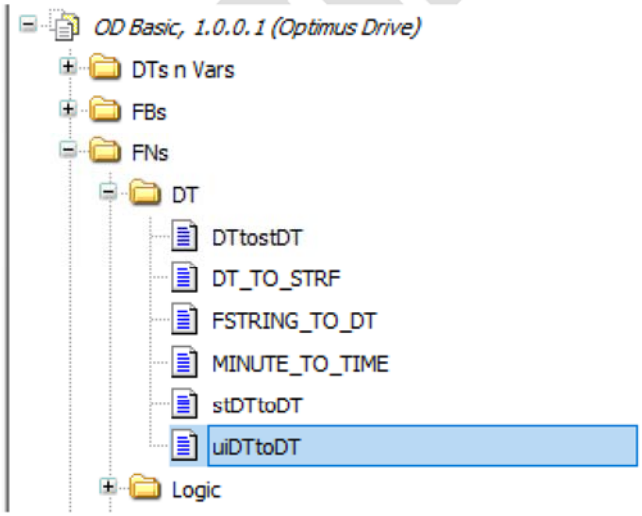


ставится библиотека **OD Basic**:

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

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
```

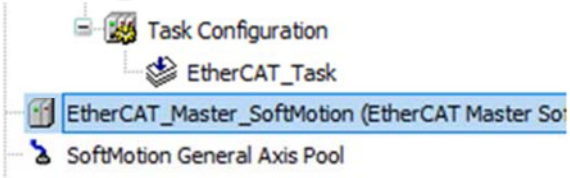
Примечание:
Минимальный год, который можно выставить – 2010 (1 января 00 00)

Работа со станцией удалённого ввода-вывода R2EC в сети EtherCAT

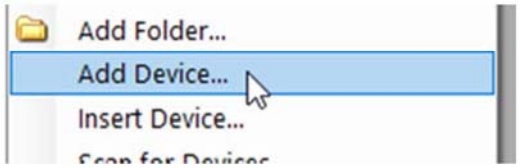
Станция (каплер) **R2EC** позволяет удалённо использовать модули ввода-вывода от контроллера MX300. Физически модули подсоединяются к станции также как к ЦПУ. Далее станция соединяется с ЦПУ кабелем EtherCAT.
В проект контроллера добавляется устройства типа R2EC и уже на него добавляются модули ввода-вывода. Обращение к модулям идёт через станцию, т.е. сетевой адрес есть только у станции. Модули находятся на её внутренней шине и не имеют прямого выхода в сеть.
Технология удалённых станций позволяет использовать дискретные и аналоговые сигналы с вынесенных от ЦПУ модулей ввода-вывода как если бы они были непосредственно на ЦПУ. Т.е. сразу через переменные в программе, без коммуникационных запросов.

Рассмотрим порядок работы с удалёнными станциями. Сначала создайте проект и добавьте в него устройство типа ЦПУ MX300 и адаптер EtherCAT, как описано в соответствующих Разделах данного Руководства.

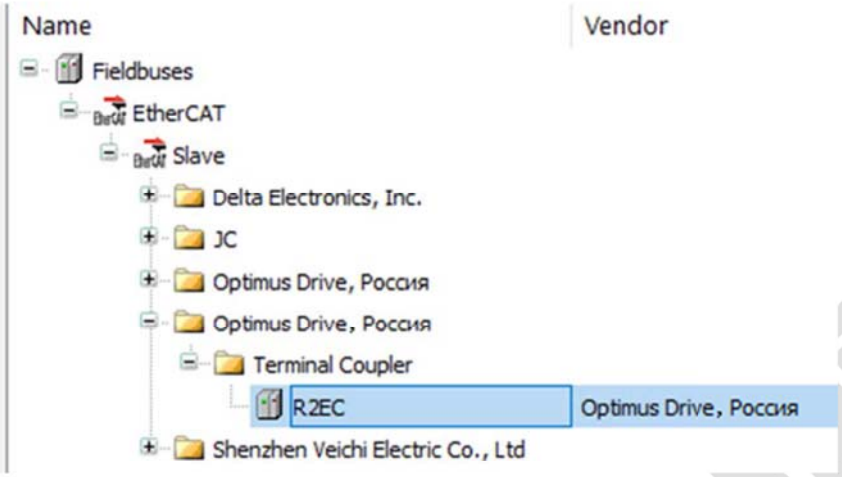
Далее выберите в древе проекта пункт EtherCAT Master и правой кнопкой мышки вызовите меню :



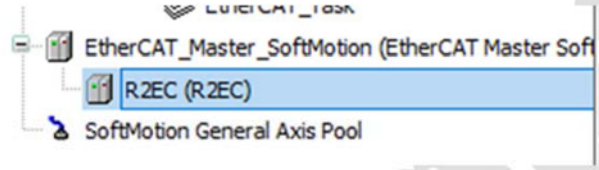
В открывшемся окне выберите пункт :



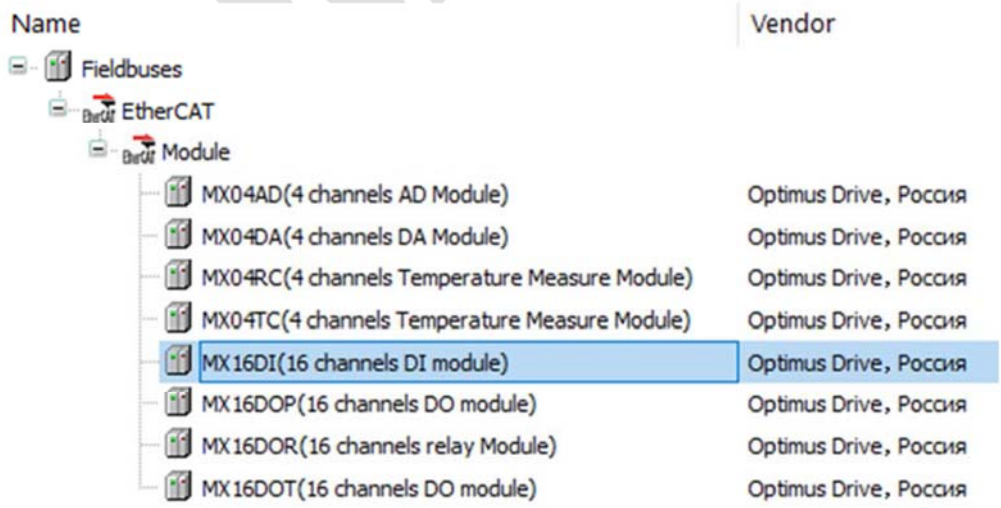
В открывшемся окне выберите **Fieldbuses - EtherCAT Slave – Optimus Drive – Terminal Coupler - R2EC**:



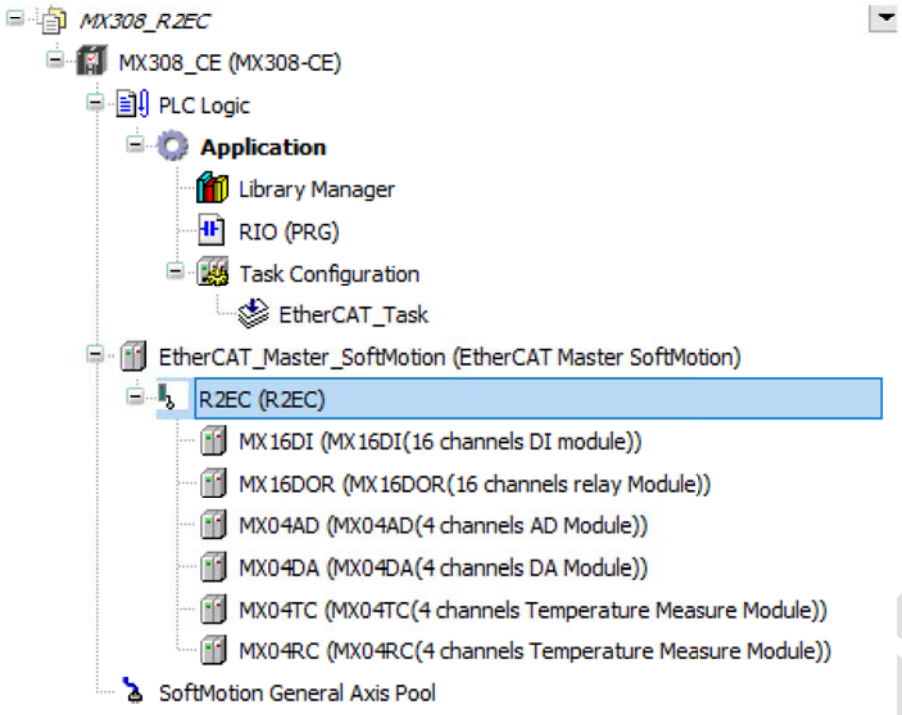
В древе проекта появится пункт:



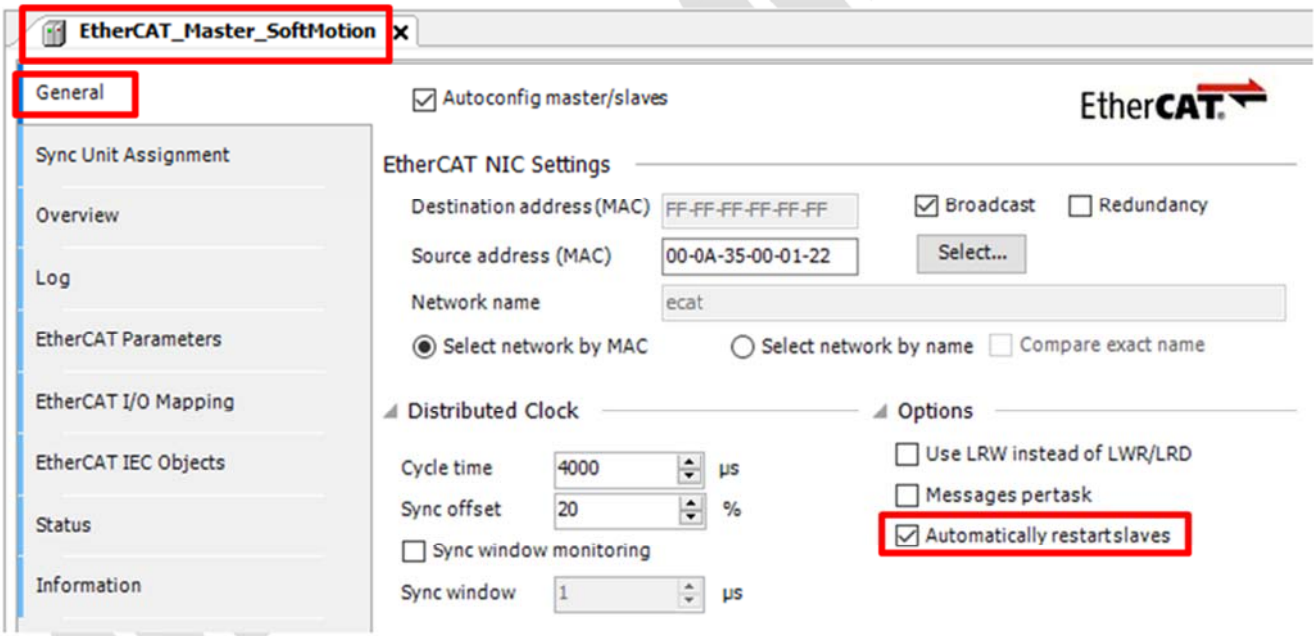
На этом этапе мы добавили станцию удалённого ввода-вывода. Далее не неё необходимо добавить нужные модули расширения. Для этого щёлкните правой кнопкой мышки на пункте **R2EC** и в отрывшемся окне выберите нужные модули:



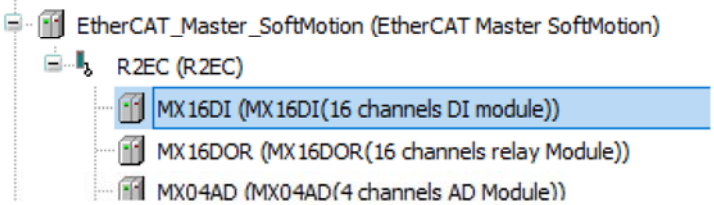
В итоге в древе проекта сформируется нужный состав оборудования, например такой:



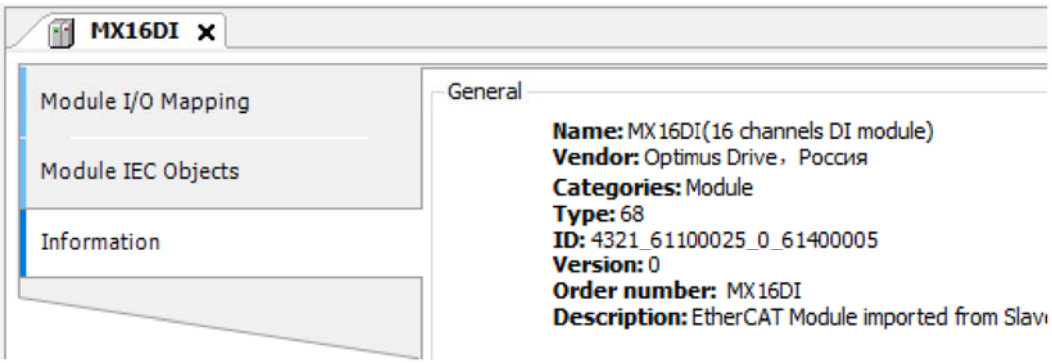
Для автоматического подхвата станции необходимо поставить флажок:



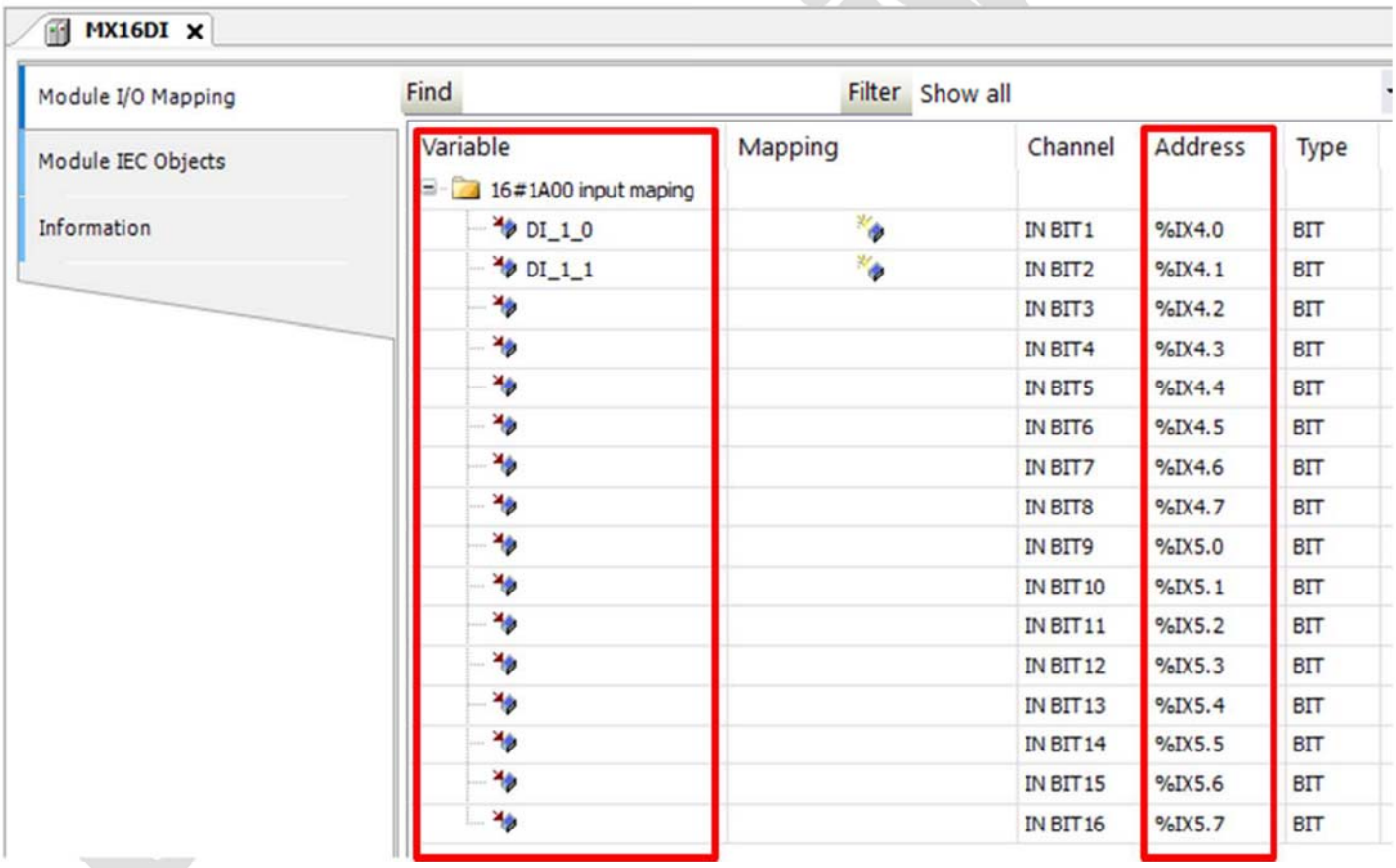
Параметры любого из модулей можно посмотреть и настроить открыв страницу данного модуля. Для этого щёлкните дважды левой кнопкой мышки на названии модуля в древе проекта, например на MX16DI:



Откроется вкладка с параметрами модуля.

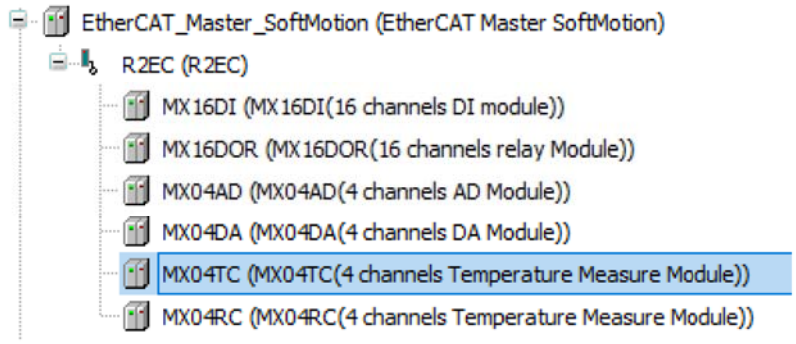


Параметры любого модуля включают вкладку **Module I/O Mapping**, в которой можно посмотреть регистры, которые выделила система для чтения/записи данных с модуля, а также привязать к ним свои переменные. В программе контроллера пользоваться предпочтительно переменными, а не физическими регистрами. Например, для модуля MX16DI данная вкладка выглядит так:

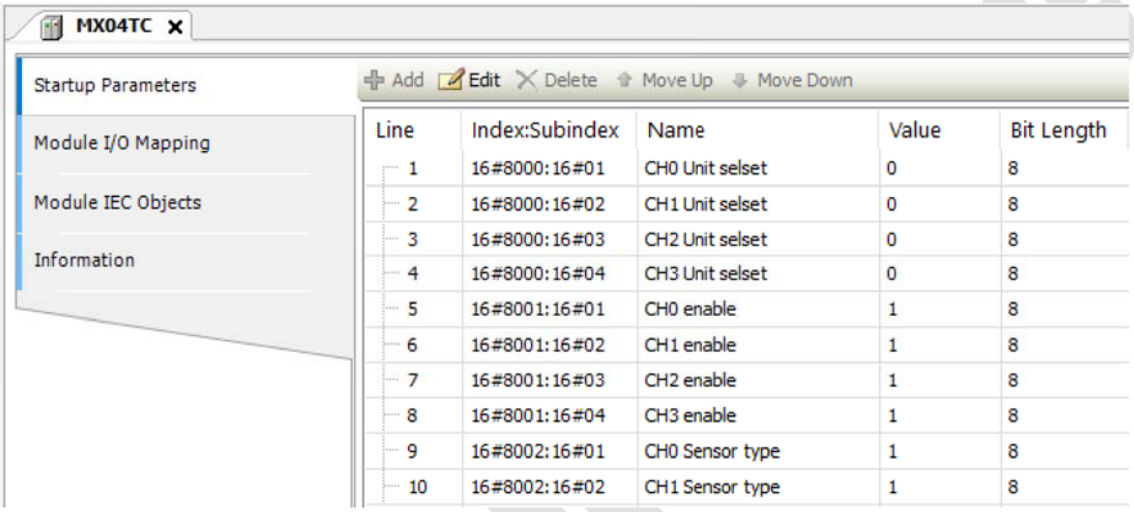


В колонке **Address** указаны физические регистры, которые система привязала к входам, а в колонке **Variable** можно дать свои названия переменным, через которые можно будет обращаться к входам в программе.

В аналоговых и температурных модулях появляется ещё вкладка **Startup Parameters** с настройками рабочих режимов. Например, откройте двойным щелчком левой кнопки мышки термопарный модуль MX04TC:



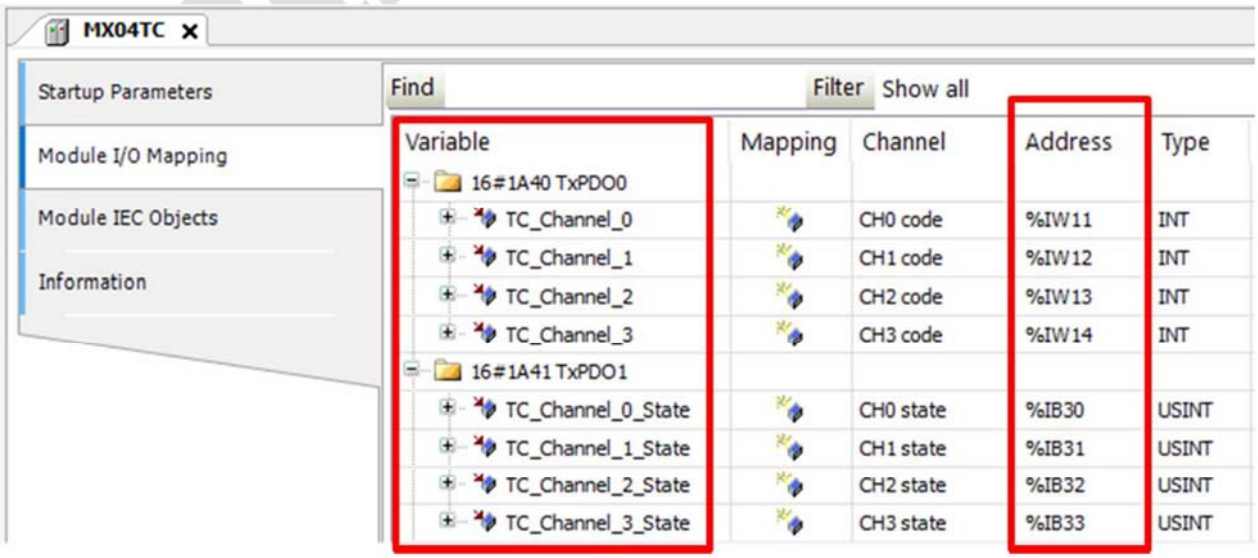
Откроется вкладка с настройками модуля:



Line	Index:Subindex	Name	Value	Bit Length
1	16#8000:16#01	CH0 Unit selset	0	8
2	16#8000:16#02	CH1 Unit selset	0	8
3	16#8000:16#03	CH2 Unit selset	0	8
4	16#8000:16#04	CH3 Unit selset	0	8
5	16#8001:16#01	CH0 enable	1	8
6	16#8001:16#02	CH1 enable	1	8
7	16#8001:16#03	CH2 enable	1	8
8	16#8001:16#04	CH3 enable	1	8
9	16#8002:16#01	CH0 Sensor type	1	8
10	16#8002:16#02	CH1 Sensor type	1	8

В пункте **Startup Parameters** можно задать рабочий режим каждого измерительного канала такие как тип датчика, единицы измерения и т.п. Параметры приведены в настоящем Руководстве в Главе с описанием температурных модулей.

В пункте **Module I/O Mapping** можно посмотреть регистры, которые выделила система для чтения/записи данных с модуля, а также привязать к ним свои переменные:

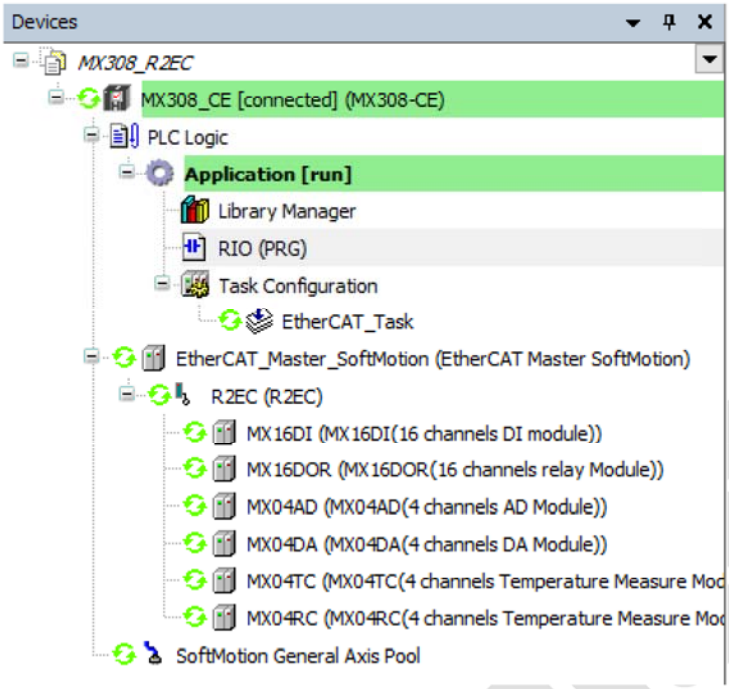


Variable	Mapping	Channel	Address	Type
16#1A40 TxPDO0				
TC_Channel_0		CH0 code	%IW11	INT
TC_Channel_1		CH1 code	%IW12	INT
TC_Channel_2		CH2 code	%IW13	INT
TC_Channel_3		CH3 code	%IW14	INT
16#1A41 TxPDO1				
TC_Channel_0_State		CH0 state	%IB30	USINT
TC_Channel_1_State		CH1 state	%IB31	USINT
TC_Channel_2_State		CH2 state	%IB32	USINT
TC_Channel_3_State		CH3 state	%IB33	USINT

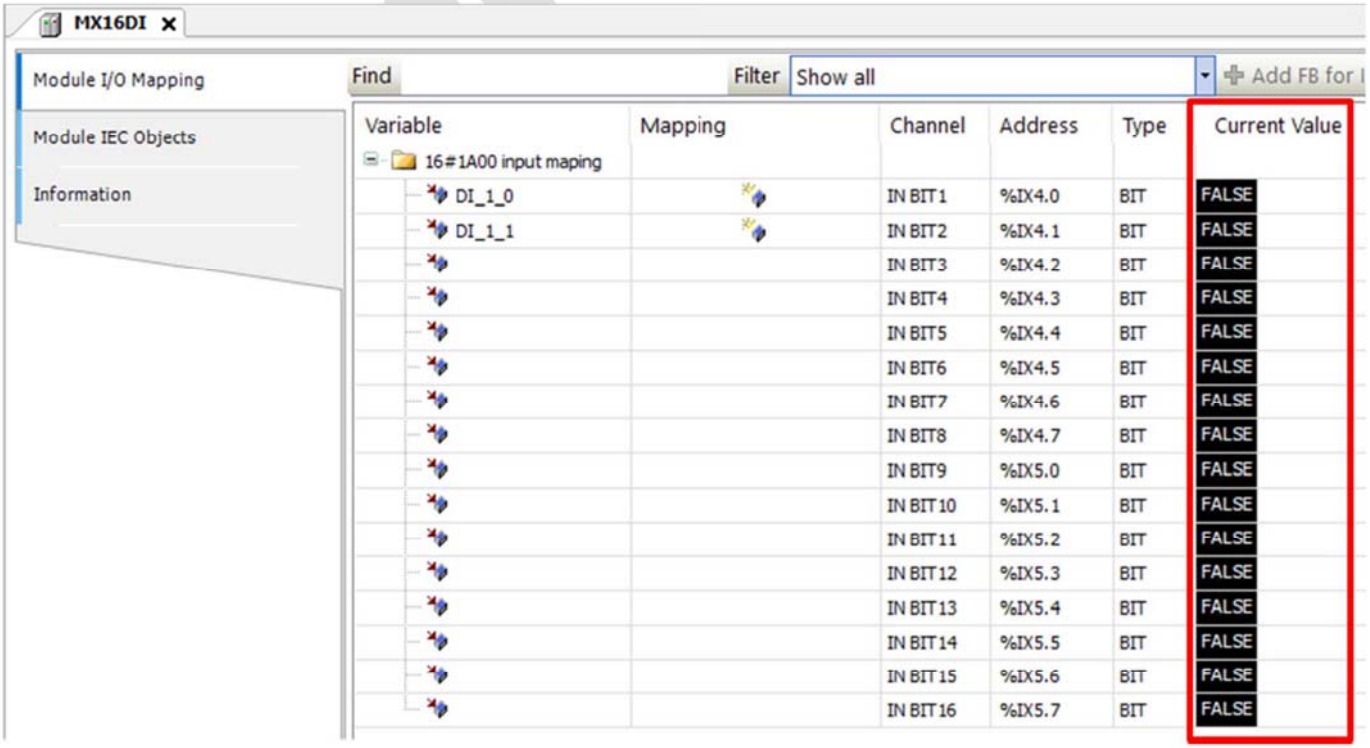
Аналогичным образом строится работы со всеми модулями.

Далее, установите модули на станцию строго в той последовательности как они сконфигурированы в проекте. Соедините ЦПУ и станцию EtherCAT кабелем. Подайте на ЦПУ, станцию и аналоговые/температурные модули питание 24 VDC.
Установите соединение с ЦПУ и загрузите проект с нужной конфигурацией станции. Войдите в онлайн режим с контроллером (см. настоящее Руководство).

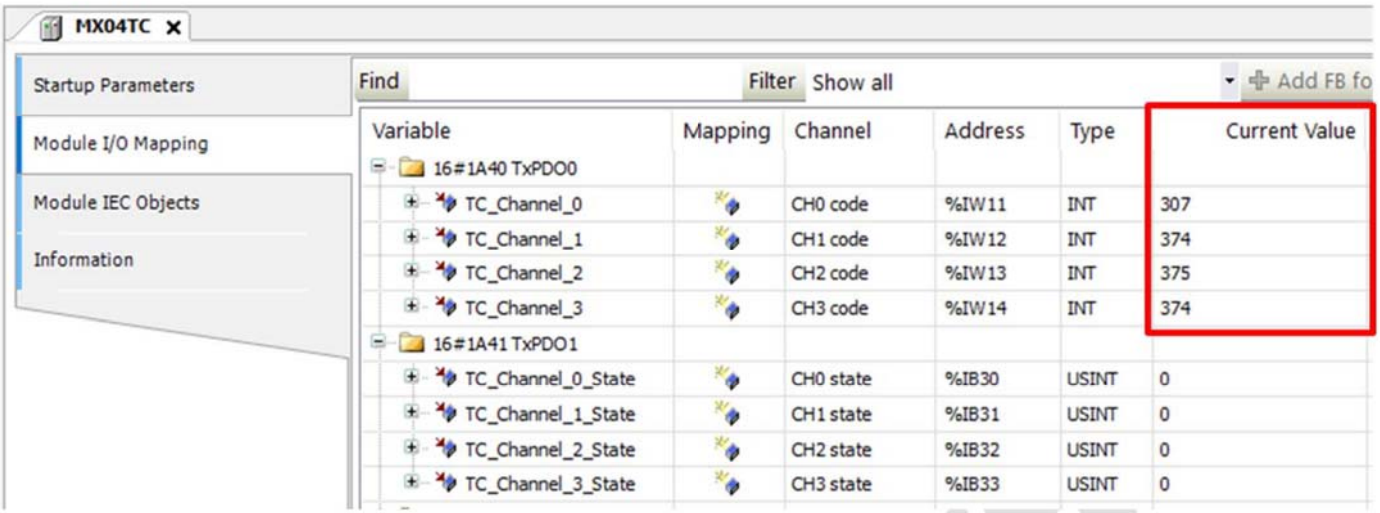
Если все настройки были сделаны правильно, то все устройства должны быть подсвечены зелёным цветом:



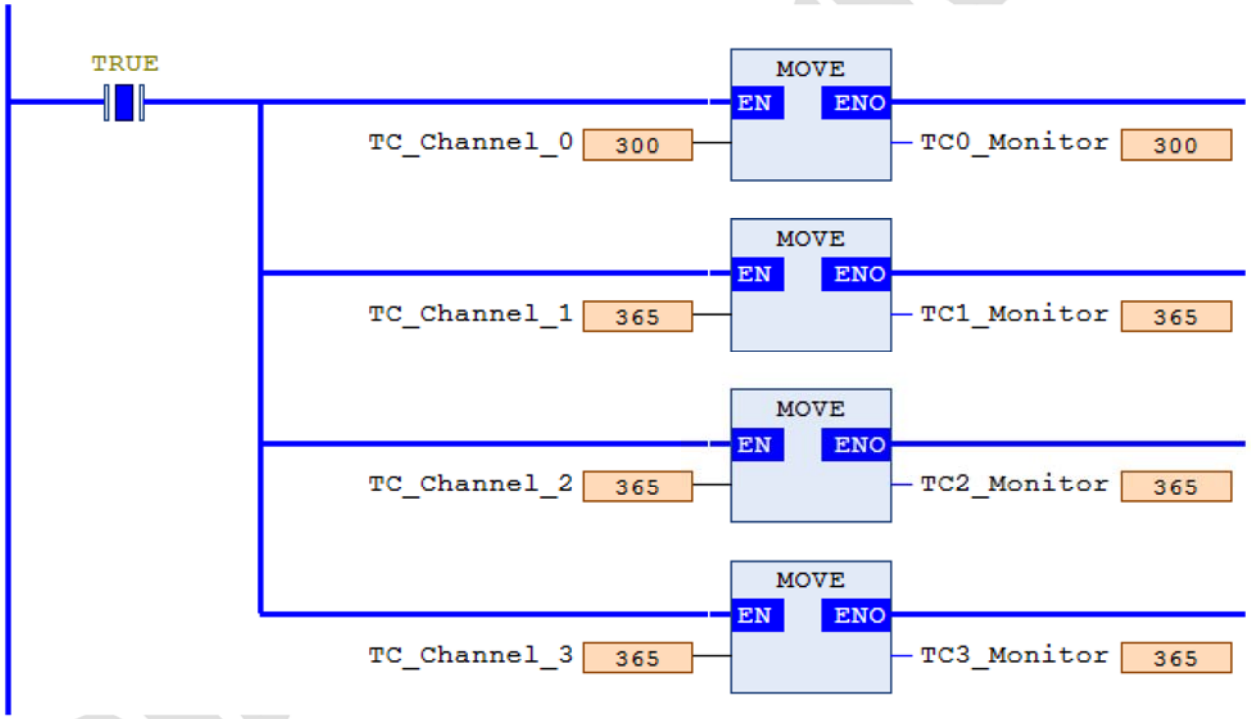
В онлайн режиме будет доступна колонка с текущим состоянием каналов ввода-вывода, например MX16DI:



Текущие значение на измерительных каналах модуля термопар MX04TC:



В программе контроллера данные можно использовать через созданные переменные:

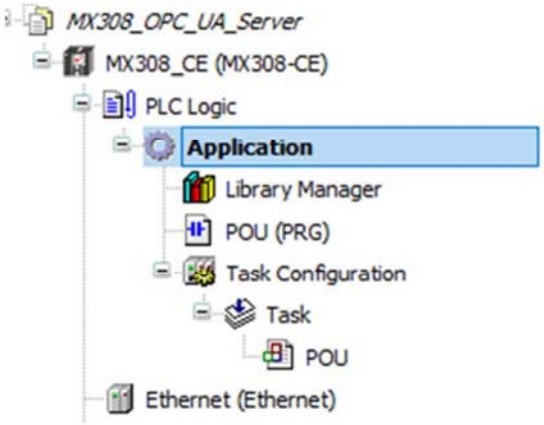


Связь по протоколу OPC UA в режиме сервера

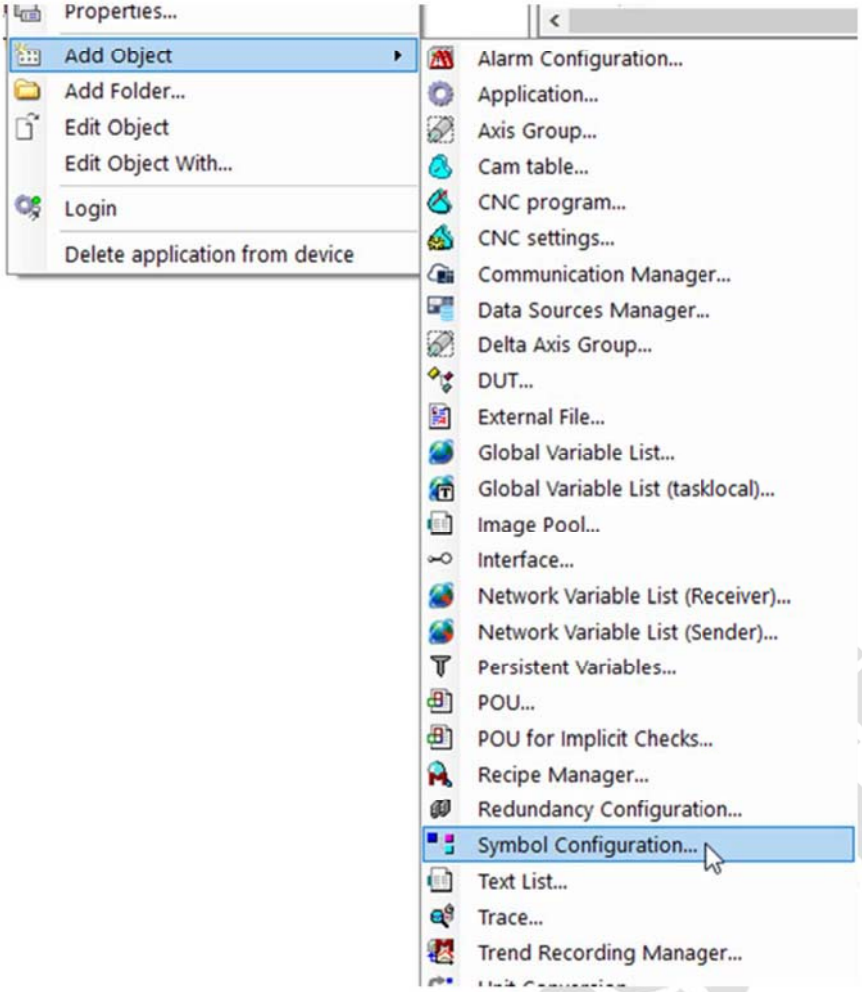
Протокол **OPC UA** (Unified Architecture) – это современный широко распространённый стандарт, описывающий передачу данных в промышленных сетях, анонсированный в 2008 году организацией OPC Foundation и закреплённый в стандарте IEC62541. Является кросс-платформенным протоколом клиент-серверной архитектуры. Контроллеры серии MX300 поддерживают **OPC UA Server** (Ведомый, отвечает на запросы Мастера).

Для начала работы создайте проект, добавьте устройство (контроллер MX308) и адаптер Ethernet, создайте POU, добавьте несколько переменных, установите связь с контроллером и загрузите проект (см. соответствующие разделы настоящего Руководства).

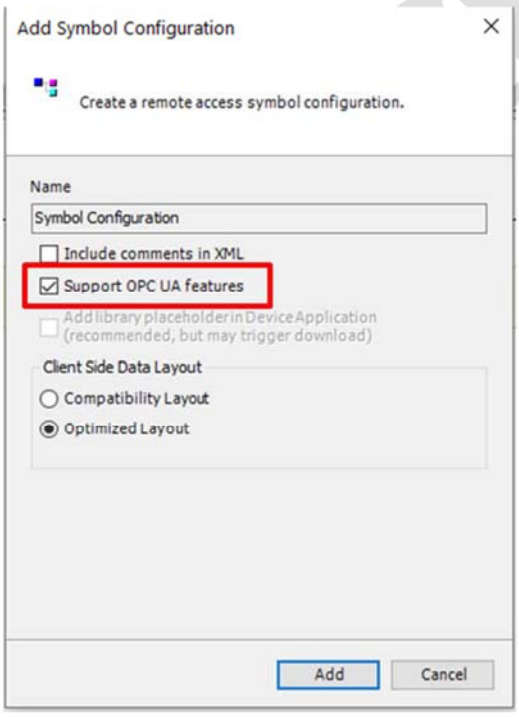
В древе проекта выберите пункт **Application** и нажмите правую кнопку мышки:



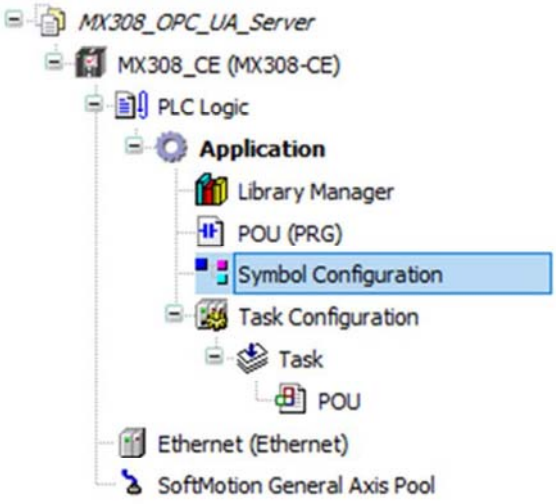
В появившемся меню выберите пункт **Add Object – Symbol Configuration**:



В открывшемся окне поставьте флажок **Support OPC UA features** и нажмите кнопку **Add**:



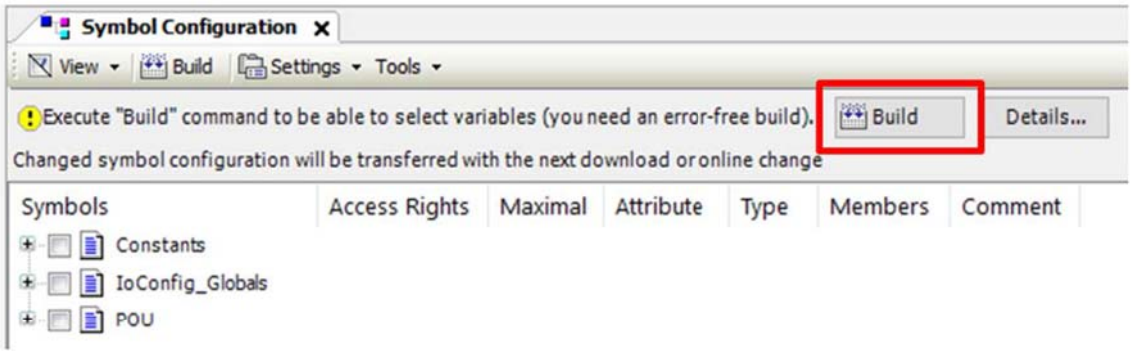
В древе проекта появится пункт **Symbol Configuration**:



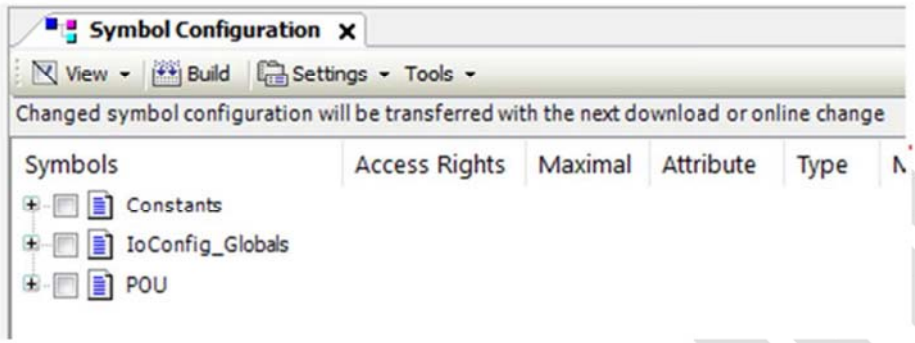
Для наглядности в нашем проекте созданы переменные различных типов данных:

Scope	Name	Address	Data type
VAR	**bool_coil**		BOOL
VAR	**bool_contact**		BOOL
VAR	**byte_IN**		BYTE
VAR	**byte_OUT**		BYTE
VAR	**dint_IN**		dint
VAR	**dint_OUT**		dint
VAR	**lint_IN**		LINT
VAR	**lint_OUT**		LINT
VAR	**lreal_IN**		LREAL
VAR	**lreal_OUT**		LREAL
VAR	**real_IN**		REAL
VAR	**real_OUT**		REAL
VAR	**string1**		string(20)
VAR	**string2**		string(20)
VAR	**word_IN**		word
VAR	**word_OUT**		word
VAR	**wstring1**		wstring(20)
VAR	**wstring2**		wstring(20)

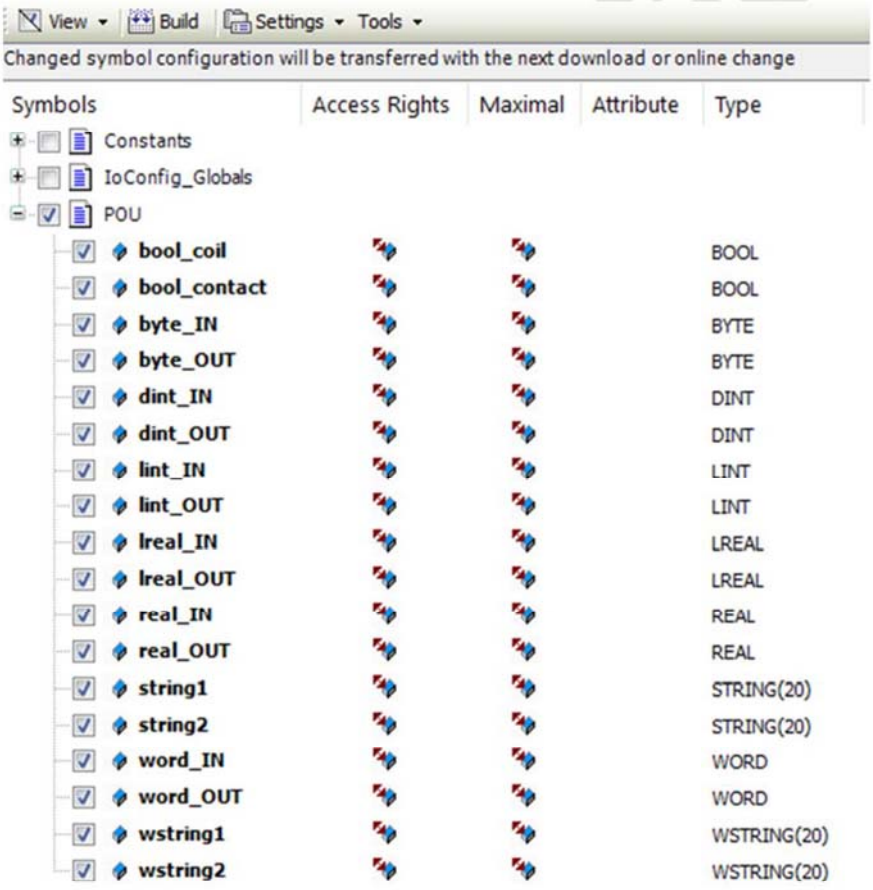
Для того, чтобы сделать данные переменные доступными по протоколу OPC UA Server необходимо двойным щелчком левой кнопки мышки открыть пункт **Symbol Configuration**. В открывшейся вкладке необходимо выполнить компиляцию проекта путём нажатия кнопки **Build**. Проект соответственно должен быть готов к компиляции. Без неё переменные не станут доступны для чтения-записи.



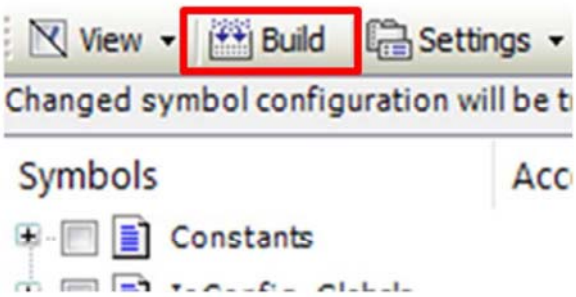
После удачной компиляции вкладка примет такой вид:



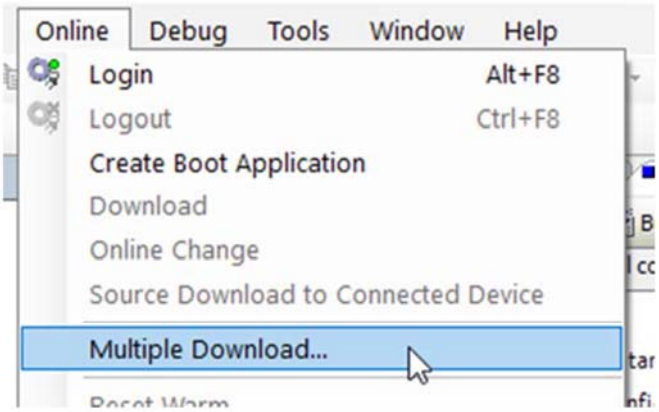
После компиляции необходимо развернут список и отметить нужные переменные, в нашем примере все. Тип доступа **(Access Rights) Read-Write**.



После этого необходимо снова выполнить компиляцию проекта:

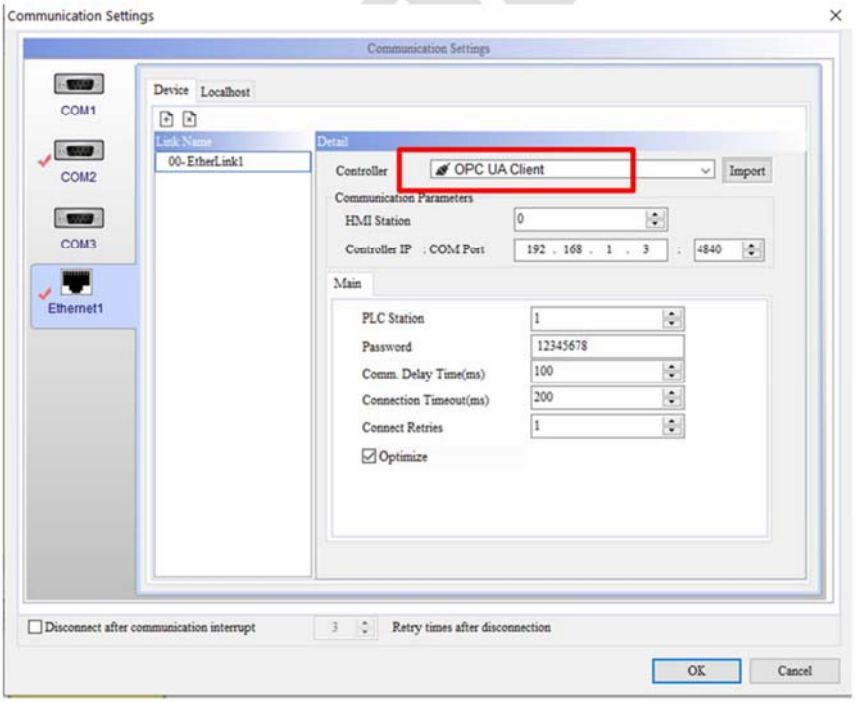


Загрузите проект в контроллер:

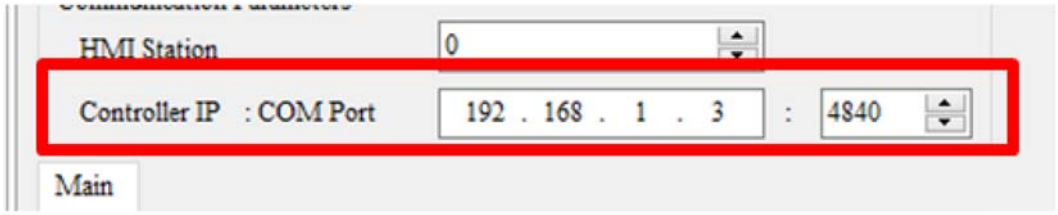


После загрузки проекта в контроллер, отмеченные переменные станут доступны для чтения-записи посредством протокола OPC UA.

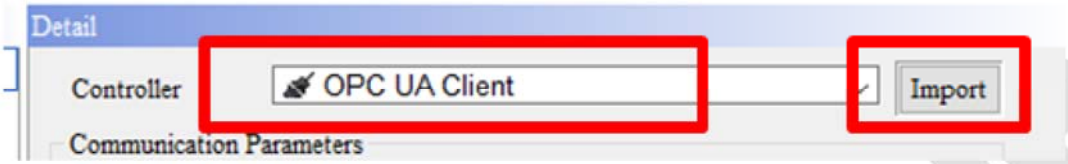
Для примера в качестве OPC UA клиента (Мастера) используем панель оператора Delta Electronics DOP-107WV. В среде программирования **DIAScreen** создайте проект для данной панели и в настройках порта Ethernet выберите драйвер **OPC UA Client**:



В качестве сервера укажите IP адрес контроллера MX308:

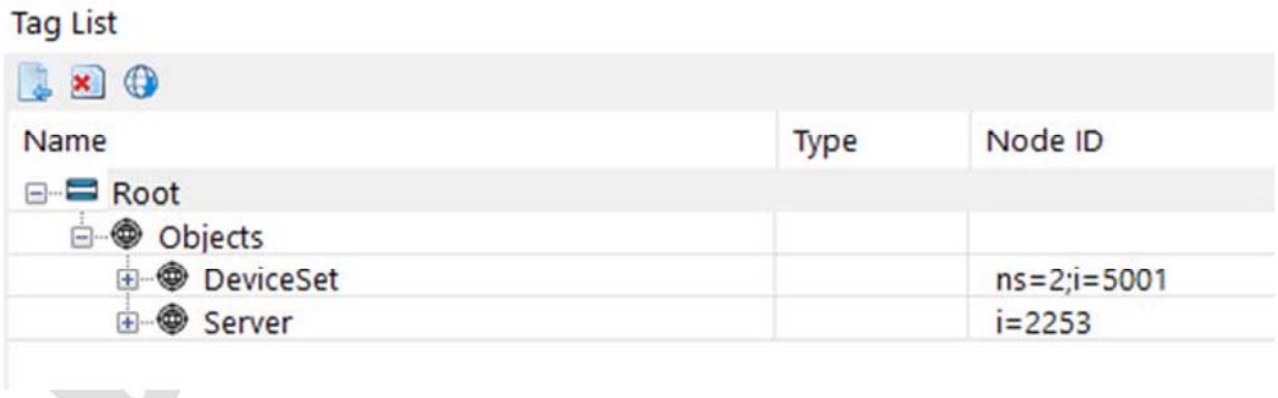


Далее нажмите кнопку **Import** справа от имени драйвера:

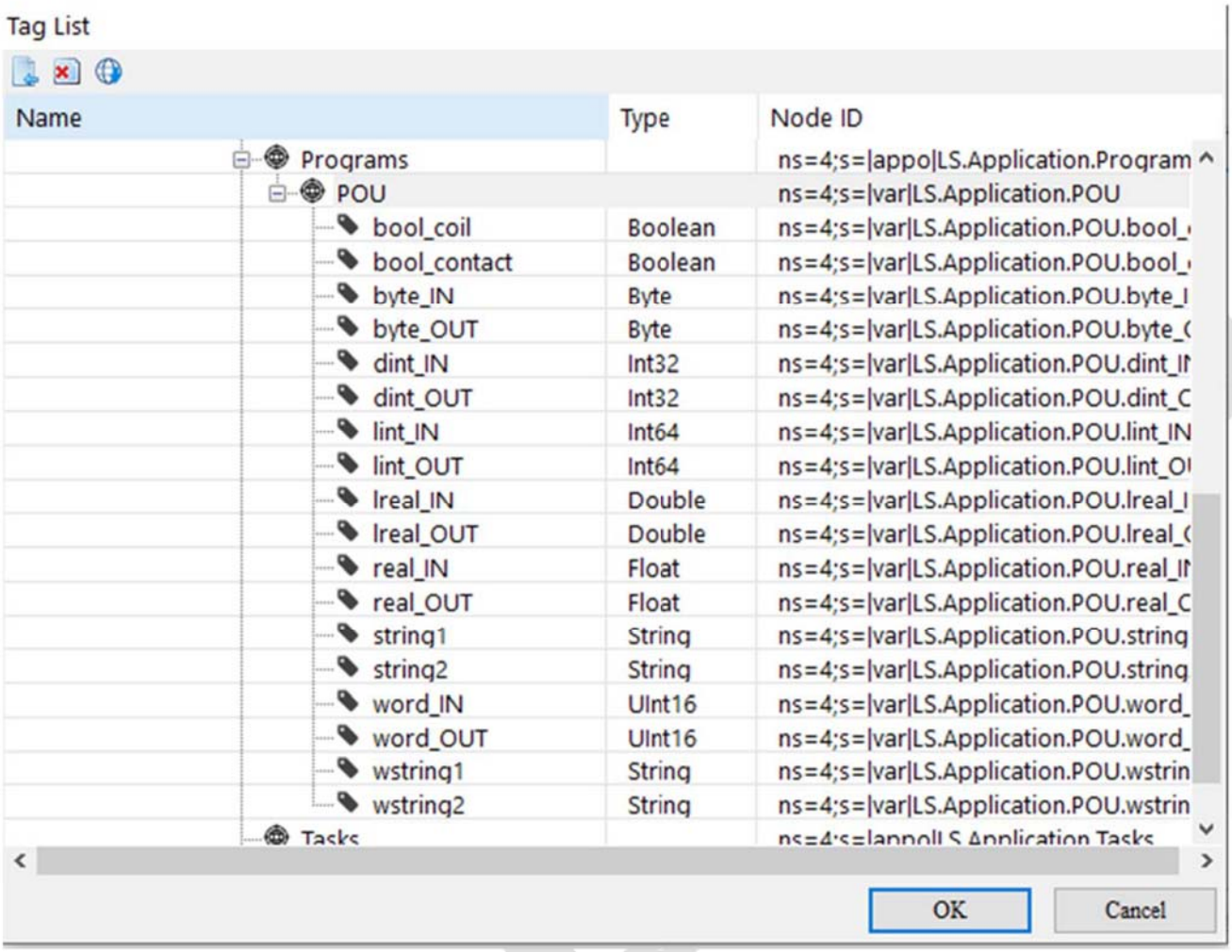


В открывшемся окне нажмите поиск сервера:

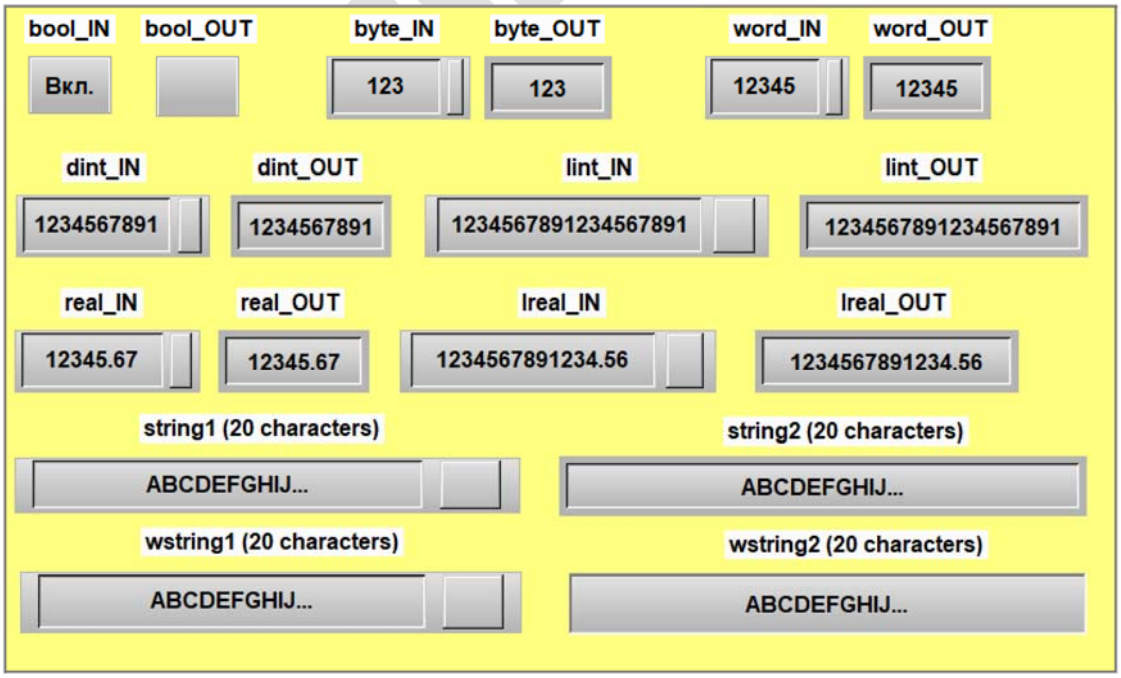
При наличии нормального соединения ПК – контроллер, среда разработки DIAScreen найдёт в сети контроллер и выведет древо сервера (контроллер MX308):



Разверните полностью пункт **Server** до появления списка тегов:



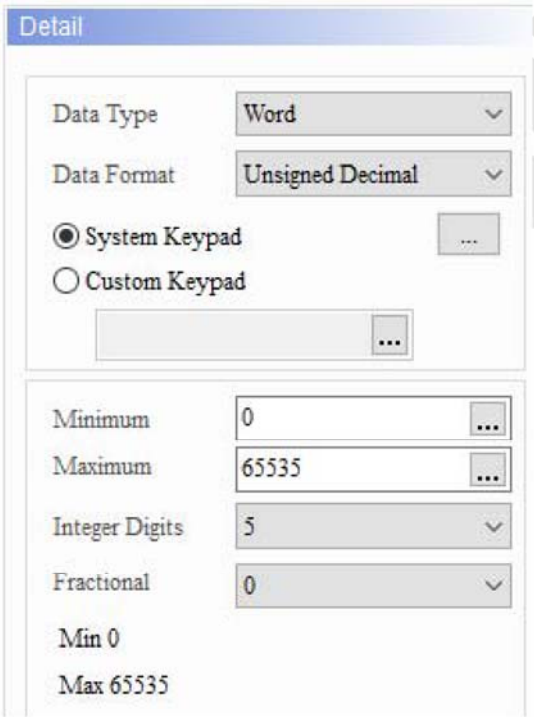
Для демонстрации передачи данных достаточно создать несколько экранных объектов в соответствии с типами данных.



Для тегов типа BOOL используются объекты типа Button Maintained и Multistate Indicator.
Числовые типы данных отображаются и вводятся через объекты типа Numeric Input/Display, которые настраиваются на соответствующий тип данных.
Например, для типа WORD:

HardwareRevision	String	ns=4;s=|vprop|LS.Applic
Programs		ns=4;s=|appo|LS.Applic
POU		ns=4;s=|var|LS.Applicati
byte_IN	Byte	ns=4;s=|var|LS.Applicati
byte_OUT	Byte	ns=4;s=|var|LS.Applicati
dint_IN	Int32	ns=4;s=|var|LS.Applicati
dint_OUT	Int32	ns=4;s=|var|LS.Applicati
lint_IN	Int64	ns=4;s=|var|LS.Applicati
lint_OUT	Int64	ns=4;s=|var|LS.Applicati
lreal_IN	Double	ns=4;s=|var|LS.Applicati
lreal_OUT	Double	ns=4;s=|var|LS.Applicati
real_IN	Float	ns=4;s=|var|LS.Applicati
real_OUT	Float	ns=4;s=|var|LS.Applicati
string1	String	ns=4;s=|var|LS.Applicati
string2	String	ns=4;s=|var|LS.Applicati
word_IN	UInt16	ns=4;s=|var|LS.Applicati
word_OUT	UInt16	ns=4;s=|var|LS.Applicati
wstring1	String	ns=4;s=|var|LS.Applicati
wstring2	String	ns=4;s=|var|LS.Applicati

необходимо сделать следующие настройки:



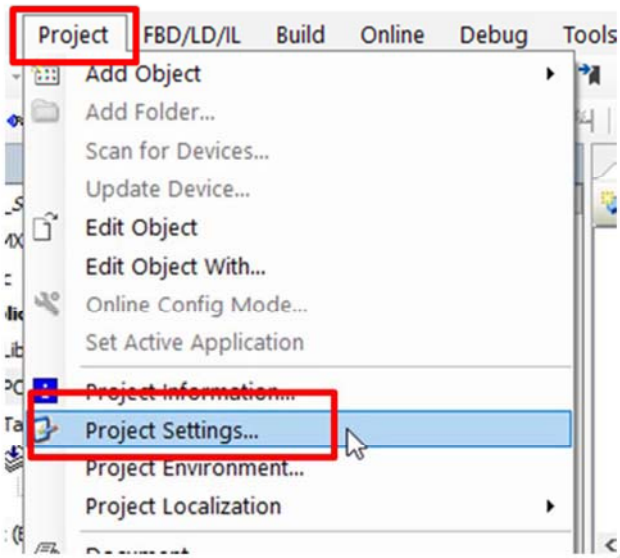
Для ввода и отображения символов типа ASCII (1 байт) используются объекты Character Input/Display. В контроллере данный тип определяется как string(**). В скобках указывается количество символов (байтов).

В настройках объектов панели количество символов указывается аналогичное (байты).

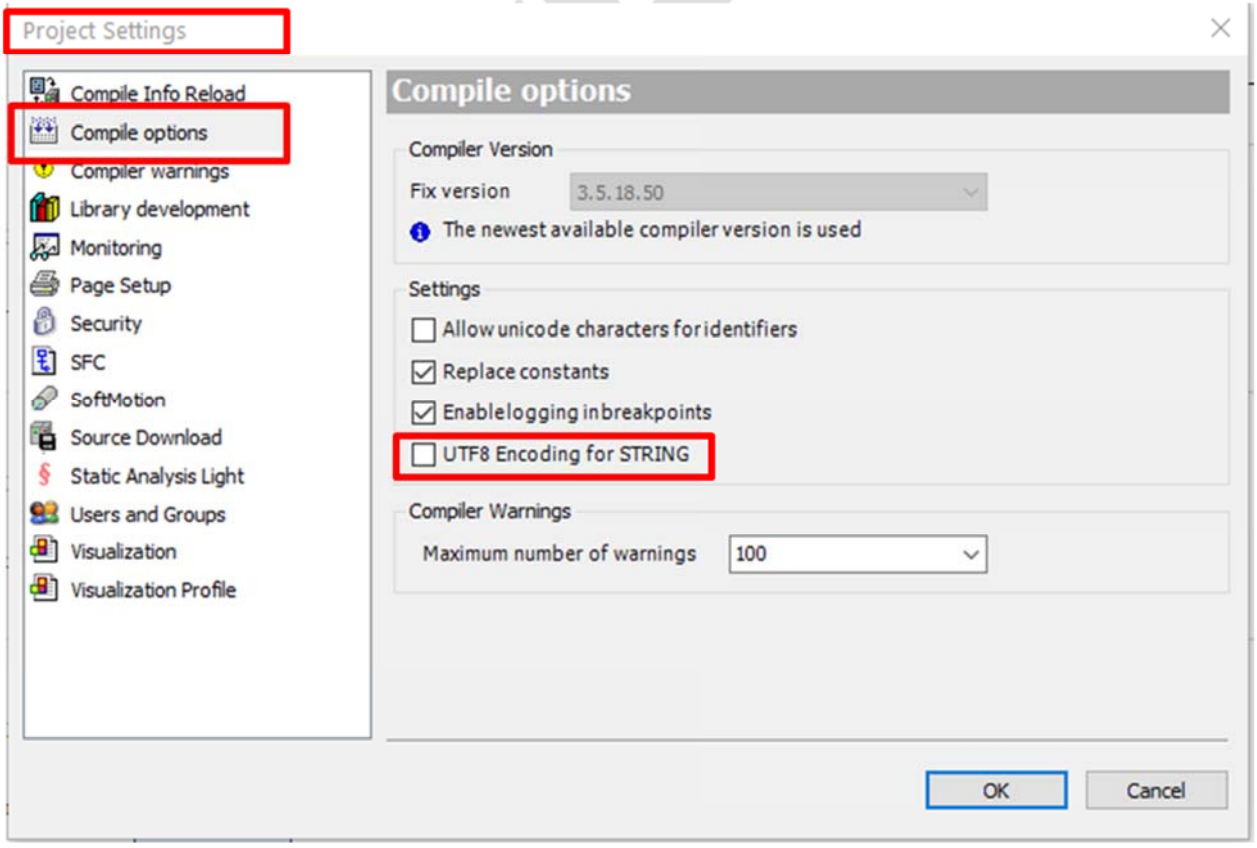
Для ввода и отображения символов типа Unicode (2 байта) используется объект Multilanguage Input. В контроллере данный тип определяется как wstring(**). В скобках указывается количество символов. В типе данных wstring используется кодировка типа UCS2, в которой каждый символ занимает 2 байта.

Для однозначности трактования типов string и wstring в настройках среды программирования контроллера необходимо убедиться в наличии следующих настроек:

Project – Project Settings:

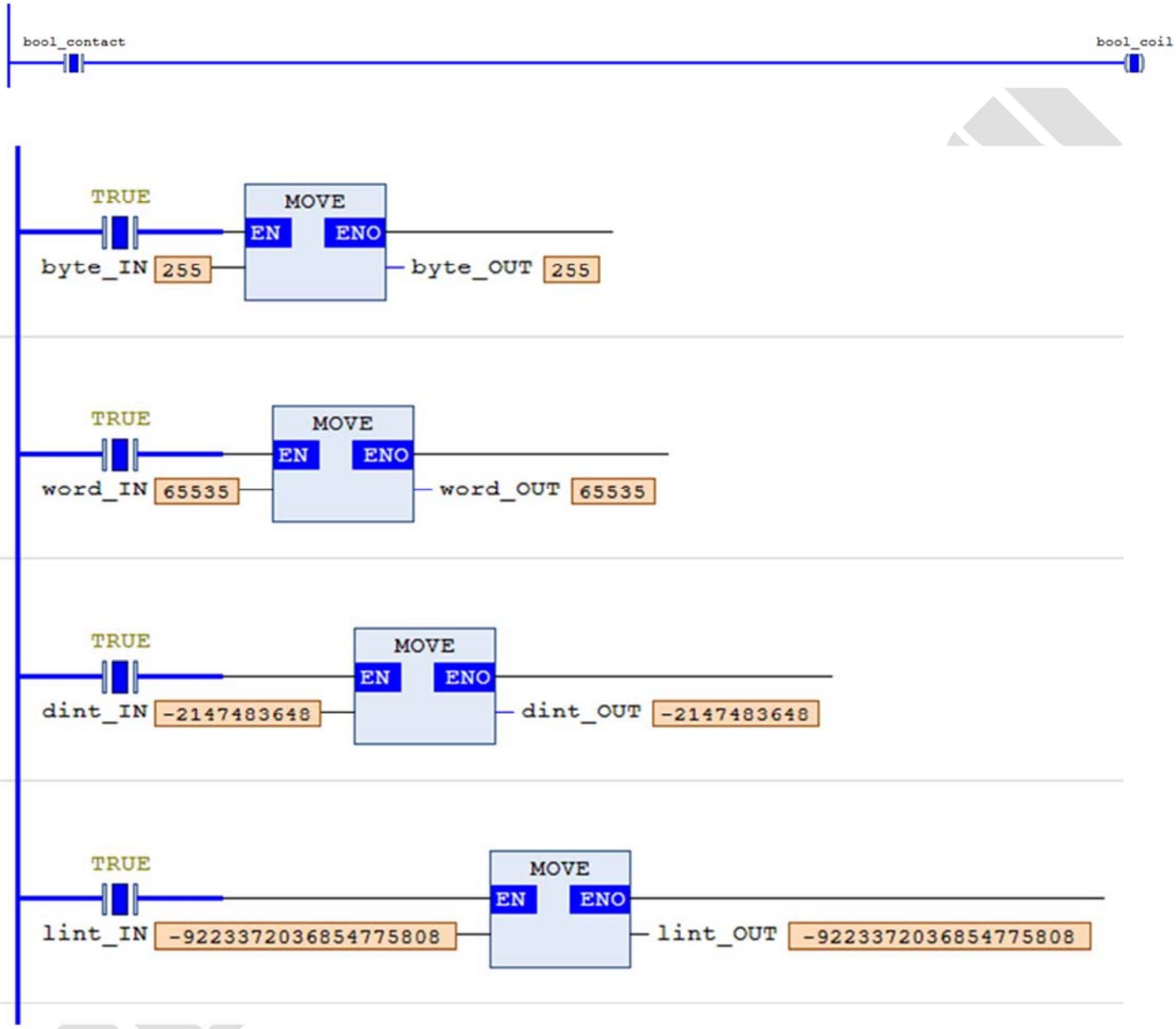


Флажок **UTF8 Encoding for String** должен быть снят!



Для наглядности в проекте контроллера можно поставить созданные теги в тело программы или вывести в таблицу мониторинга Watch Table.

Загрузите проекты в контроллер и в панель. Введите значения в переменные. Они будут отображаться одинаково и в панели и в контроллере. В нашем примере будет выглядеть так:



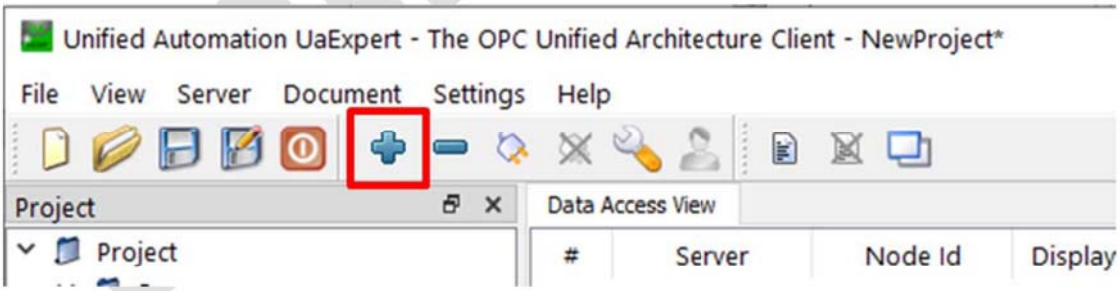
Watch 1

Expression	Application	Type	Value
POU.wstring2	MX308_CE.Application	WSTRING(20)	"АБВГуке123@#$%&<=про"
POU.string2	MX308_CE.Application	STRING(20)	'ABCDdsw!@#$%^&*()123'
POU.real_OUT	MX308_CE.Application	REAL	-99999.99
POU.lreal_OUT	MX308_CE.Application	LREAL	-9999999999999.99

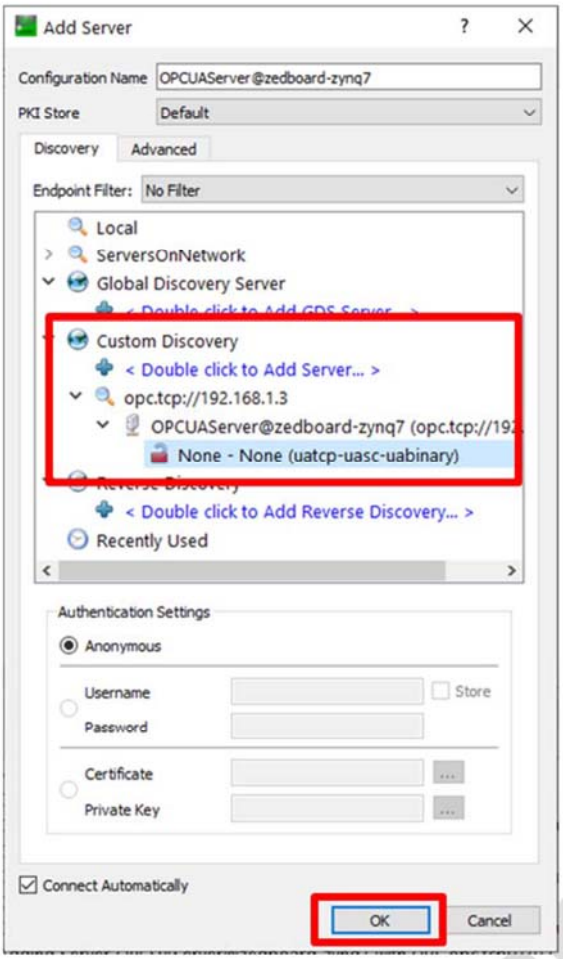
Для демонстрации сделано так, что через переменные типа ***_IN данные вводятся, а через переменные типа ***_OUT эти же данные выводятся.

Также, посмотреть работу OPC UA Server в контроллере можно через стандартную утилиту **UaExpert**. Для этого необходимо иметь связь ПК – контроллер, запустить утилиту, добавить сервер, установить связь с ним, выбрать переменные, с которыми необходимо работать.

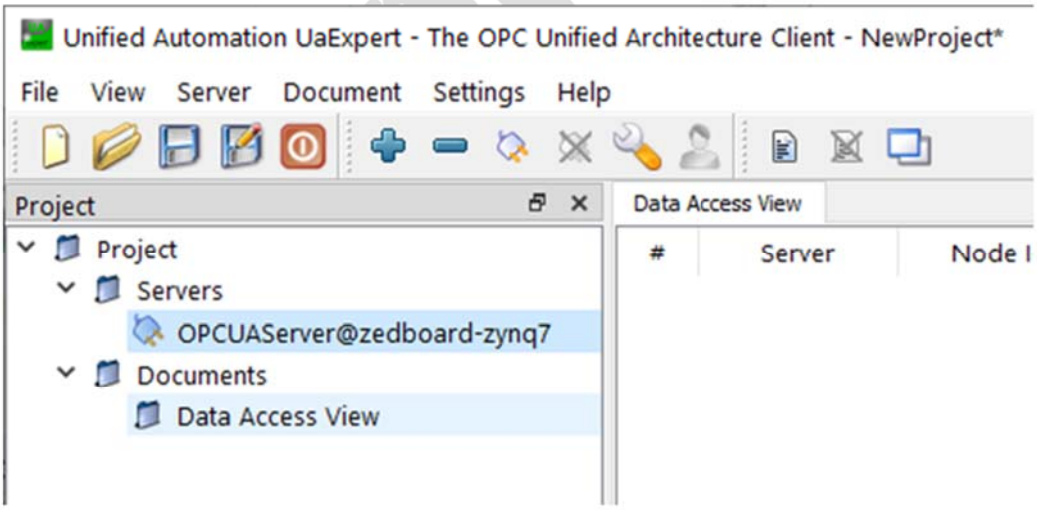
Добавление сервера:



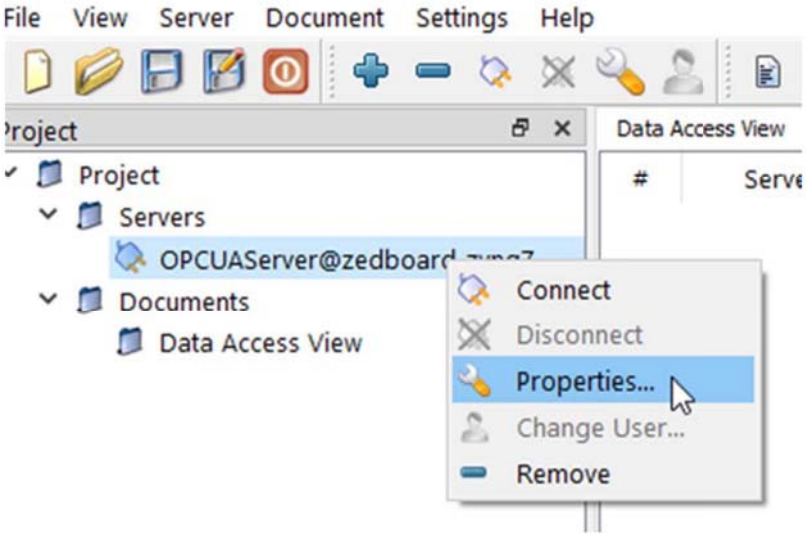
В открывшемся окне выберите сервер (задать IP адрес) и нажать **OK**:



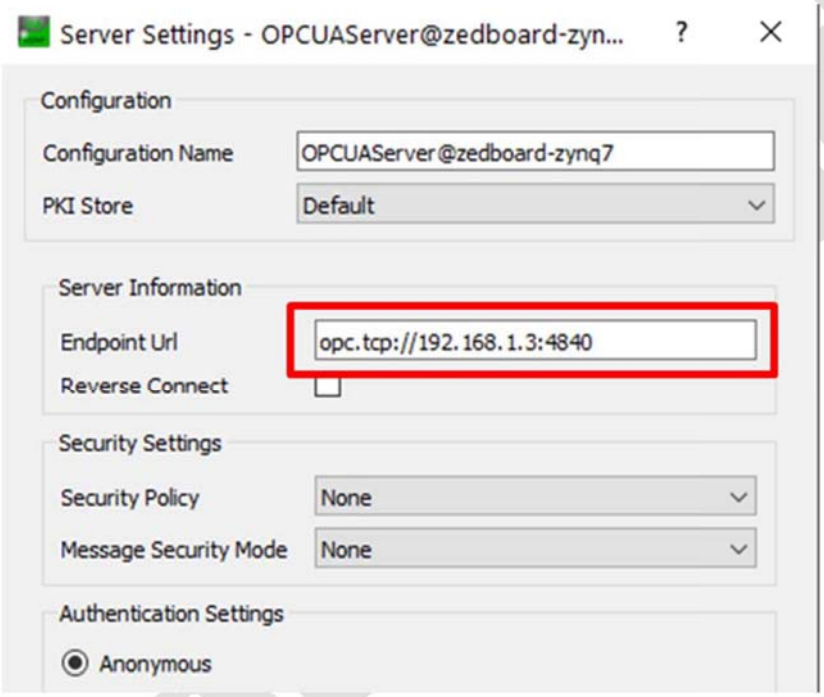
В древе проекта появится сервер:



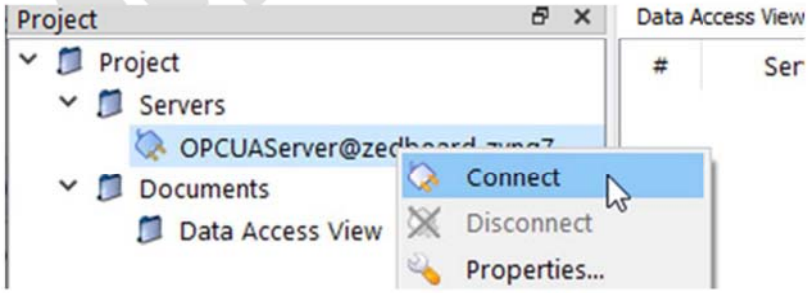
Щёлкните правой кнопкой мышки на названии сервера и выберите пункт **Properties**:



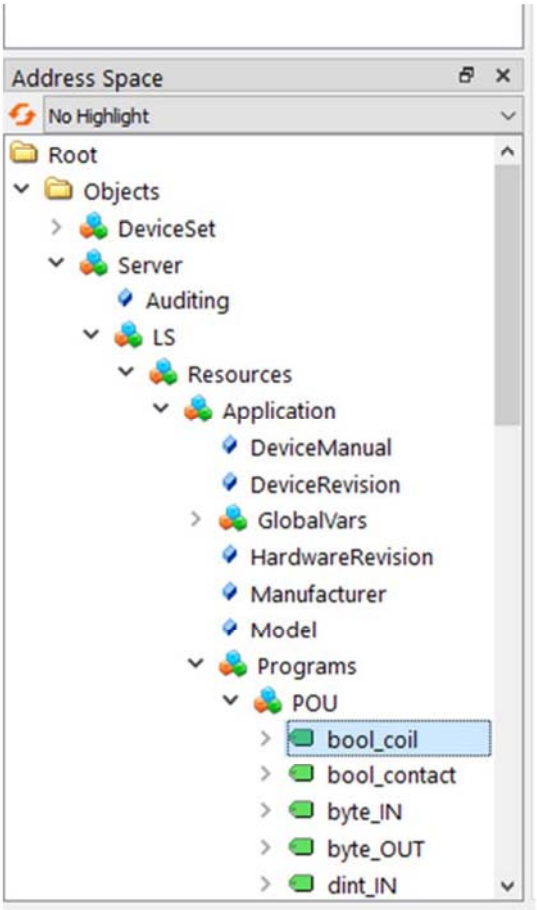
Введите IP адрес сервера (контроллера) и нажмите **ОК**:



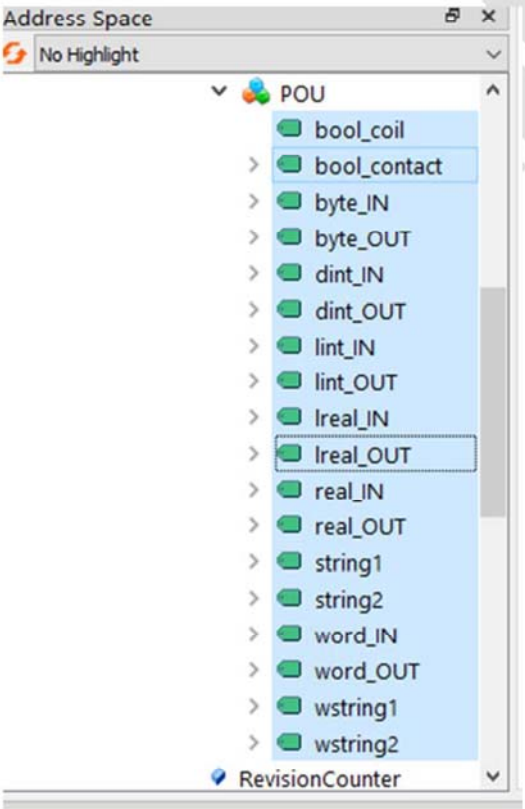
Ещё раз нажмите правой кнопкой на имени сервера и нажмите пункт **Connect**:



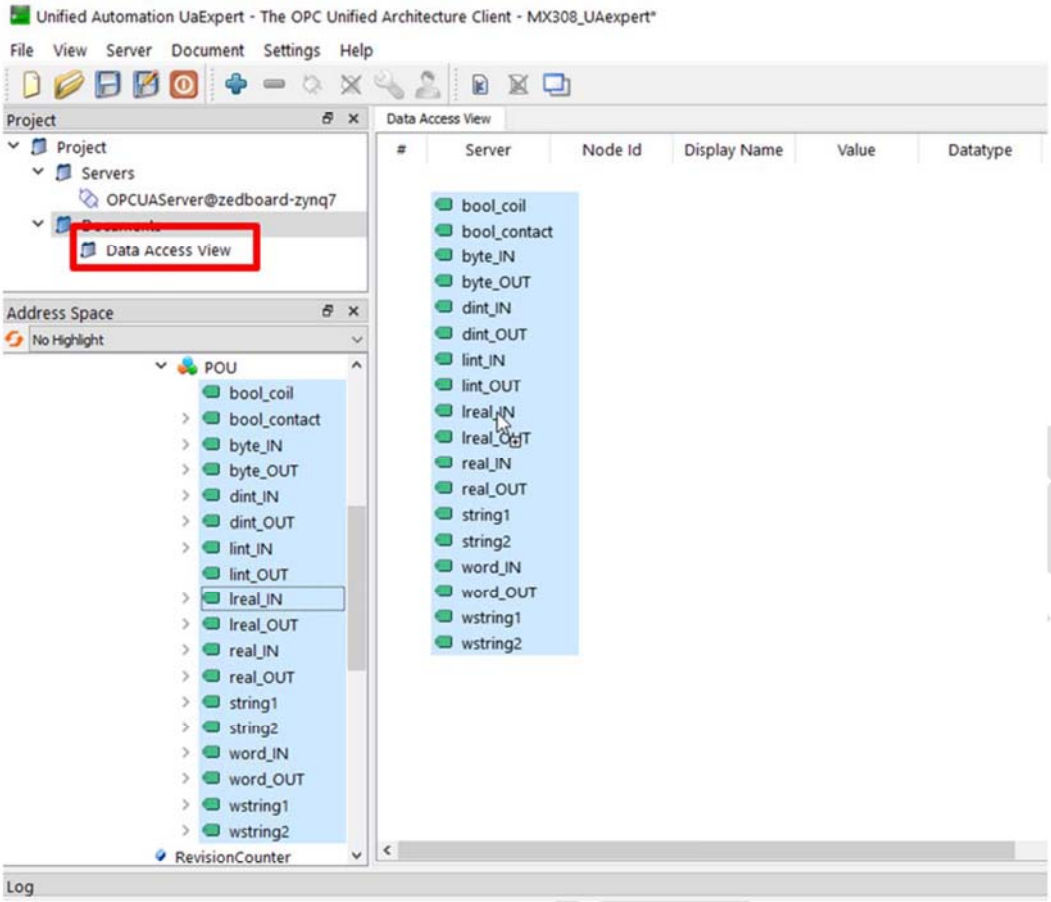
Откроется древо сервера, развернув которое Вы увидите теги:



Выделите нужные теги:



Далее откройте вкладку **Data Access View** и мышкой по технологии DRAG&DROP переместите теги в область мониторинга:



Далее появятся данные, которые в колонке Value можно менять (если переменные были помечены как Read-Write при создании).

Data Access View

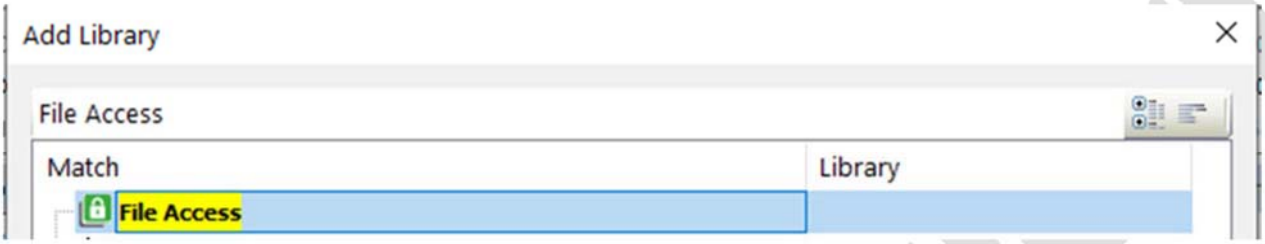
#	Display Name	Value	Datatype	ource Timestam	erver Timestam	Statusc
1	bool_coil	true	Boolean	18:53:58.489	18:53:58.489	Good
2	bool_contact	true	Boolean	18:53:58.489	18:53:58.489	Good
3	byte_IN	255	Byte	18:53:58.489	18:53:58.489	Good
4	byte_OUT	255	Byte	18:53:58.489	18:53:58.489	Good
5	dint_IN	-2147483648	Int32	18:53:58.489	18:53:58.489	Good
6	dint_OUT	-2147483648	Int32	18:53:58.489	18:53:58.489	Good
7	lint_IN	-9223372036854775808	Int64	18:53:58.489	18:53:58.489	Good
8	lint_OUT	-9223372036854775808	Int64	18:53:58.489	18:53:58.489	Good
9	string1	ABCDdsw!@#$%^&*()123	String	18:53:58.489	18:53:58.489	Good
10	string2	ABCDdsw!@#$%^&*()123	String	18:53:58.489	18:53:58.489	Good
11	word_IN	65535	UInt16	18:53:58.489	18:53:58.489	Good
12	word_OUT	65535	UInt16	18:53:58.489	18:53:58.489	Good
13	wstring1	АБВГуке123@#$%&<=про	String	18:53:58.489	18:53:58.489	Good
14	wstring2	АБВГуке123@#$%&<=про	String	18:53:58.489	18:53:58.489	Good

Работа с SD картой

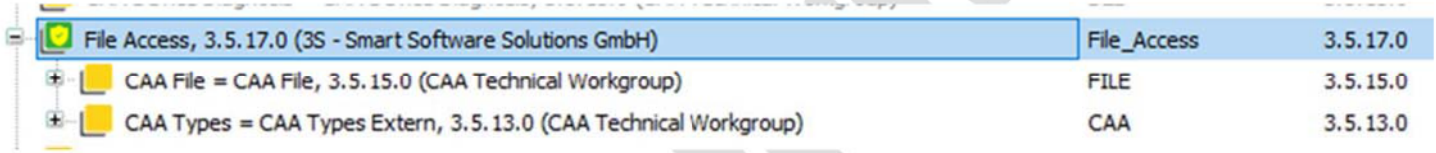
В данной Главе рассматриваются вопрос записи файла из памяти контроллера на SD карту, вставленную в слот на контроллере. Формат карты – FAT32. Объём до 32 Гб. Также, рассматривается обратная процедура копирования файла с SD карты в память контроллера. С целью демонстрации данного функционала будет показан механизм создания файла, но подробно операции с файлами в данной Главе не рассматриваются.

Для выполнения указанных процедур в проект должны быть добавлены следующие библиотеки:

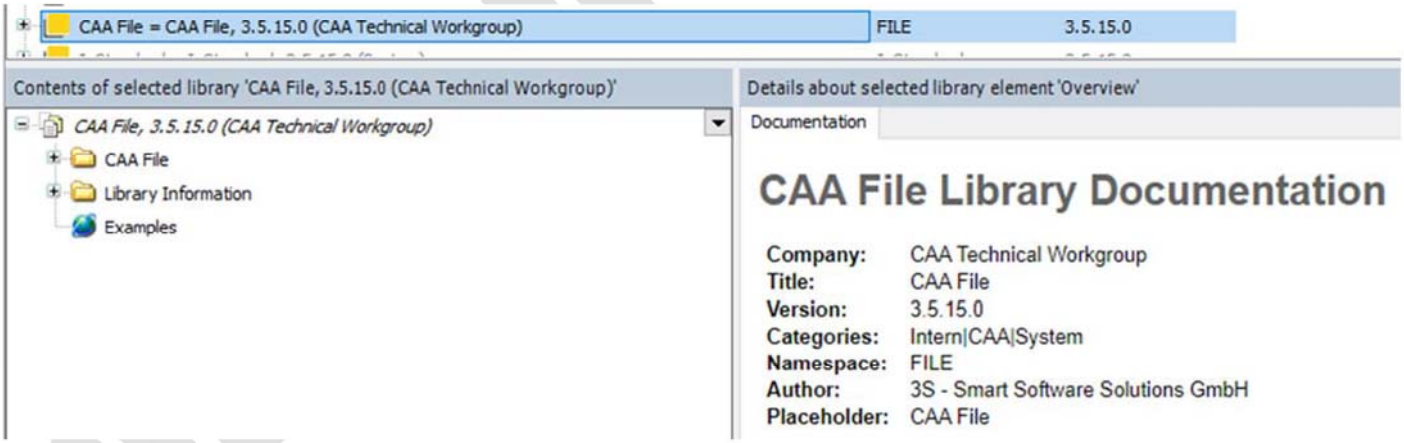
File Access



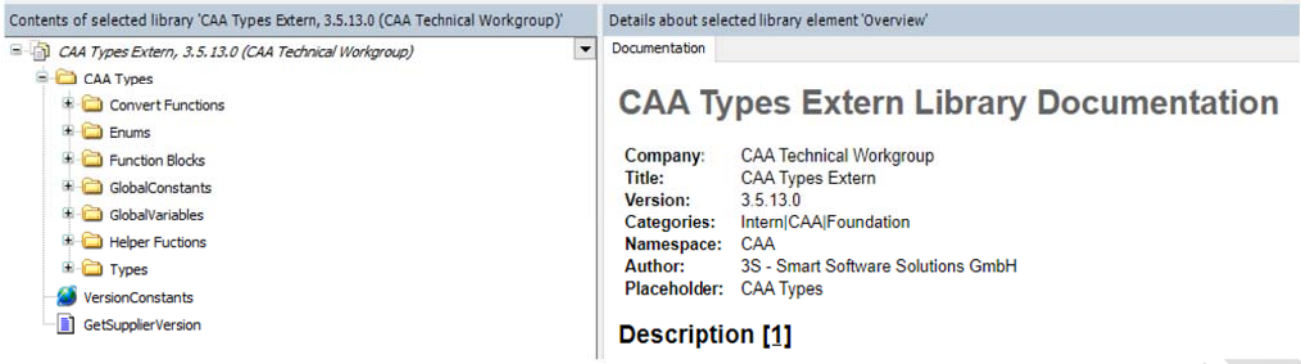
После установки в списке библиотек должны появится два пункта **CAA File** и **CAA Types**:



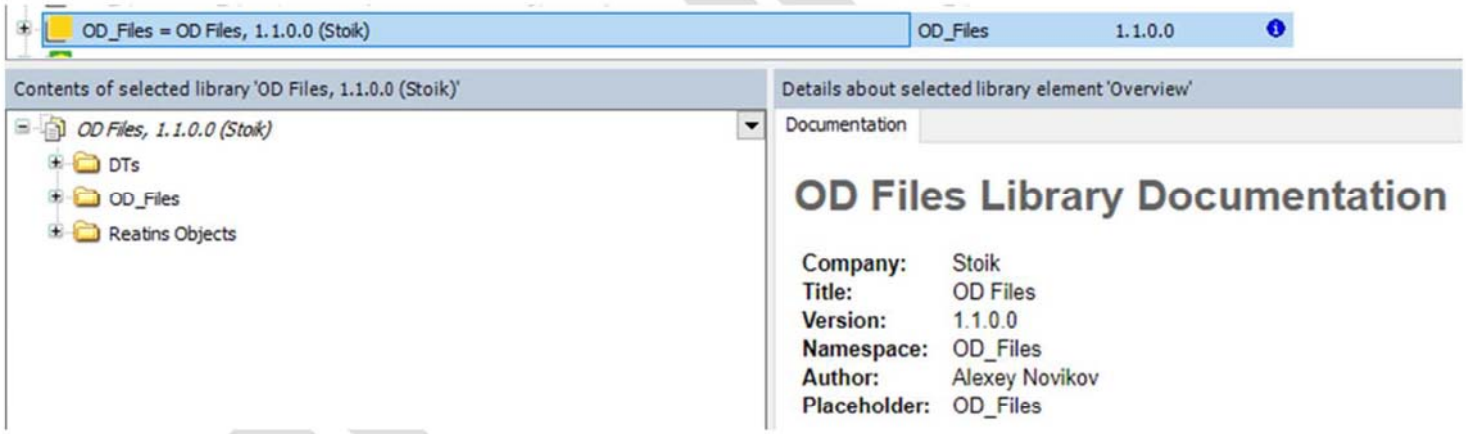
Библиотека **CAA File** позволяет осуществлять операции с папками и файлами.



а **CAA Types** содержит необходимые типы данных



Также, потребуется ФБ **OD_FileCopy** из библиотеки **OD_Files**, которая позволяет копировать удобным способом файлы из памяти контроллера на SD карту и обратно с возможностью создания при копировании новых папок и имени файла.



В библиотеке **OD_Files** содержится перечисление **FileDevice (ENUM)**, позволяющее удобным способом выбрать источник и приёмник файла:

FileDevice (ENUM)

TYPE FileDevice :

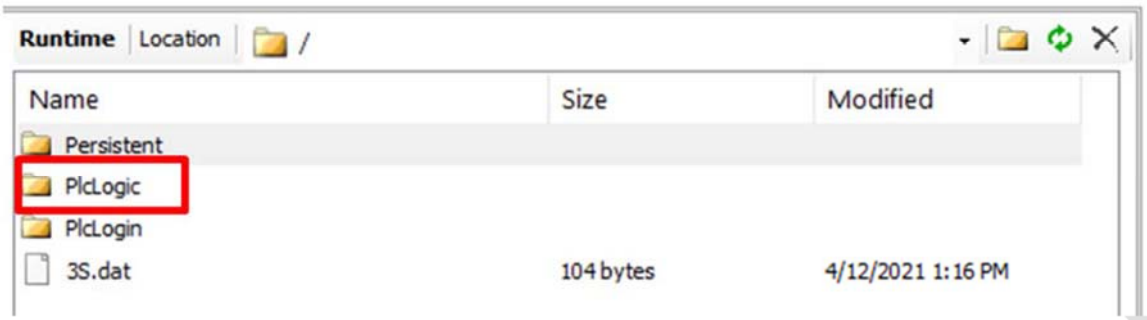
Файловое устройство

InOut:

Name	Initial	Comment
CDS_DIR	0	Рабочая директория PlcLogic CoDeSys. ВНИМАНИЕ: Для этой папки имена файлов/ папок на кириллице в Files(Файлы) отображаются НЕ КОРРЕКТНО!
SD	1	SD накопитель. Просим учитывать особенность MX300: Вставлять/вынимать SD карту можно только при выкл.питании контроллера!
CDS_DFD	2	Пользовательская директория CoDeSys (Data File Directory): см.Data file в IEC Controller file management

В наших примерах далее будут задействованы две именованные константы из данного перечисления:

CDS_DIR – внутренняя память контроллера, папка **PLcLogic**



SD – вставленная в контроллер SD карта

Для процедуры копирования файлов используется ФБ **OD_FileCopy (FB)**, который берёт файл по указанному пути (папки и файл должны существовать) и помещает по указанному пути. Причём в приёмнике будут созданы указанные в пути папки, если они не существовали, а файл при копировании может быть переименован:

OD_FileCopy (FB)

FUNCTION_BLOCK OD_FileCopy

Копирование файла В данном ФБ применяются следу
CDS_DFD) готово к работе, то папка(-и) будет(-ут) соз

InOut:

Scope	Name	Type
Input	xExecute	BOOL
	eFileDevSource	FileDevice
	sDirPathSource	STRING(120)
	sFileNameSource	STRING
	eFileDevDest	FileDevice
	sDirPathDest	STRING(120)
	sFileNameDest	STRING
Output	xDone	BOOL
	xBusy	BOOL
	xError	BOOL
	eERROR	ERROR
	dwFileSize	__XWORD

Ножка ФБ	Описание
xExecute -	Выполнить ФБ
eFileDevSource	Источник, откуда копируется файл. Память контроллера или SD карта
sDirPathSource	Папка (папки) откуда копируется файл. Разделитель ‘/’
sFileNameSource’	Укажите имя файла в источнике с расширением. (без символов ‘/’)
eFileDevDest	Приёмник, куда копируется файл. Память контроллера или SD карта
sDirPathDest	Папка (папки) куда копируется файл. Разделитель ‘/’
sFileNameDest	Укажите имя файла в приёмнике с расширением. (без символов ‘/’)
xDone	Флаг окончания процедуры копирования файла
xBusy	Флаг работы ФБ
xError	Флаг ошибки копирования файла
eERROR	Код ошибки
dwFileSize	Размер скопированного файла в байтах

Для демонстрации процедуры доступа на SD карту для чтения-записи файлов необходимо последовательно выполнить следующие действия (в скобках указаны используемые библиотеки):

Для записи файла на SD карту:

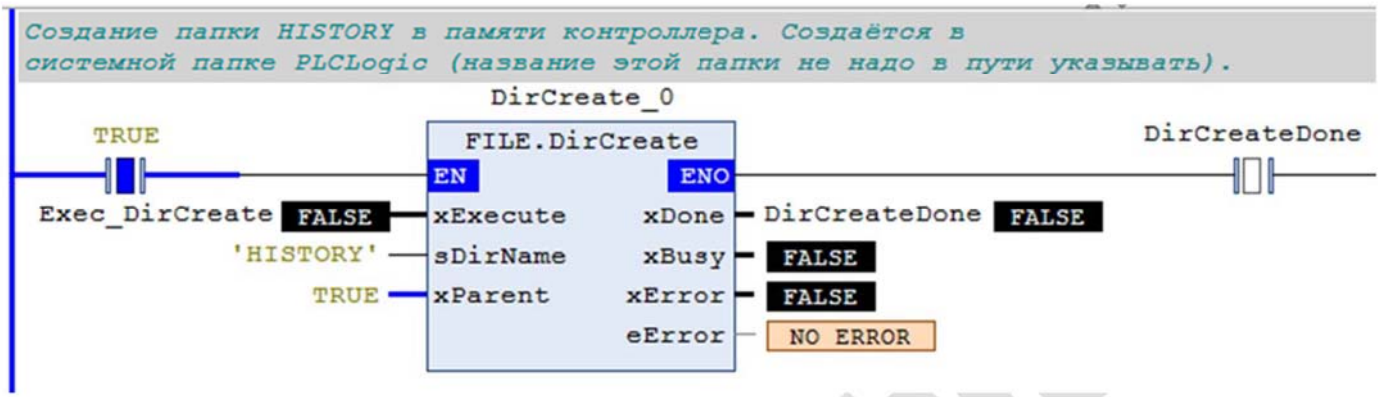
1. Создать папку в памяти контроллера в системной папке PLcLogic (CAA File)
2. Создать файл в этой папке (CAA File)
3. Открыть файл (CAA File)
4. Записать данные в файл в виде строковой переменной (программный код)
5. Закрыть файл (CAA File)
6. Скопировать файл на SD карту с созданием папки (OD_FileCopy)

Для чтения файла с SD карты:

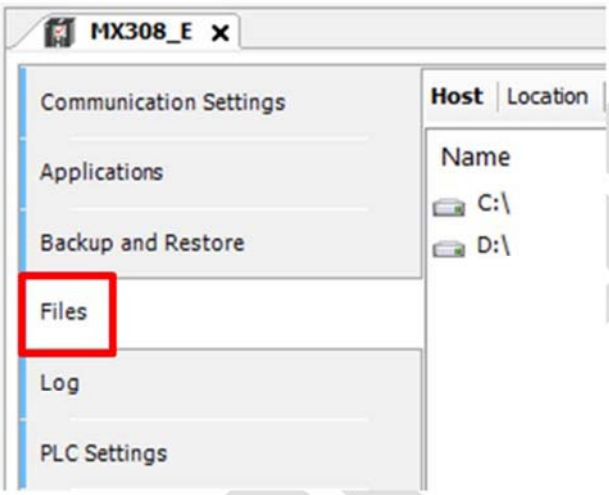
1. Создать на ПК папку и файл на SD карте (пользователь на ПК)
2. Скопировать файл с SD карты в память контроллера с созданием папк (OD_FileCopy)
3. Посмотреть через среду разработки, что файл записался в память контроллера (SoftWare)

Копирование файлов из памяти контроллера на SD карту

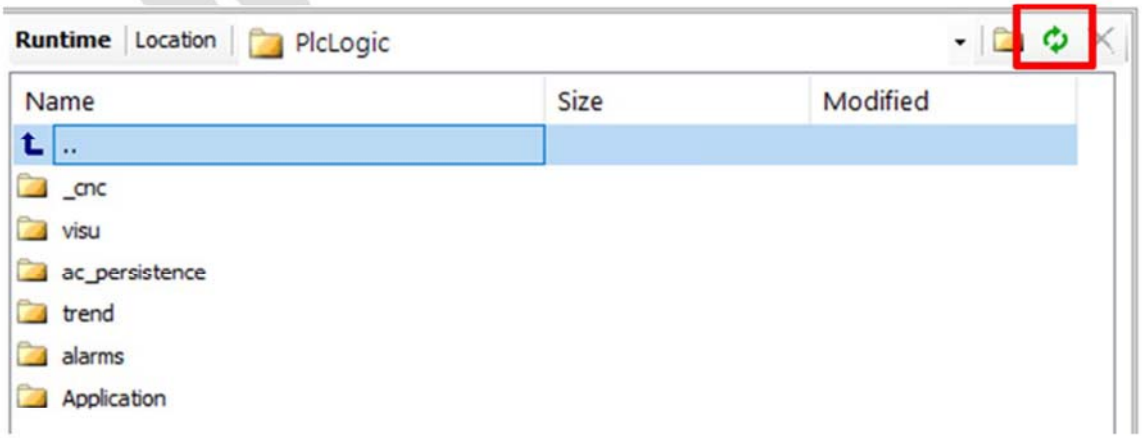
Создайте папку HISTORY в памяти контроллера при помощи ФБ **FILE.DirCreate**. Папка будет создана в системной папке **PLcLogic**. Название папки указывается в прямых кавычках (как для всех строковых переменных).



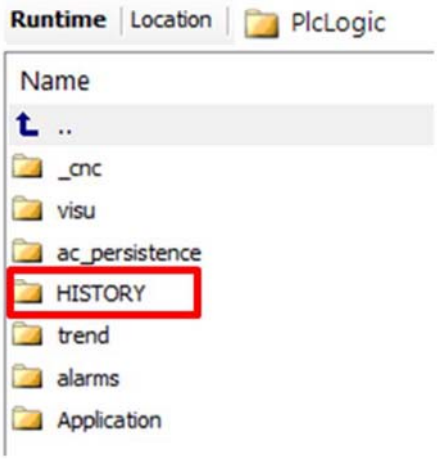
Создание папки можно проверить через среду программирования. Для этого откройте вкладку **Device** и выберите пункт **Files**:



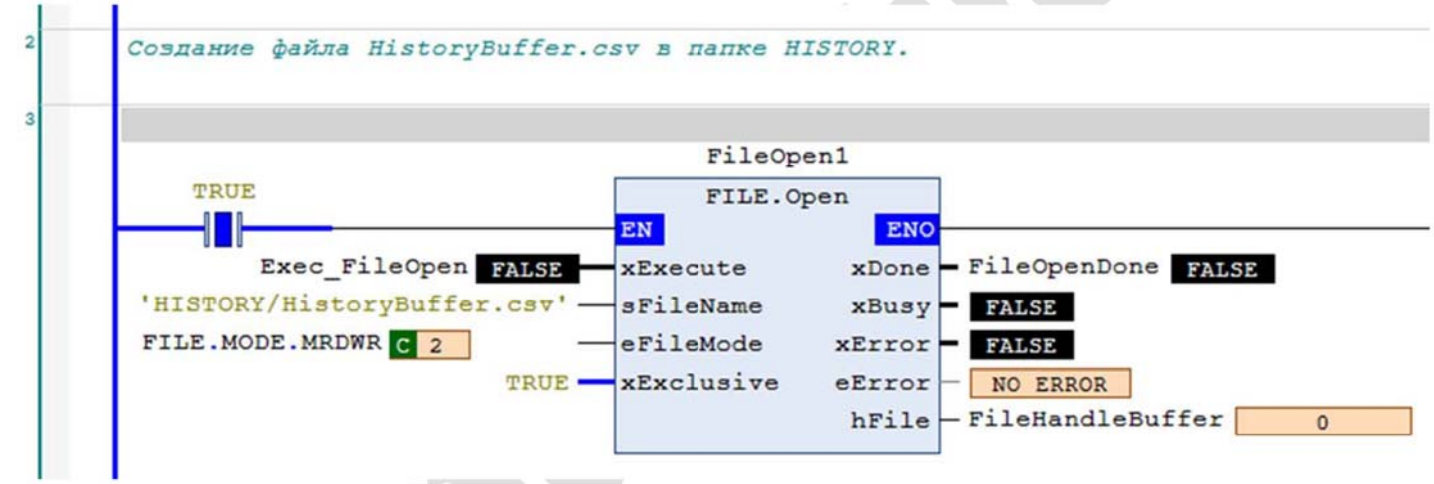
В правой части вкладки нажмите кнопку обновления информации, откройте папку **PLcLogic** и снова обновите:



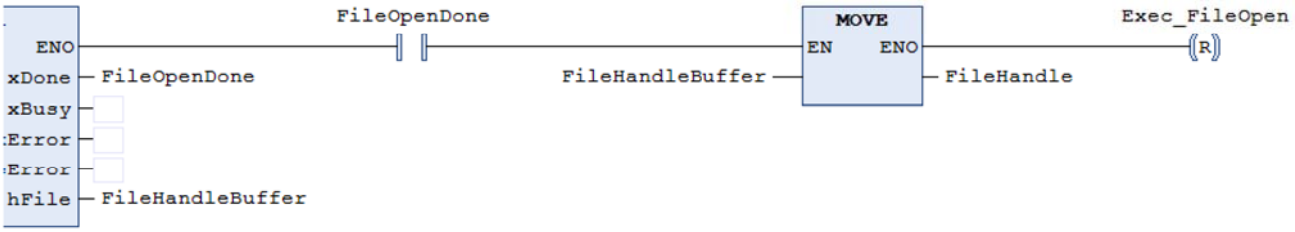
В папке PLcLogic должна быть видна созданная папка HISTORY:



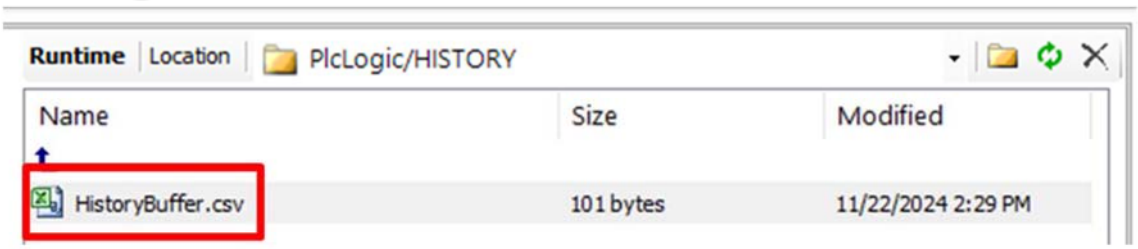
Далее создайте файл HistoryBuffer.csv в папке HISTORY при помощи ФБ **FILE.Open**.



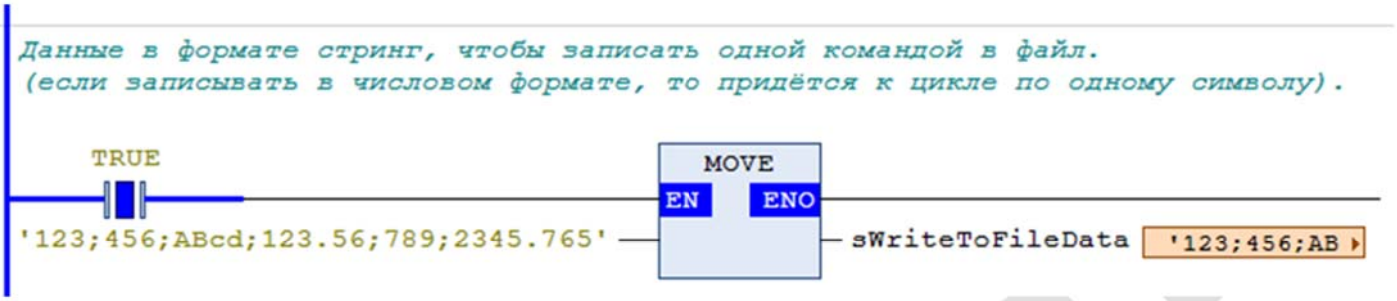
Для сохранения дескриптора файла **hFile** по ножке **xDone** переместите его в другую переменную:



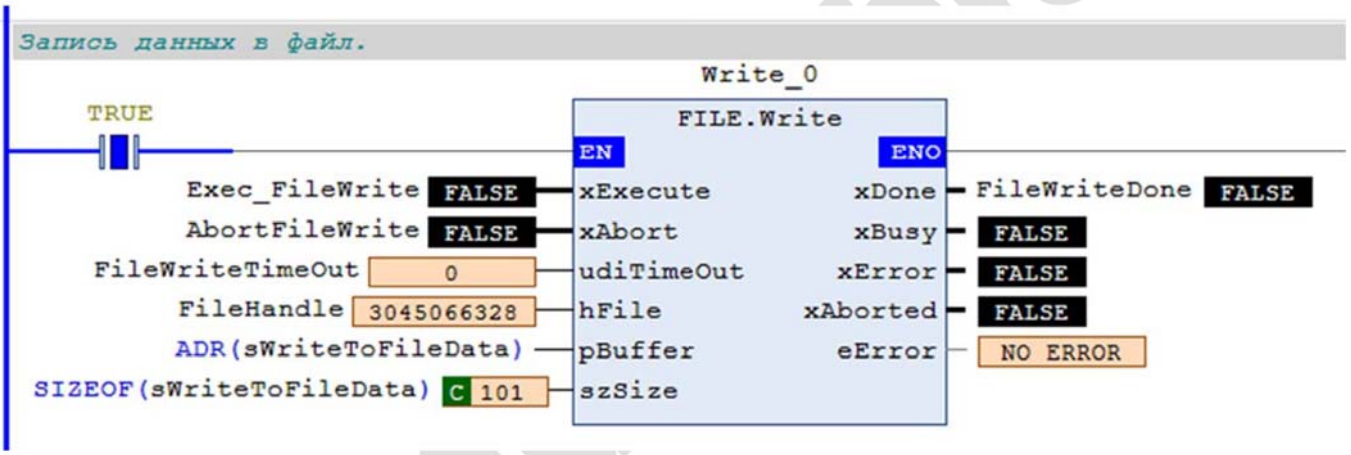
Проверить создание файла можно также в среде программирования обновив содержимое:



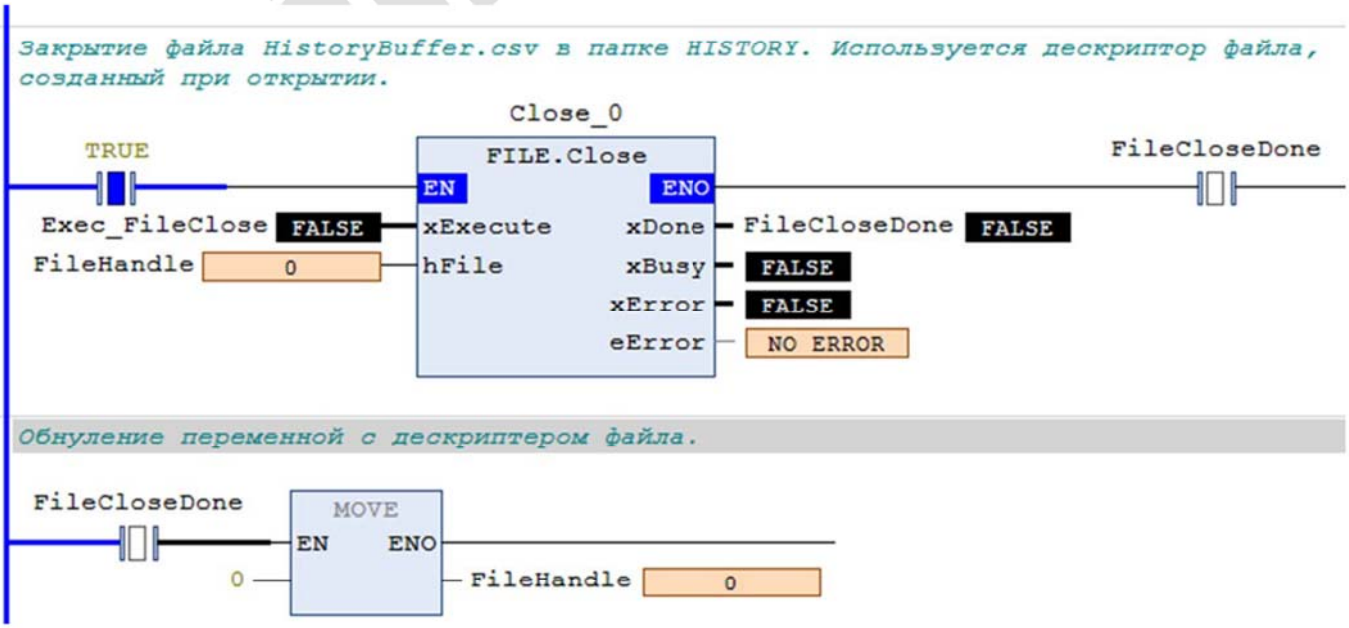
Далее запишите данные в строковую переменную, например просто скопировав строковую константу в строковую переменную.
(в данной Главе не рассматриваются операции с данными для записи в файл, поэтому используется простейший и кратчайший способ исключительно для целей демонстрации)



Переместите данные из переменной в файл при помощи ФБ **FILE.Write**:



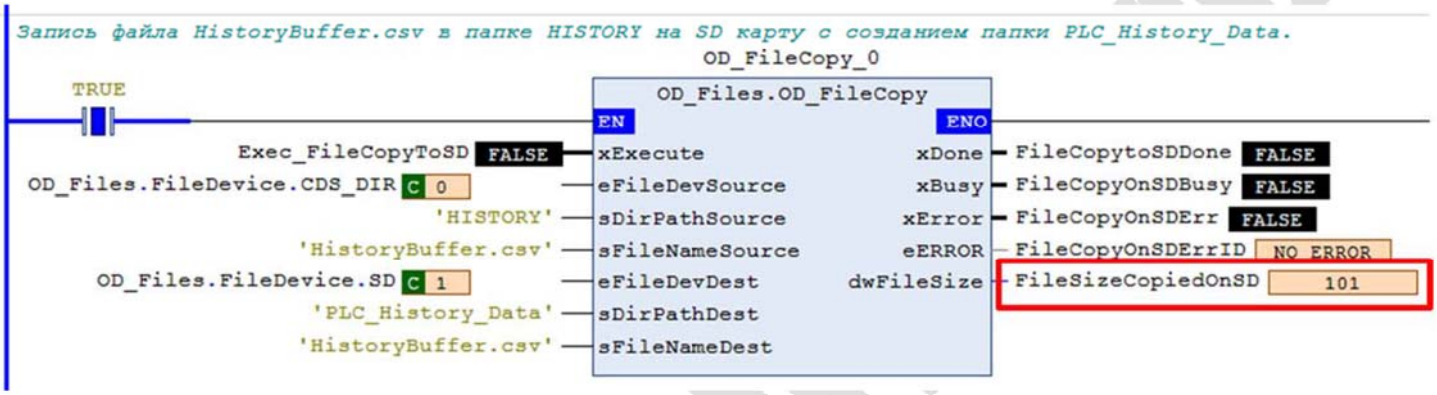
Закройте файл при помощи ФБ **FILE.Close** и обнулите переменную, хранящую дескриптор файла:



Вставьте SD карту в слот на контроллере. Карта должна иметь файловую систему FAT32 и объем не более 32 Гб.

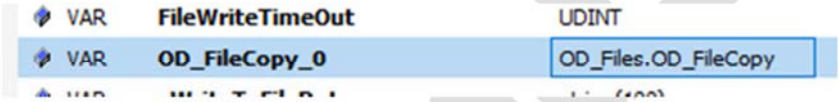
Тип:	Съемный диск
Файловая система:	FAT32

При помощи ФБ **OD_FileCopy** скопируйте файл HistoryBuffer.csv из папки HISTORY на SD карту с созданием папки PLC_History_Data. Название файла не меняется, но при необходимости при копировании название файла можно задать новое. Содержимое при этом останется как у исходного файла.

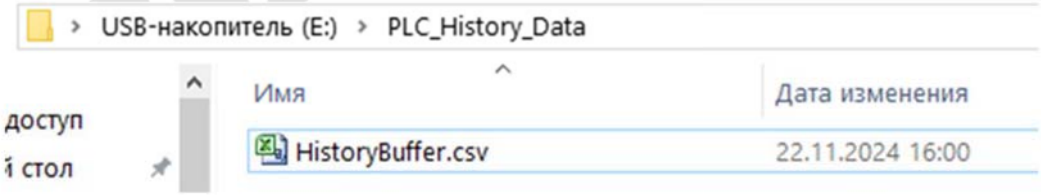


На ножке **dwFileSize** будет отображаться размер записанного файла в байтах.

При объявлении экземпляра ФБ **OD_FileCopy** и фактических переменных для него необходимо использовать при определении типа переменной имя библиотеки **OD_Files**:

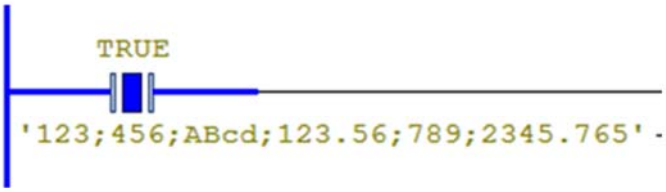


Для проверки записи файла извлеките SD карту из контроллера и вставьте в картовод на ПК. В проводнике можно будет увидеть в корне карты папку и в ней файл:

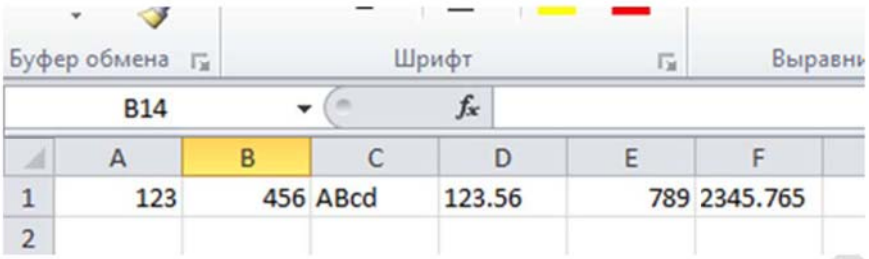


Запустите файл через Excel для отображения содержимого файла, которое будет соответствовать записанным в программе данным.

Программа:



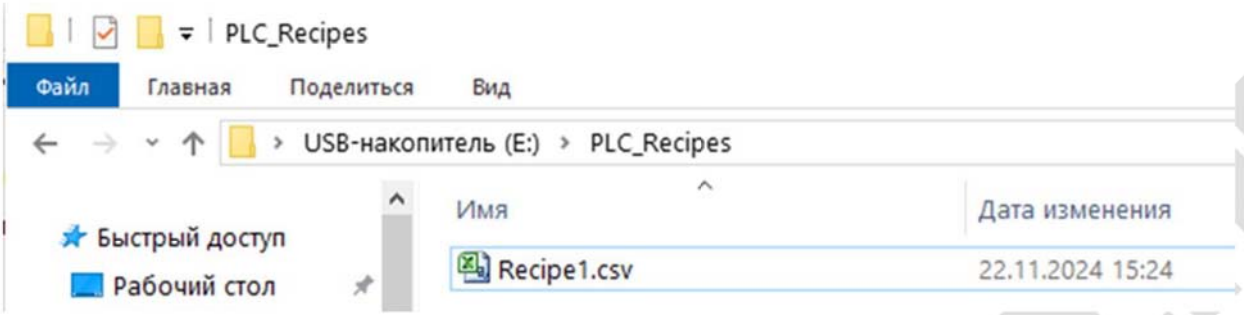
Содержимое файла:



Копирование файлов с SD карты в память контроллера

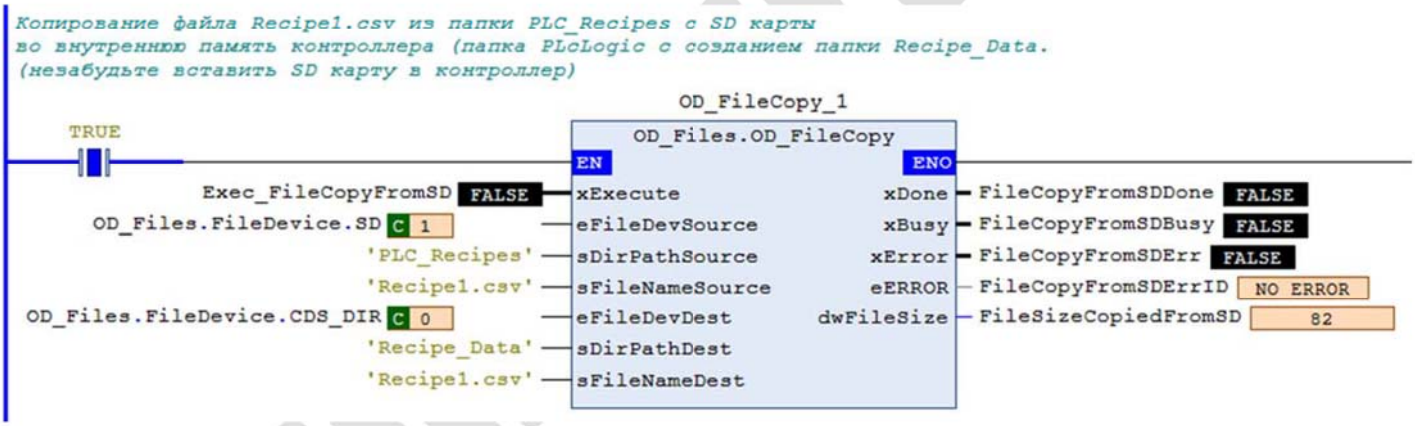
Вставьте в картовод на ПК SD карту. Формат файловой системы должен быть FAT32 и объем не более 32 Гб.

Запишите средствами на карту средствами ПК папку PLC_Recipe и файл Recipe1.csv:

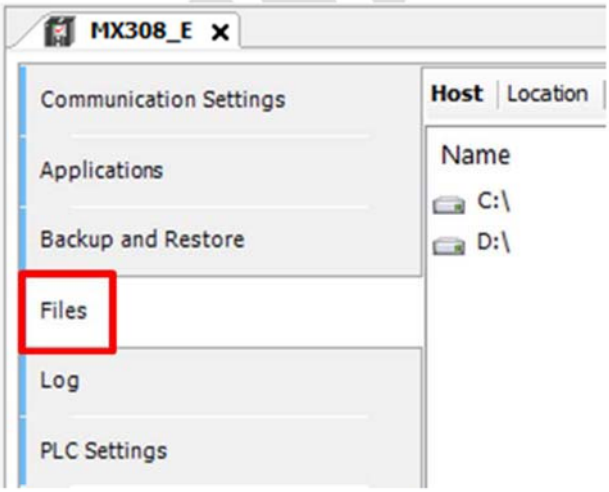


Вставьте карточку в слот на контроллере.

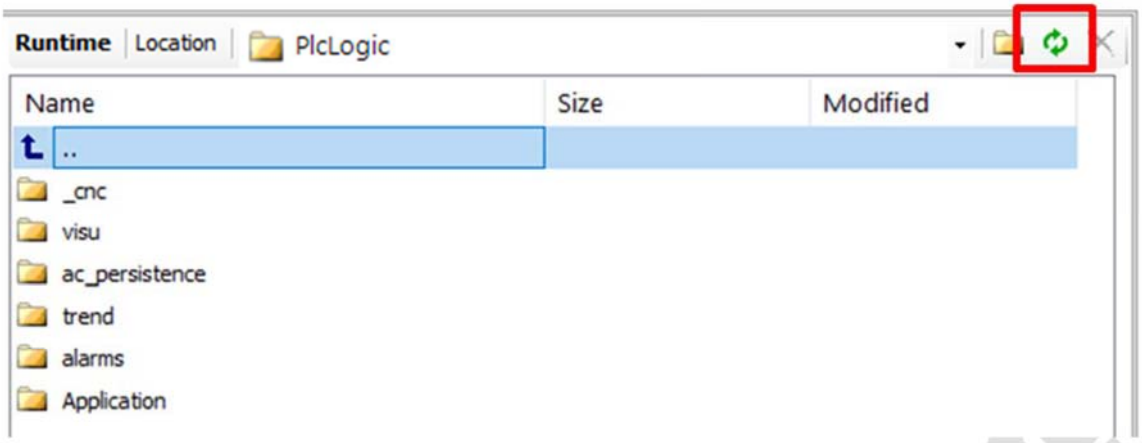
При помощи ФБ **OD_FileCopy** скопируйте папку PLC_Recipe и файл Recipe1.csv во внутреннюю память контроллера (папка **PLcLogic**) с созданием папки Recipe_Data.



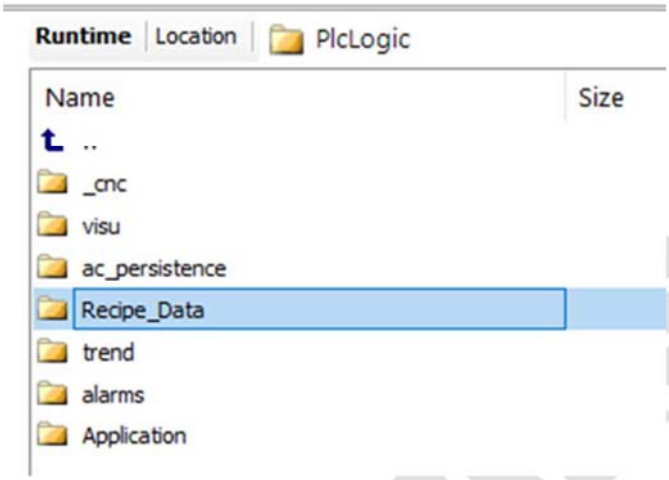
Создание папки можно проверить через среду программирования. Для этого откройте вкладку **Device** и выберите пункт **Files**:



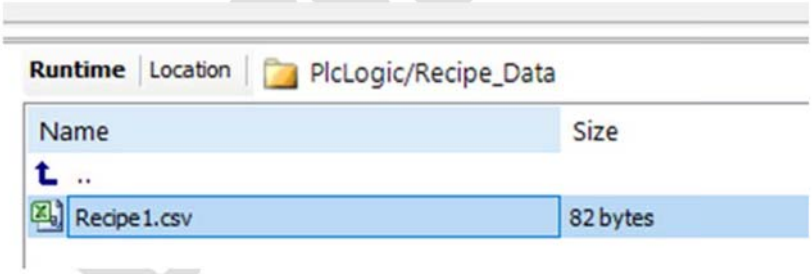
В правой части вкладки нажмите кнопку обновления информации, откройте папку **PLcLogic** и снова обновите:



В папке PLcLogic должна быть видна созданная папка Recipe_Data:



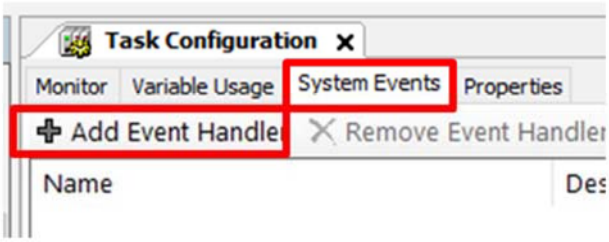
и в ней файл Recipe1.csv:



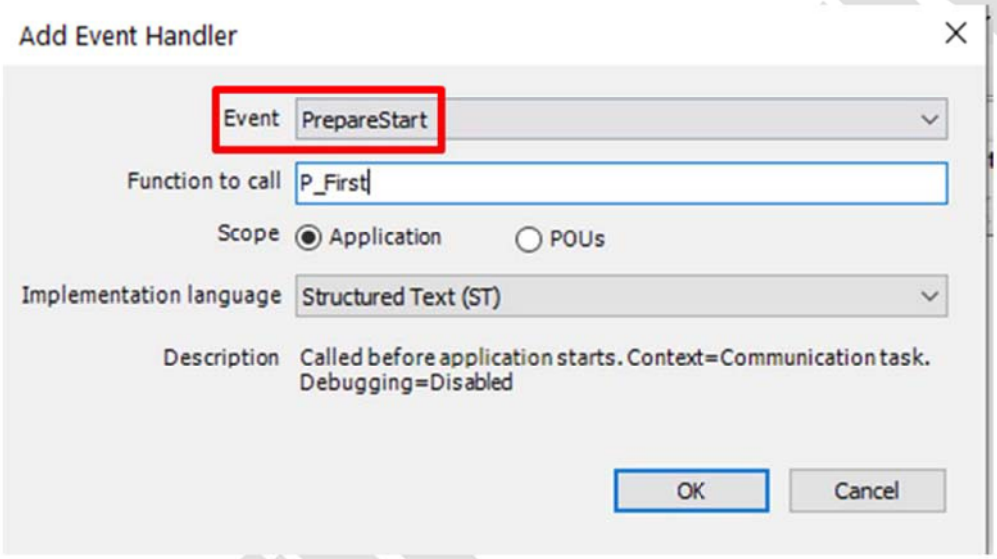
Импульс при переводе в состояние RUN

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.

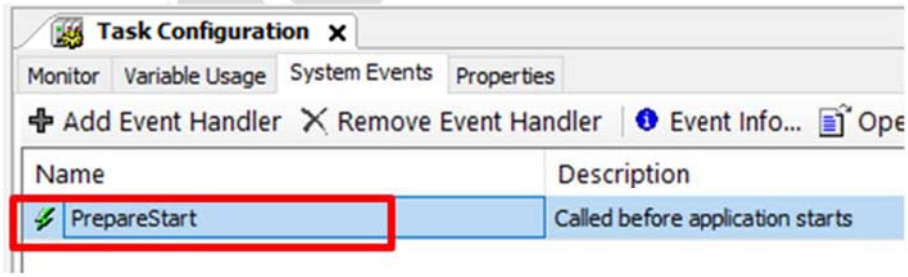
Для этого встаньте в древе проекта на пункт **Task Configuration** и двойным щелчком левой кнопки мышки откройте вкладку **Task Configuration** и выберите пункт **System Events** и нажмите кнопку **Add Event Handler**:



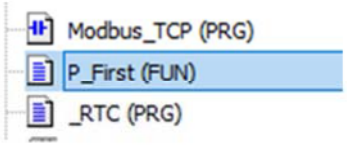
В открывшемся окне выберите тип события **Event => PrepareStart**, дайте имя, например **P_First** и нажмите **OK**.



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:

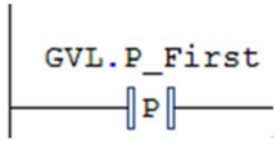
```
1  GVL.P_First:=TRUE;
2
```

В таблице переменных система автоматически уже создала переменную:

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

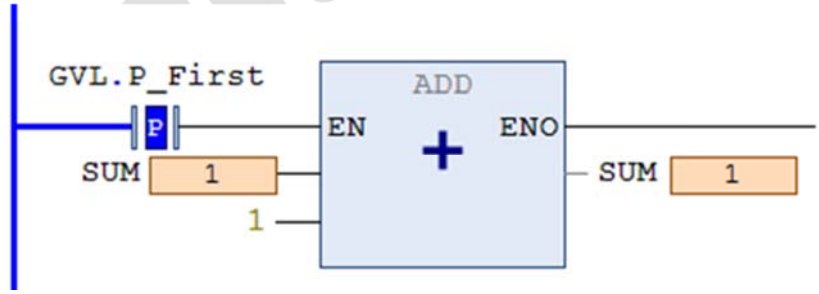
никаких других переменных в функции-обработчике создавать не надо!

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

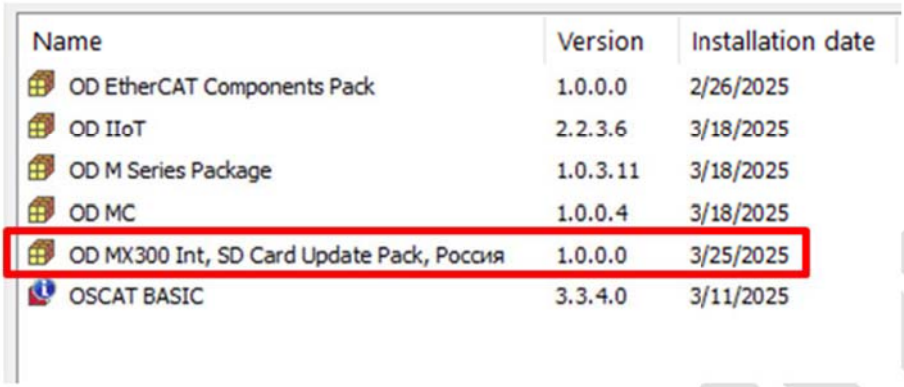


Переменная SUM будет равна 1 на всё время работы контроллера.

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. Контроллеры серии MX300 поддерживают включение выхода по достижению порога сравнения на всех шести высокоскоростных счётчиках (0-5), а привязку Задачи, вызываемую по событию сравнения, поддерживают четыре высокоскоростных счётчика (0-3).

Для доступности данной функции необходимо установить дополнительный пэкидж **OD MX300 Int, SD Card Update Pack 1.0.0.0.package** (или выше версией).



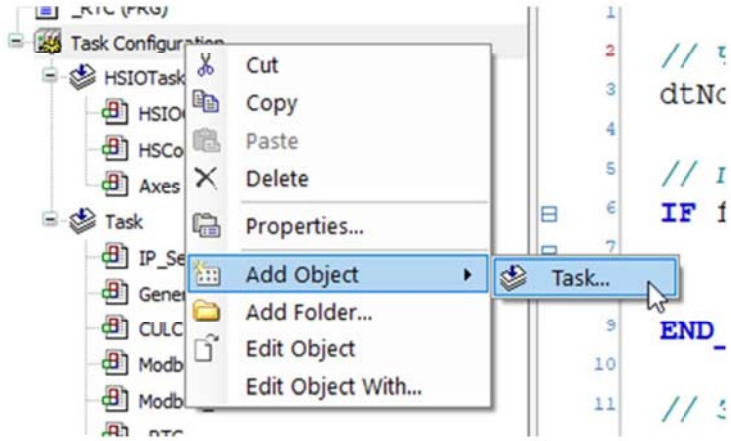
Кроме того, для ЦПУ MX308/316 выпуска 2024 года необходимо обновить встроенное ПО (firmware).

Внимание!
Процедура обновления встроенного ПО выполняется только в офисе компании «Оптимус Драйв». Поэтому перед отгрузкой контроллеров со склада необходимо заказать через Отдел продаж обновление встроенного ПО.

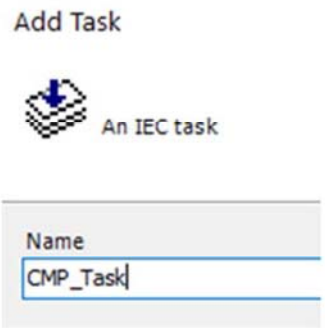
Контроллеры выпуска 2025 года уже имеют необходимую версию встроенного ПО и не требуют обновления.

Для реализации высокоскоростного сравнения в древо проекта контроллера должен быть добавлен адаптер входов-выходов на ЦПУ – **MXHighSpeedIO** (см. соответствующую Главу данного Руководства). Это необходимо для организации высокоскоростного счётчика, текущее значение которого и будет сравниваться с заданным порогом. Создайте счётчик **Counter2** (см. Работа с высокоскоростными счётчиками, стр. 132 настоящего Руководства).

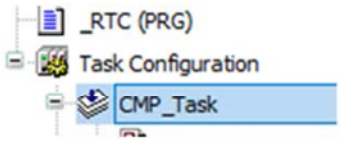
При необходимости вызова задачи при достижении условия высокоскоростного сравнения нужно создать Задачу (Task) обработки прерывания. Для этого встаньте в древе проекта на пункт **Task Configuration** и правой кнопкой мышки вызовите меню, в котором выберите пункт **Add Object – Task**:



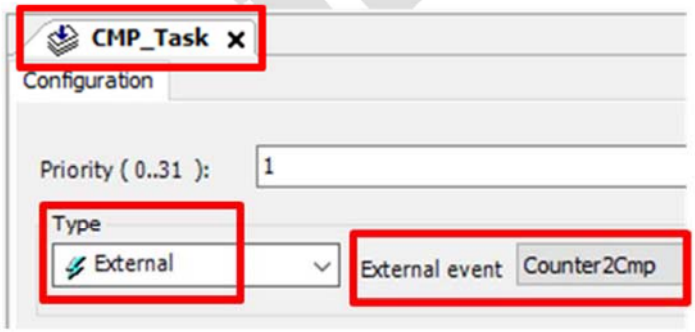
Дайте название Задаче, например **CMP_Task**.



В древе проекта появится Задача **CMP_Task**:

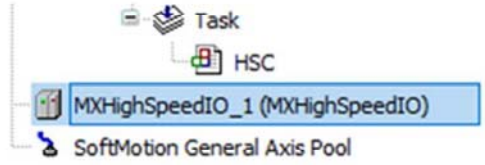


Щёлкните дважды левой кнопкой мышки на пункте **CMP_Task**, откроется вкладка с настройками параметров Задачи. Выберите тип задачи **External** и назначьте счётчик, в нашем примере **Counter2Cmp**:

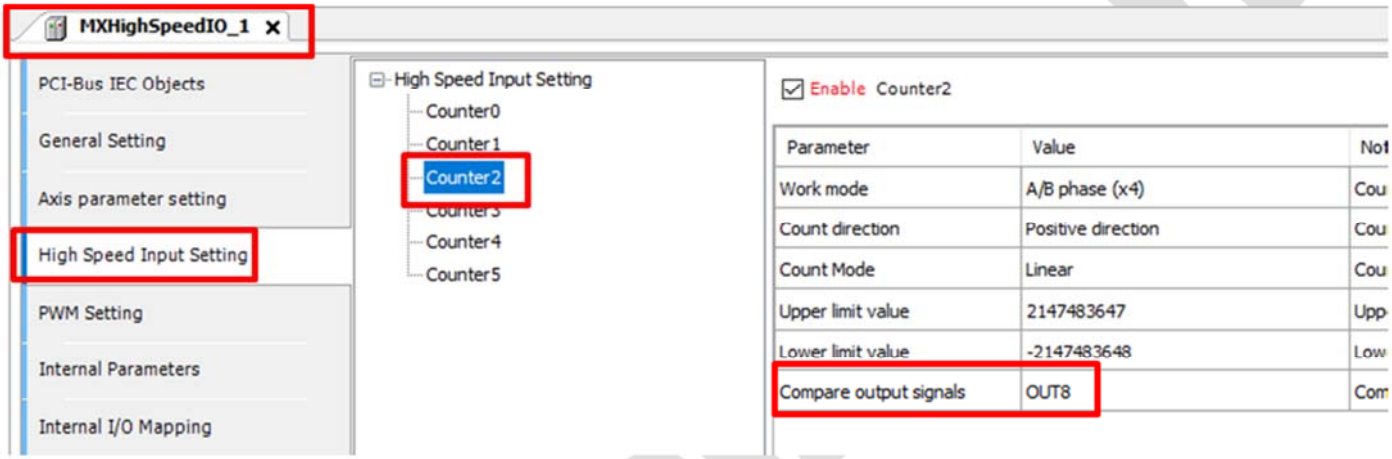


Данная задача будет вызываться при достижении порога сравнения счётчиком **Counter2**.

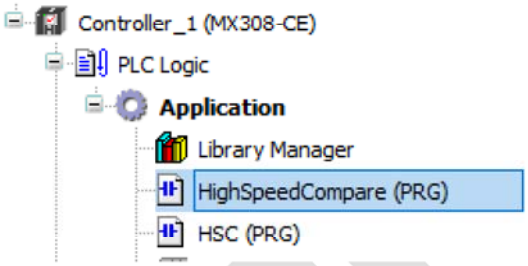
Далее щёлкните дважды левой кнопкой мышки на пункте **MXHighSpeedIO** в древе проекта:



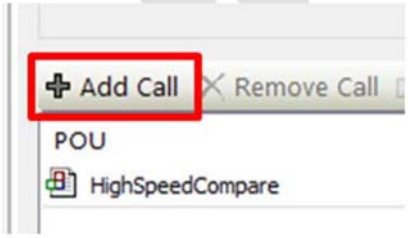
Откроется вкладка с настройками, в которой необходимо выбрать раздел **High Speed Input Setting – Counter2** и в параметре **Compare Output Signals** выбрать выход, который будет включаться при наступлении события прерывания по достижению порога высокоскоростного счёта. На рисунке ниже выбран выход 08 (**OUT8**):



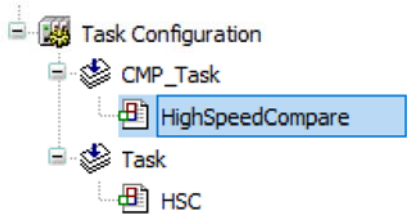
Далее создайте программный блок, например **HighSpeedCompare**:



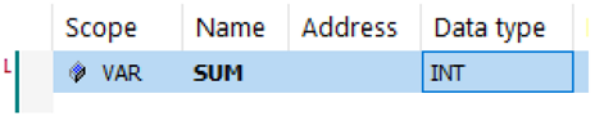
и привяжите его к Задаче **CMP_Task**:

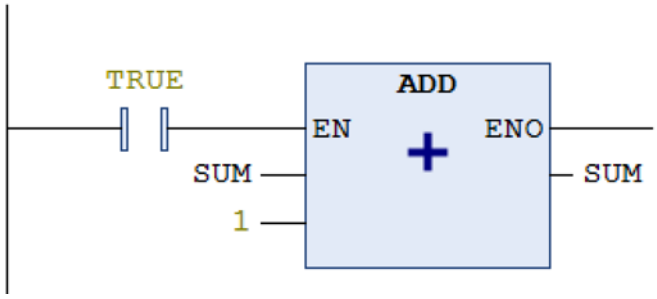


В древе проекта появится пункт:



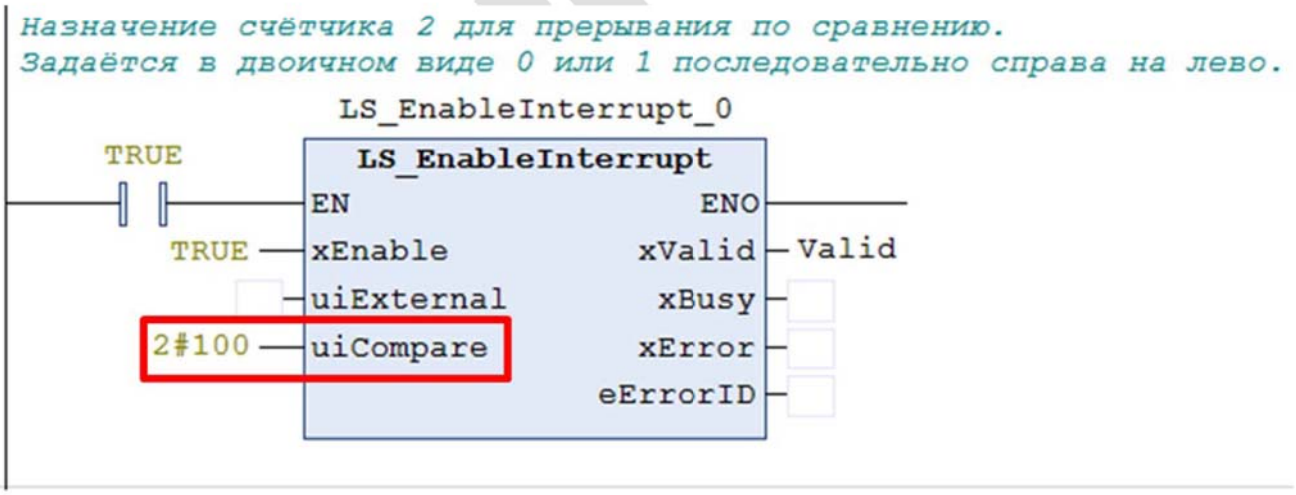
Далее в программном блоке HighSpeedCompare для целей демонстрации можно создать простой код, который будет исполняться только по наступлению события достижения порога высокоскоростного счёта:



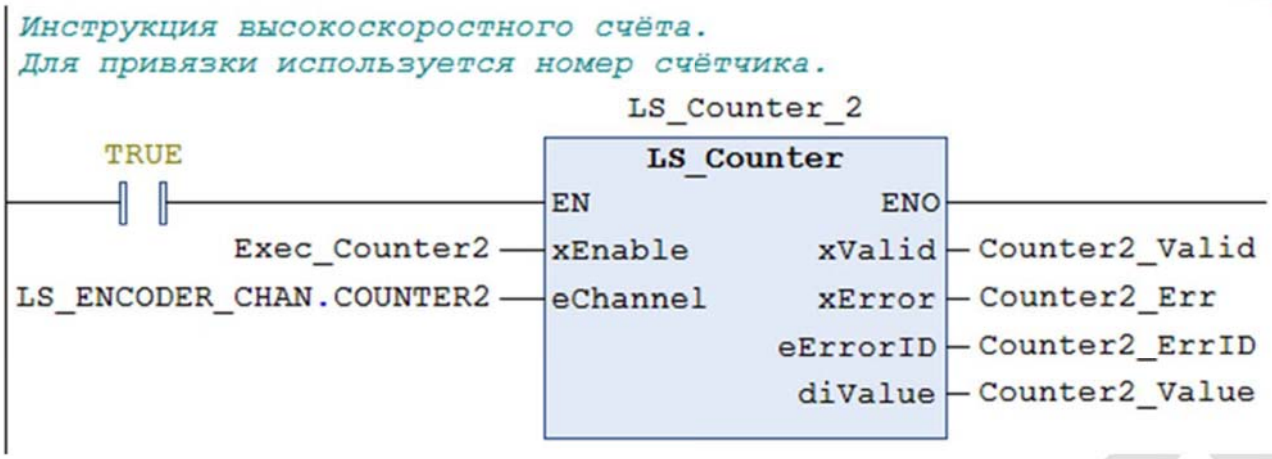


(если бы данная команда стояла в циклической Задаче, то исполнялась бы в каждом скане, а так только по наступлению события).

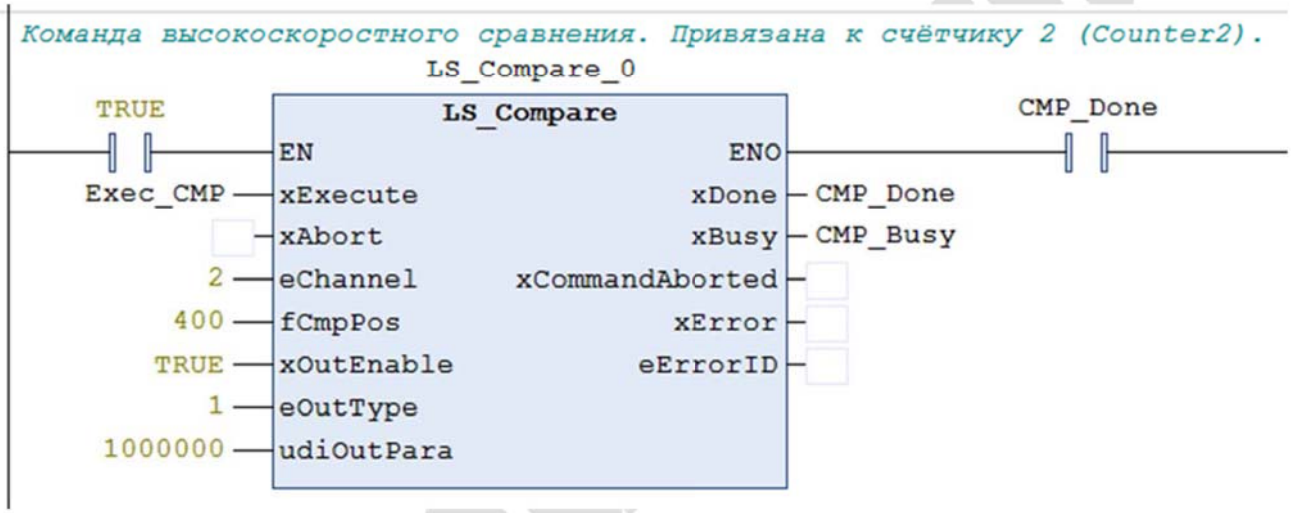
Прерывание необходимо зарегистрировать в команде **LS_EnableInterrupt** в двоичном коде по номеру счётчика:



Также, в программе в основной циклической Задаче должна стоять инструкция обращения к высокоскоростному счётчику **LS_Counter**. На ножке **diValue** будет текущее значение счётчика, которое инструкция **LS_Compare** и будет сравнивать с заданным порогом.



Далее в основной циклической задаче необходимо поставить команду **LS_Compare** и привязать по ***номеру*** счётчик:



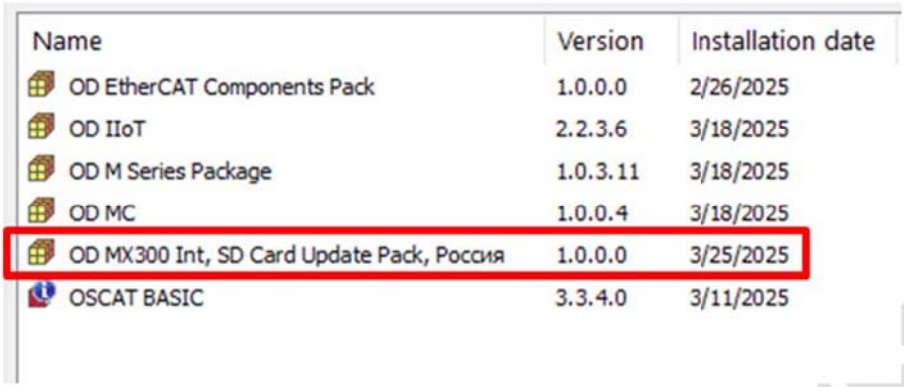
Описание основных ножек ФБ:
xExecute – запускает ФБ (необходимо взводить заново каждый раз после срабатывания ФБ)
eChannel – номер счётчика (0-5). Можно использовать структуру типа LS_ENCODER_CHAN.COUNTER**
fCmpPos – порог для сравнения. Количество импульсов в user units LREAL
xOutEnable – разрешение установки выхода при достижении порога, поставить TRUE
eOutType – режим сравнения, поставить 1 (выход включен на установленное время по наступлению события)
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. Причём сброс счётчика происходит в следующем скане (или даже через несколько сканов).

Прерывание по сигналу на входе контроллера

Контроллеры типа MX300 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 доступности данной функции необходимо установить дополнительный пэкидж **OD MX300 Int, SD Card Update Pack 1.0.0.0.package** (или выше версией).



Name	Version	Installation date
OD EtherCAT Components Pack	1.0.0.0	2/26/2025
OD IIoT	2.2.3.6	3/18/2025
OD M Series Package	1.0.3.11	3/18/2025
OD MC	1.0.0.4	3/18/2025
OD MX300 Int, SD Card Update Pack, Россия	1.0.0.0	3/25/2025
OSCAT BASIC	3.3.4.0	3/11/2025

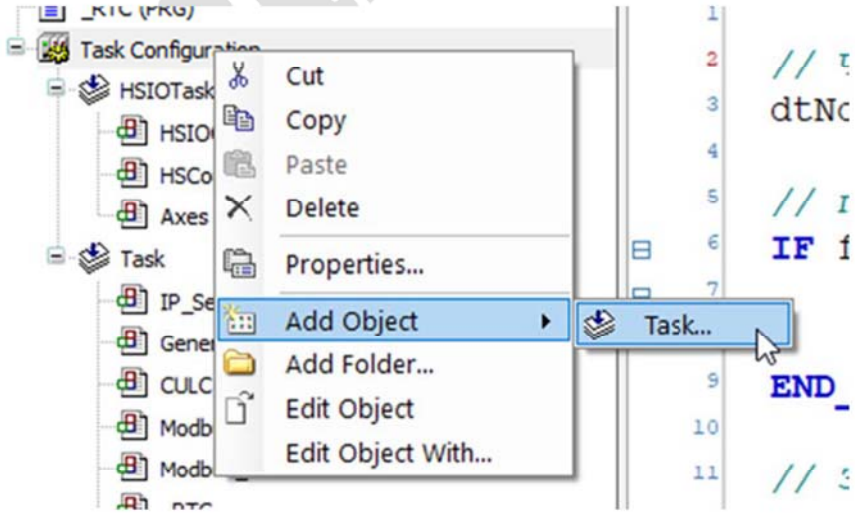
Кроме того, для ЦПУ MX308/316 выпуска 2024 года необходимо обновить встроенное ПО (firmware).

Внимание!
Процедура обновления встроенного ПО выполняется только в офисе компании «Оптимус Драйв». Поэтому перед отгрузкой контроллеров со склада необходимо заказать через Отдел продаж обновление встроенного ПО.

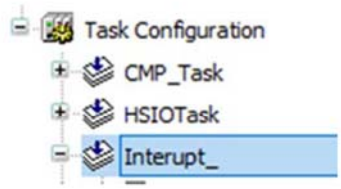
Контроллеры выпуска 2025 года уже имеют необходимую версию встроенного ПО и не требуют обновления.

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

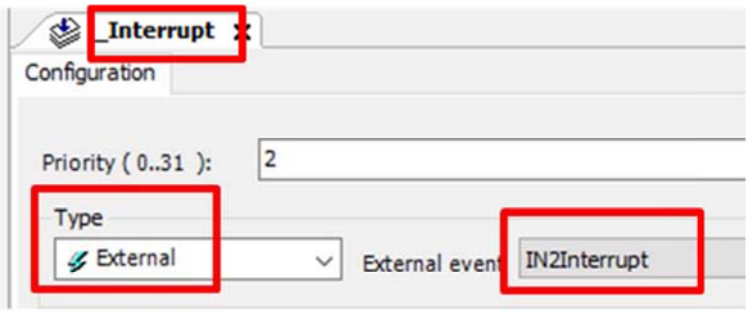
В начале необходимо создать Задачу (Task) обработки прерывания. Для этого встаньте в древе проекта на пункт **Task Configuration** и правой кнопкой мышки вызовите меню, в котором выберите пункт **Add Object – Task**:



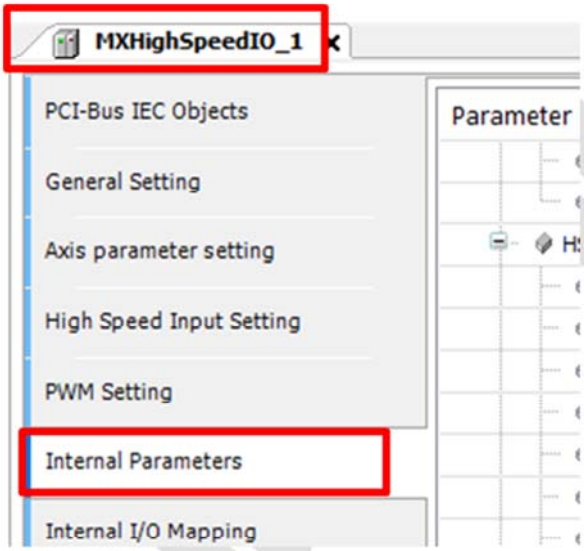
Дайте название Задаче, например **_Interrupt**. В древе проекта появится Задача **_Interrupt**:



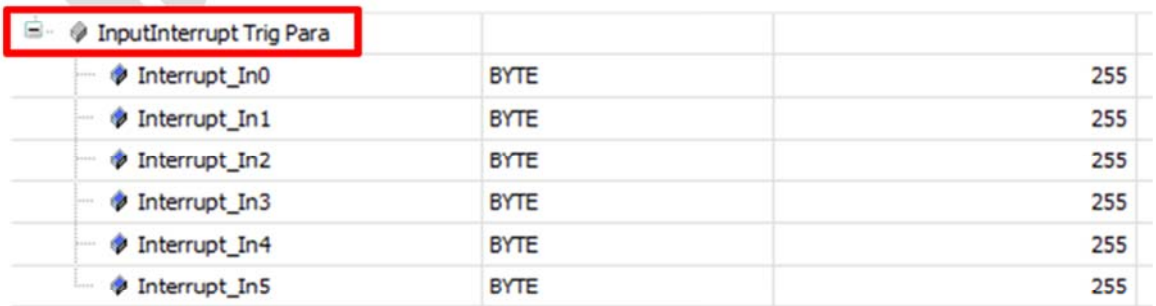
Щёлкните дважды левой кнопкой мышки на пункте **_Interrupt**, откроется вкладка с настройками параметров Задачи. Выберите тип задачи **External** и назначьте вход, например 02 (**IN2Interrupt**):



Щёлкните дважды на пункте **MXHighSpeedIO** и в открывшейся вкладке выберите раздел **Internal Parameters** :



пункт **InputInterrupt Trig Para**:

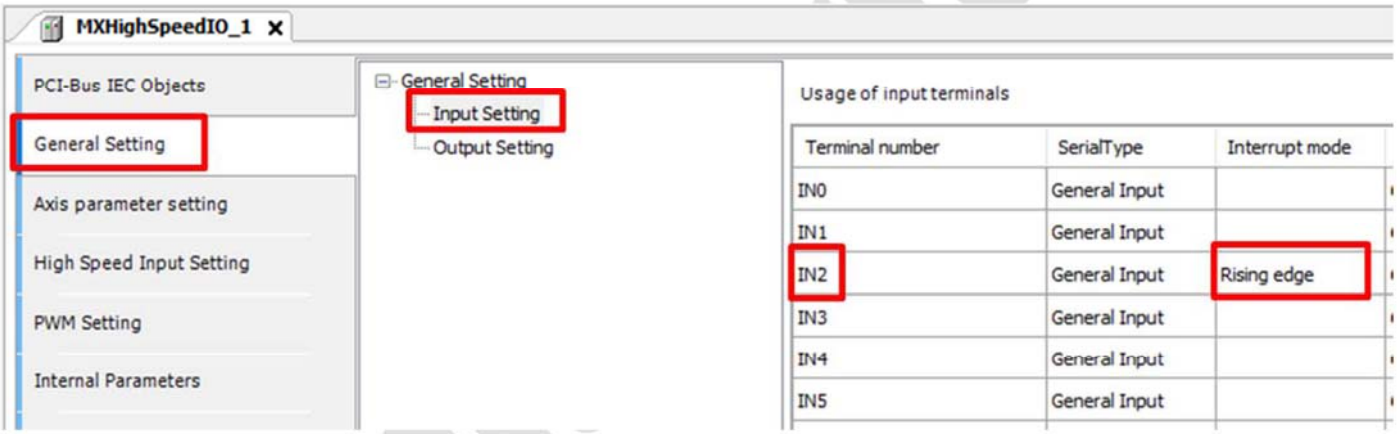


Выберите нужное прерывание (в нашем примере номер 2) и выставьте требуемый режим:

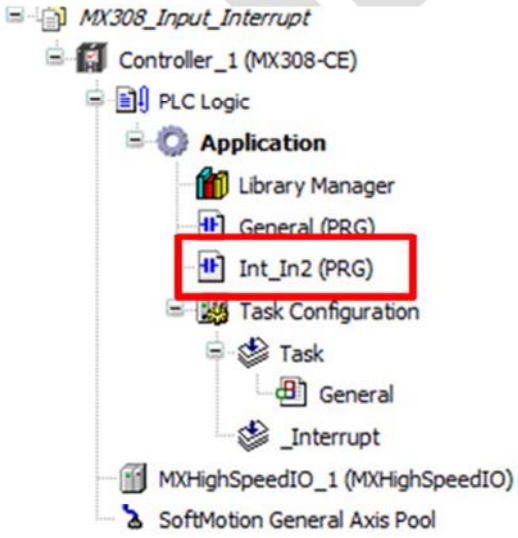
0 – Rising Edge (передний фронт)
1 - Falling Edge (задний фронт)
2 – Any Edge (передний и задний фронт)

Например передний фронт на входе 2:

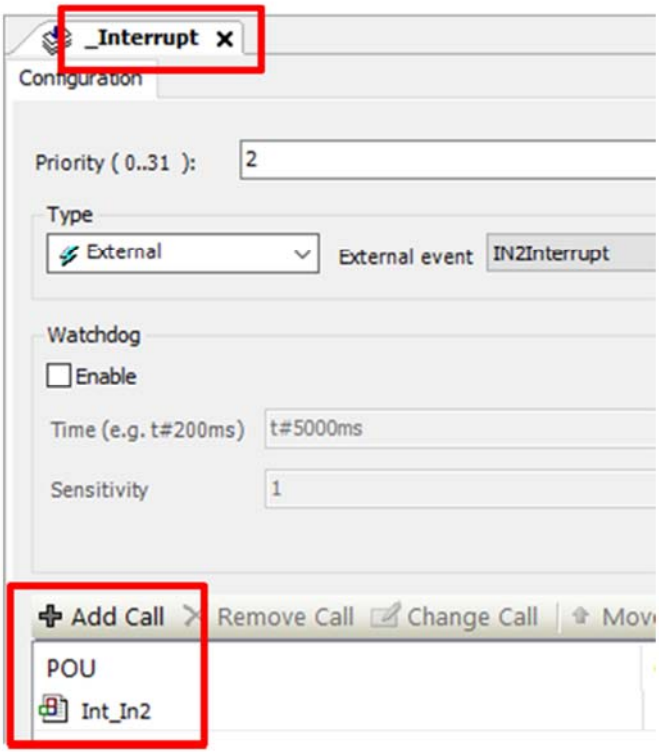
Во вкладке **General Setting – Input Setting**. Напротив входа IN2 появится выбранный фронт сигнала. На картинке ниже выбран передний фронт (**Rising edge**):



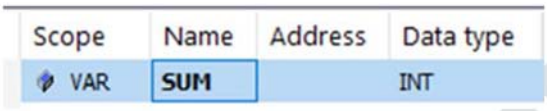
Далее создайте программный блок (POU), в котором необходимо поместить код, который должен быть исполнен по наступлению события, в нашем случае переднего фронта на входе 02. Дайте название, например **Int_In2**:

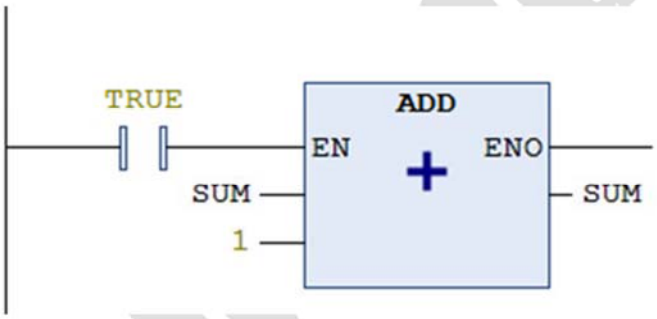


Созданный программный блок Int_In2 необходимо привязать к Задаче **_Interrupt**.



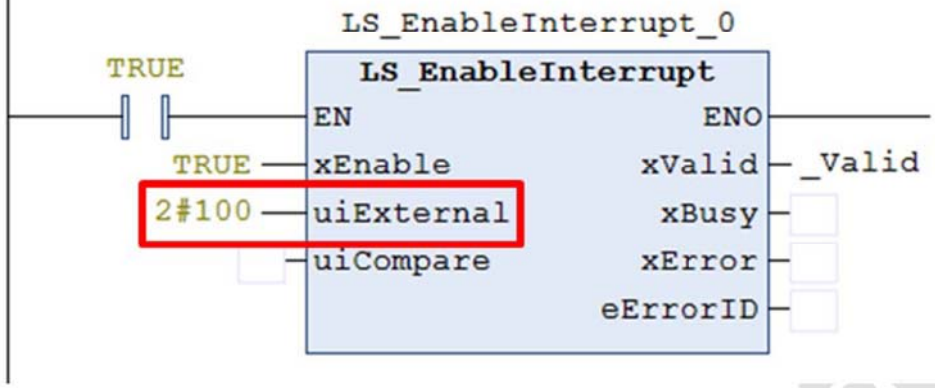
В теле POU **Int_In2** создайте одну строчку программы чисто в целях демонстрации работы прерывания:





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

```
Назначение входа 2 в качестве входа для прерывания.
Задаётся в двоичном виде по номеру входа справа налево.
Каждый вход на ЦПУ это бит.
Число 1 - вход задействован в прерывании
Число 0 - вход используется как обычный
(2#000000) - все входы 00-05 как обычные
(2#111111) - все входы 00-05 в режиме прерывания.
```



На ножке **iuExternal** 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 налево.

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.

Замена батарейки часов реального времени

В контроллерах типа MX300 используется батарейка CR2032 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. Заказной номер производителя **17100808** (CR2032 с коннектором).

Отсек с батарейкой находится на боковой плоскости контроллера в специальном отсеке под крышкой с надписью **BATTERY**:

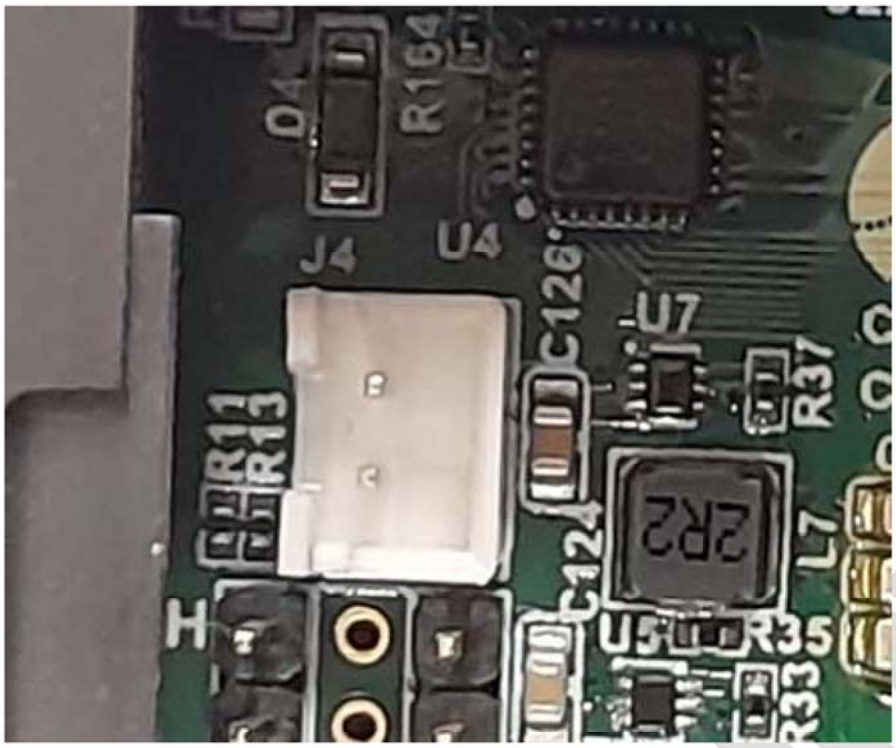
Для извлечения батарейки откройте крышку:

Батарейка с коннектором выглядит так:

Разъём питания шаг 2.54 мм, 2 контакта

провода приварены к батарейке:

На плате стоит ответная часть разъёма питания:



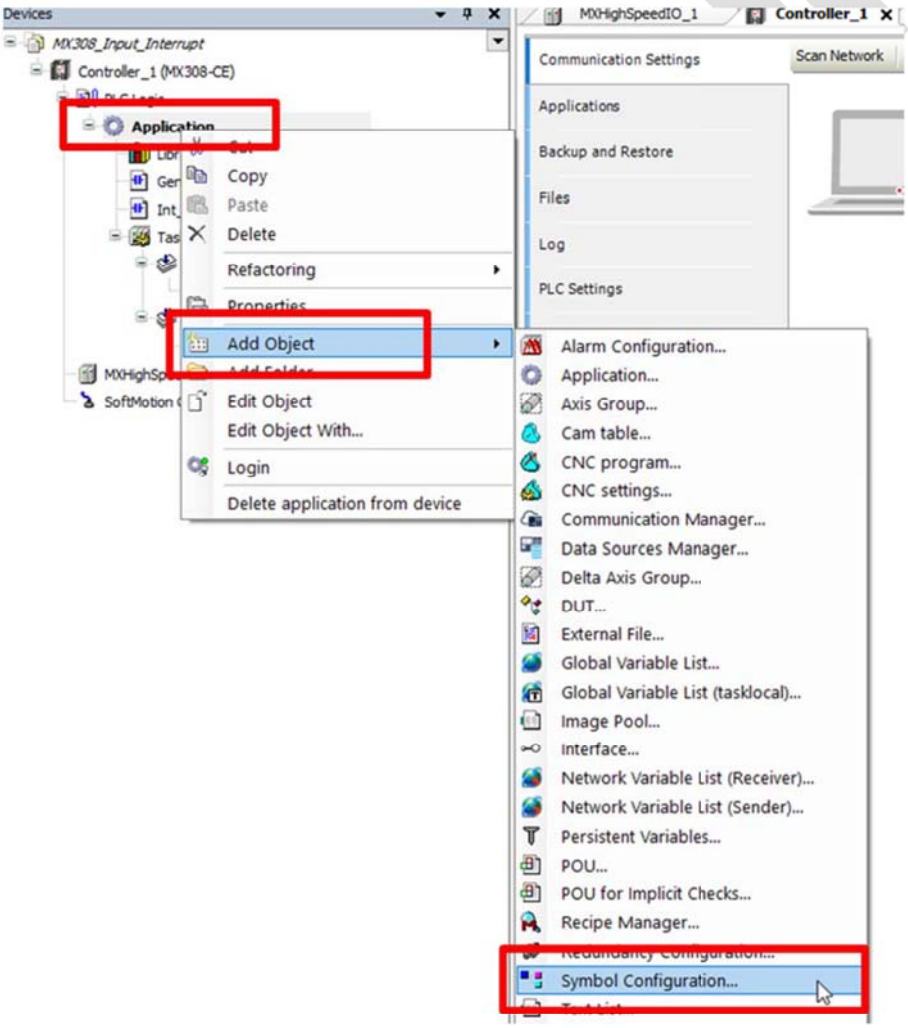
Обмен тегами CODESYS с панелью оператора

В среде программирования CODESYS существует стандарьтная 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 MX308-CE, в котором есть созданные переменные.

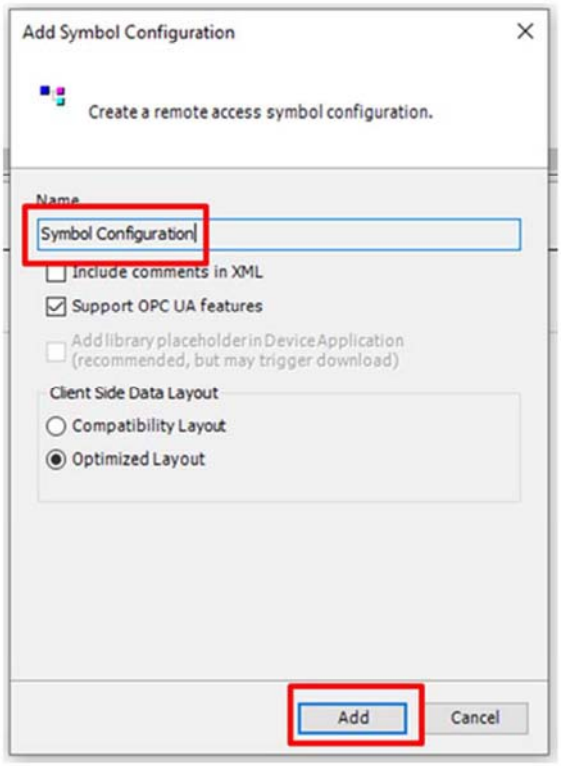
	Scope	Name	Address	Data type
2	VAR	**Coil1**		BOOL
3	VAR	**Int1**		INT
4	VAR	**Int2**		INT
5	VAR	**rARRAY**		ARRAY[0..2999] OF REAL
6	VAR	**Byte1**		BYTE
7	VAR	**Byte2**		BYTE
8	VAR	**str1**		string(20)
9	VAR	**wstr**		wstring(20)

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

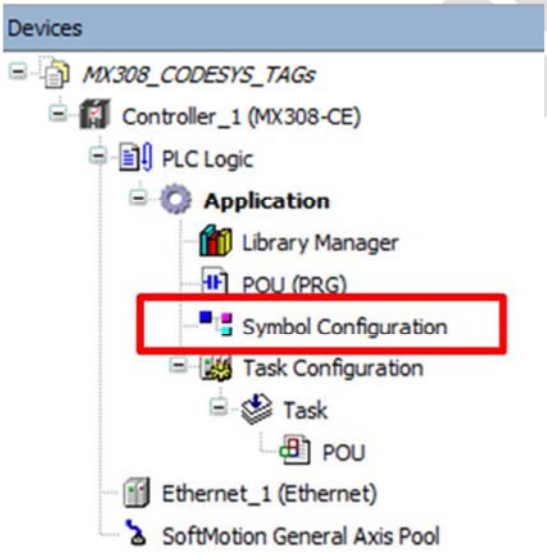


Внимание: ***В составе одного приложения (Application) может быть только одна таблица символов.***

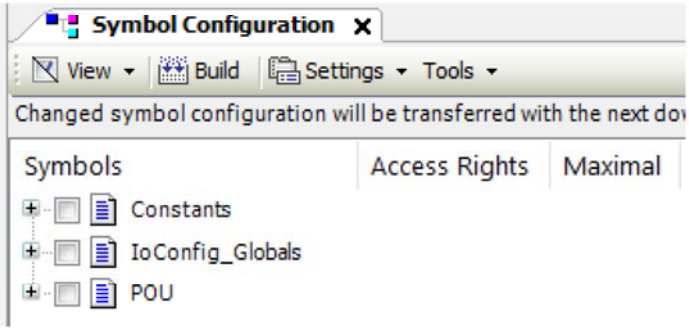
В появившемся окне определите название и нажмите **Add**:



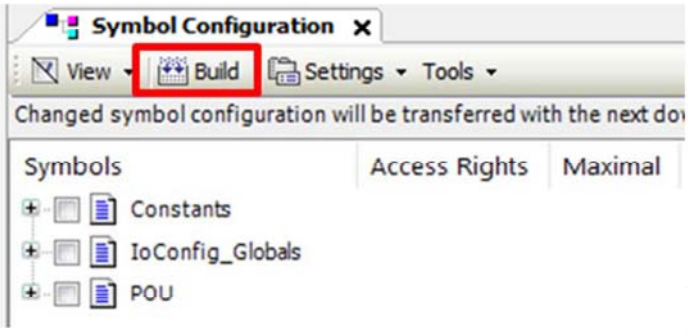
В разделе Application будет создан пункт с этим названием (в нашем примере используется название по умолчанию **Symbol Configuration**):



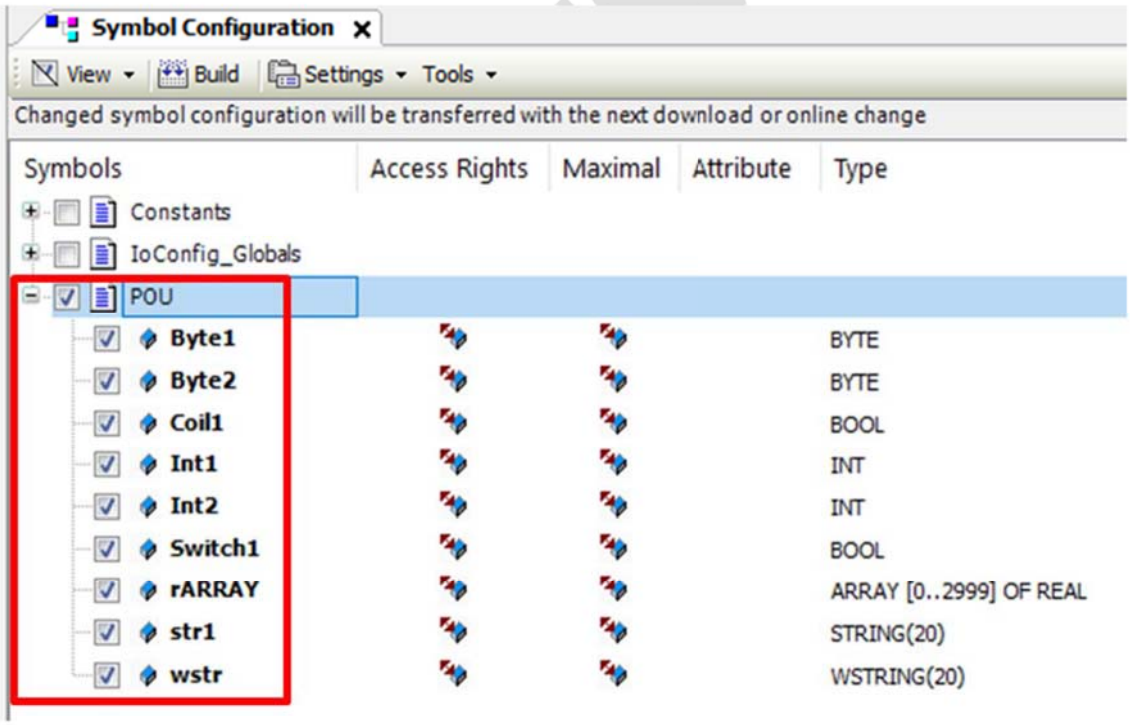
Также система сразу откроет одноименную вкладку:



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

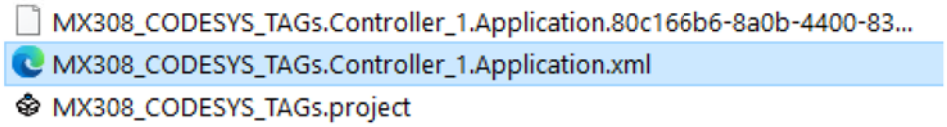


После компиляции отметьте флажками необходимы переменные:



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

MX308_CODESYS_TAGs.Controller_1.Application.xml



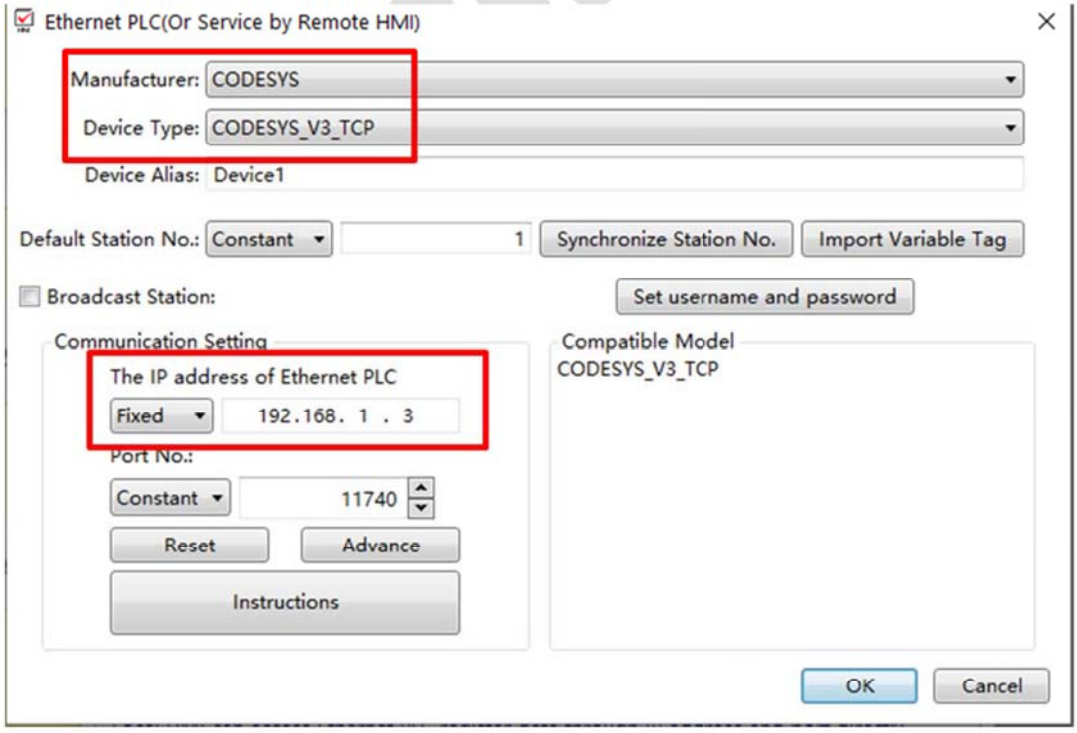
В созданном XML файле будут содержаться выбранные переменные:

```
  <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>
```

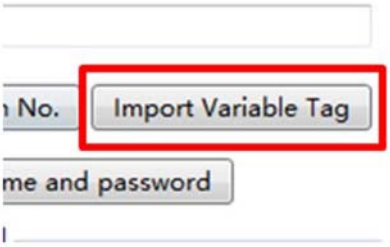
Загрузите проект в контроллер, чтобы таблица с тегами стала доступна для Мастера (Клиента)!

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):

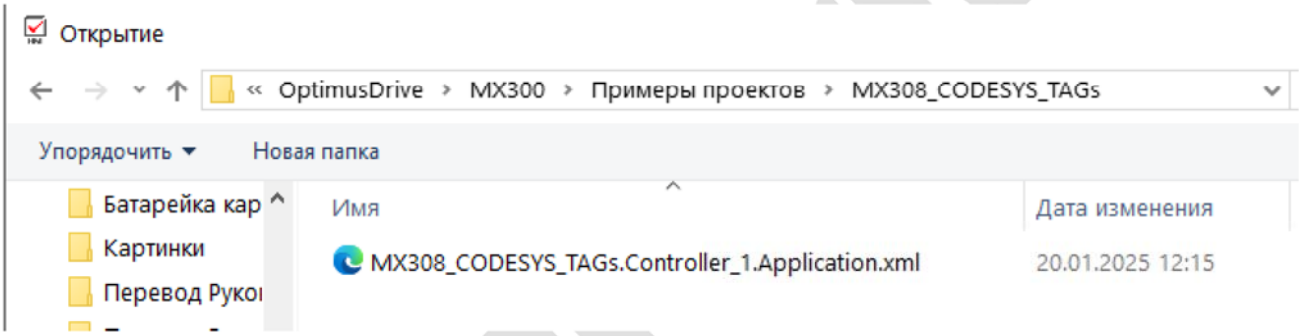


Нажмите кнопку **Import Variable Tag** и импортируйте XML файл с тегами:

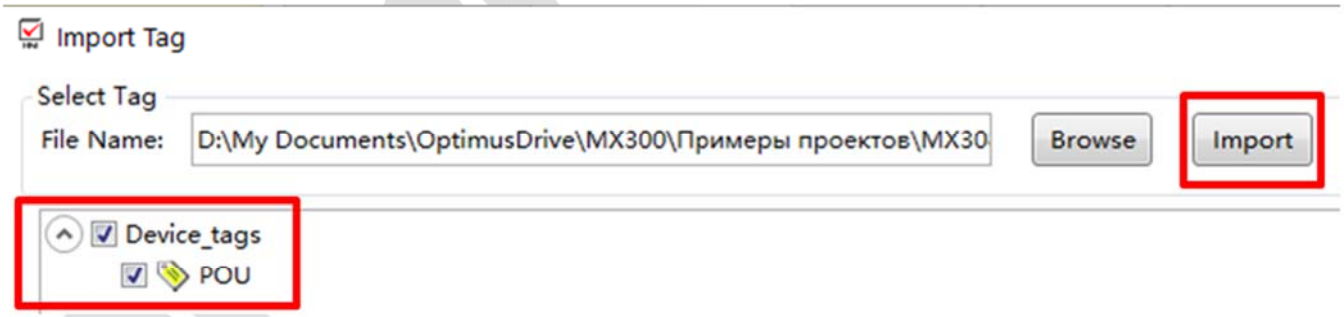


В открывшемся окне нажмите кнопку **Browse**:

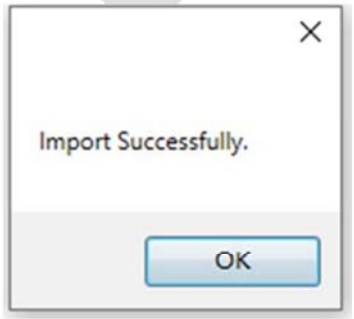
выберите XML файл:



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

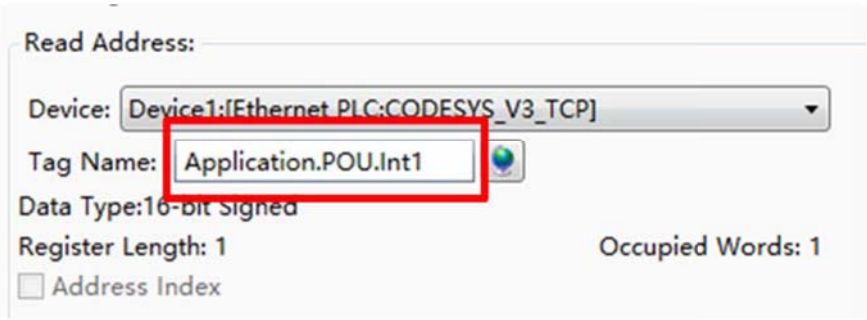


При удачном импорте тегов появится окошко с надписью **Import Successfully**:

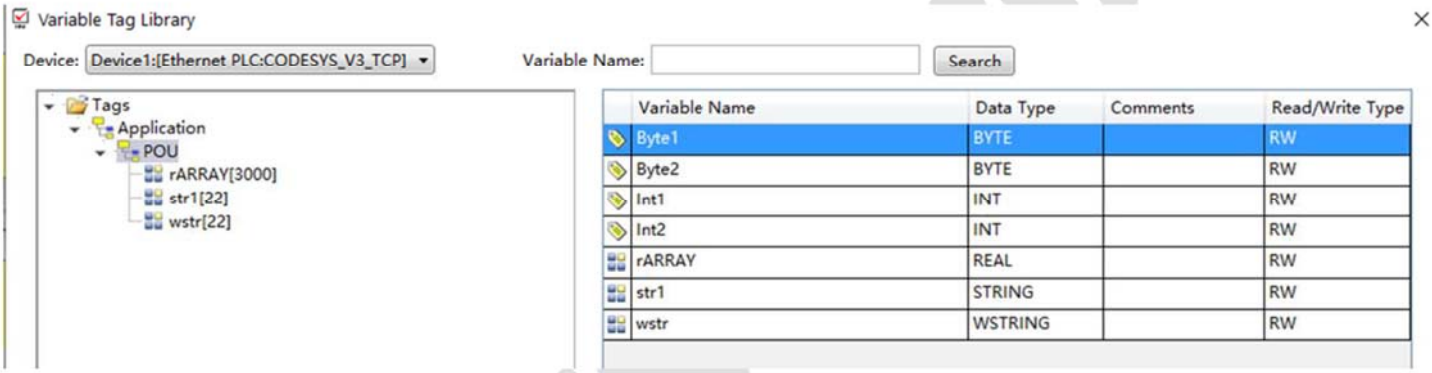


Нажмите **ОК**.

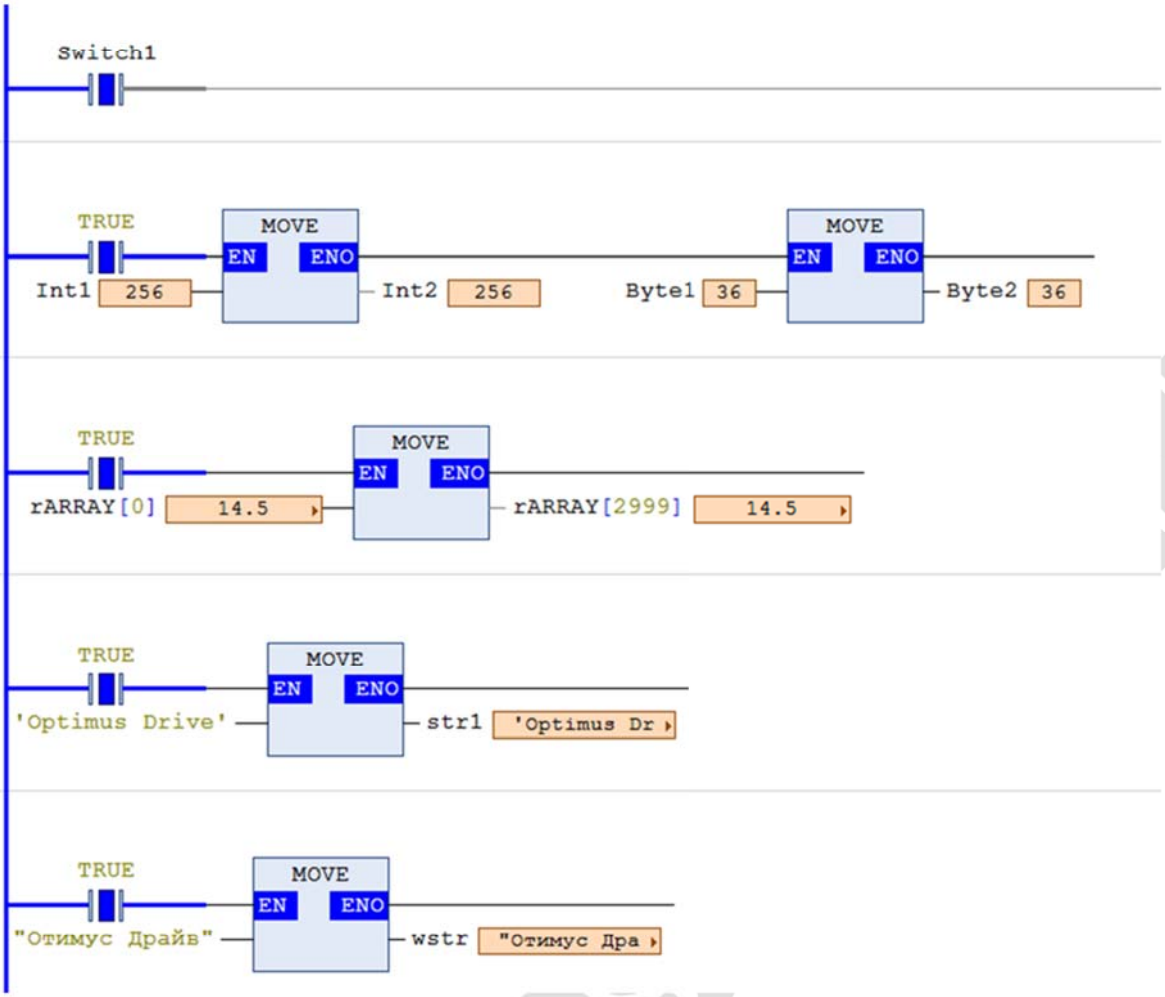
Теперь теги будут доступны как адреса у экранных объектах:



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



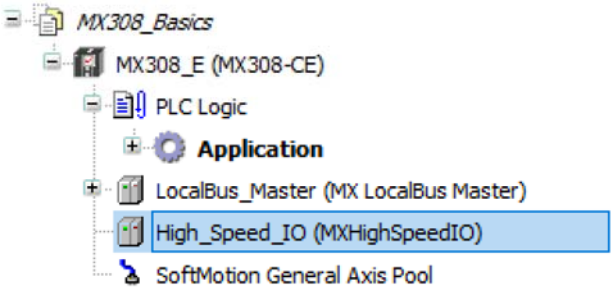
Загрузите в панель или запустите онлайн симулятор. На экране будут отображаться те же данные, что и в программе контроллера.



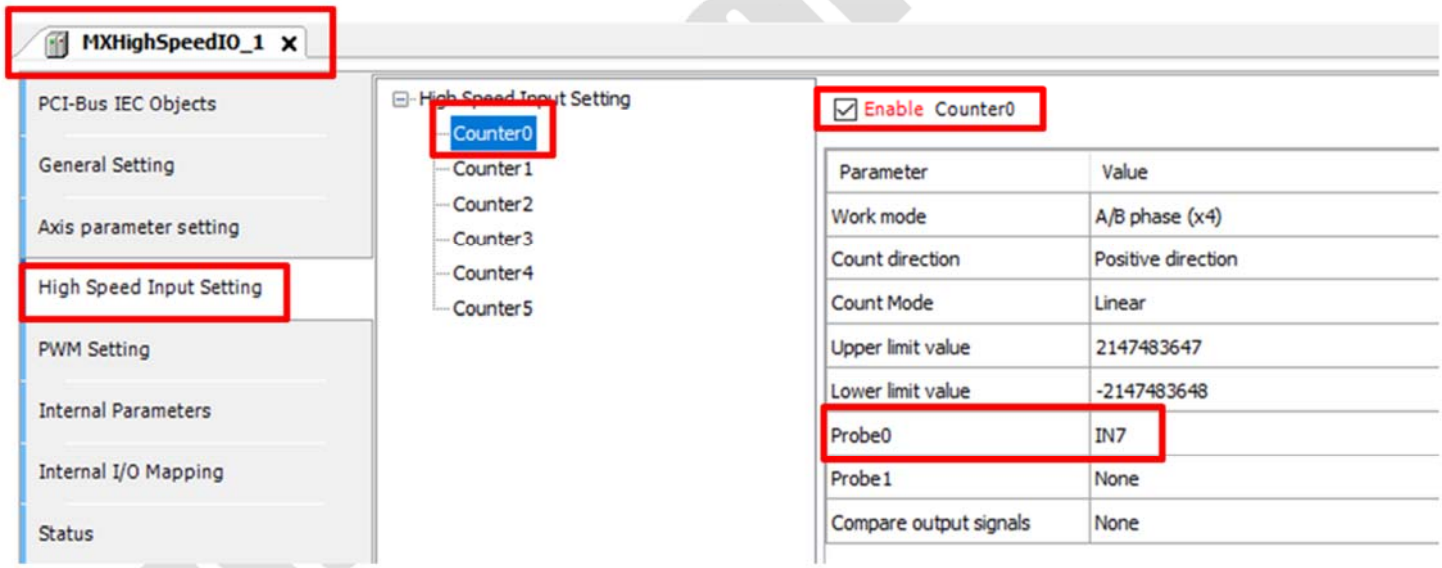
Функция Touch Probe

Контроллеры серии MX300 поддерживают функцию захвата значения высокоскоростного счётчика по переднему фронту сигнала на входе контроллера. В мировой практике данная функция называется **Touch Probe**. Поддерживается для счётчиков **Counter0** и **Counter1**.

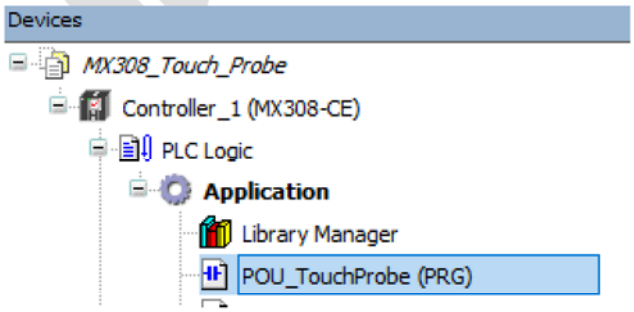
Для реализации функции захвата в древо проекта контроллера должен быть добавлен адаптер входов-выходов на ЦПУ – **MXHighSpeedIO** (см. соответствующую Главу данного Руководства). Это необходимо для организации высокоскоростного счётчика, текущее значение которого и будет захватываться.



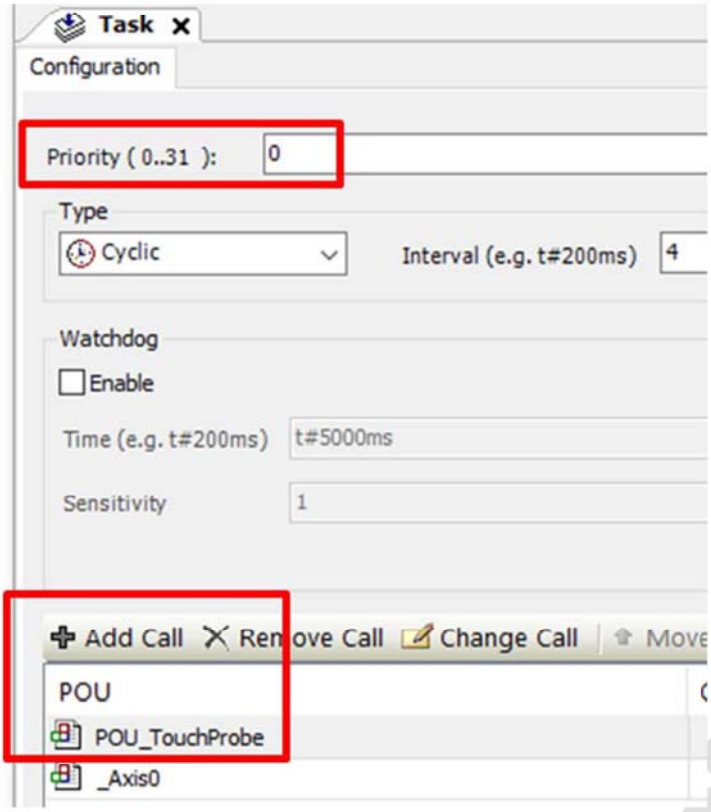
Создайте счётчик **Counter0** (см. Работа с высокоскоростными счётчиками, стр. 132 настоящего Руководства). Откройте вкладку **MXHighSpeedIO** – **High Speed Input Setting – Counter0**. Поставьте флажок **Enable** и в пункте **Probe0** выберите нужный вход на контроллере. В картинке ниже выбран вход I07.



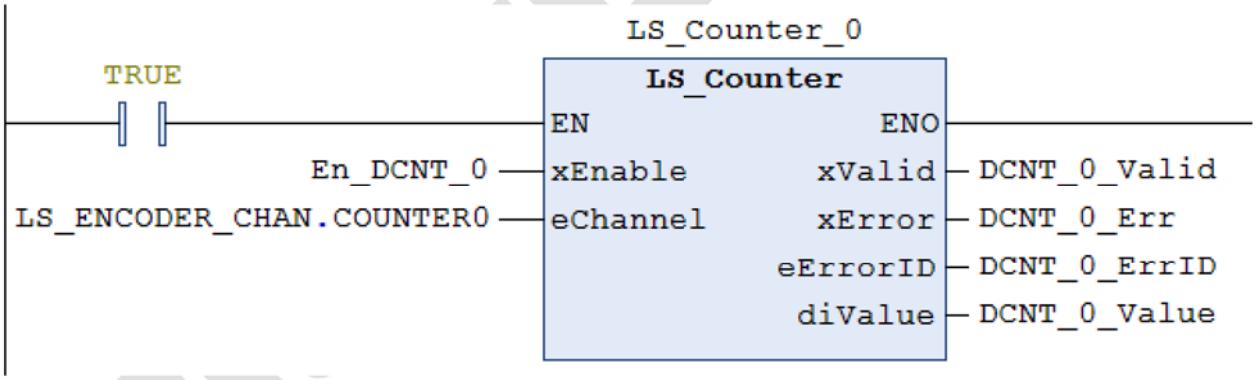
Создайте программный блок, например **POU_TouchProbe:**



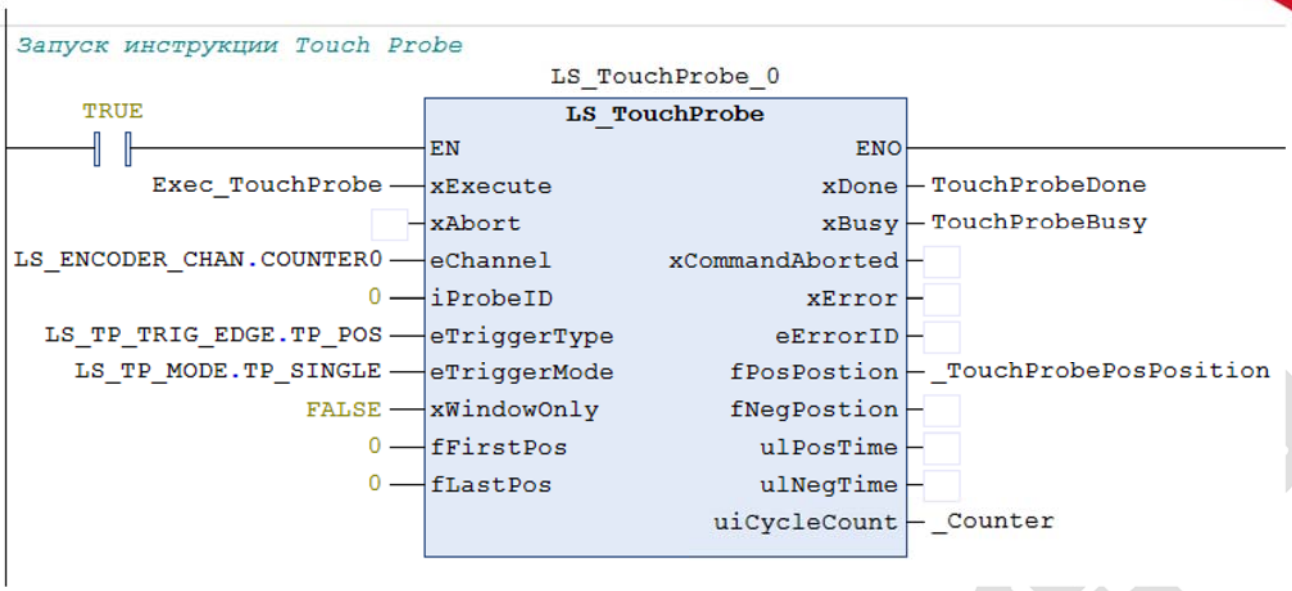
и привяжите его к основной Задаче с приоритетом 0 (обработка шины обычно).



Поставьте в данный программный блок инструкцию **LS_Counter** для запуска высокоскоростного счётчика **Counter0**:



Поставьте в этот же программный блок инструкцию **LS_TouchProbe** для запуска функции сравнения:



Описание основных ножек ФБ **LS_TouchProbe**:

xExecute – запуск ФБ в работу
eChannel – номер счётчика (0 или 1)
iProbeID – 0 – Probe0, 1 – Probe1
eTriggerType – TP_POS – захват в положительном направлении, TP_NEG – захват в отрицательном направлении, TP_TWO – захват в обоих направлениях
eTriggerMode – TP_SINGLE – разовый захват значения, TP_CONTINUE – непрерывный захват значения
xWindowOnly – FALSE – захват на всём диапазоне значений, TRUE – захват в разрешённом диапазоне
fFirstPos – начало разрешённого диапазона значений для захвата
fLastPos – конец разрешённого диапазона значений для захвата

xDone – работа ФБ завершена, захват выполнен
xBusy – ФБ стоит в готовности к захвату
xError – флаг ошибки
xErrorID – код ошибки
fPosPosition – захваченная позиция в положительном направлении
fNegPosition – захваченная позиция в отрицательном направлении
uiCycleCount – количество выполненных захватов

Для запуска ФБ необходимо взвести ножку **xExecute**. ФБ выставит **xBusy**. После захвата текущего значения счётчика ФБ сбросит **xBusy** и выставит флаг **xDone**. В полях **fPosPosition** или **fNegPosition** будет захваченное значение, которое будет сохраняться в данных полях до снятия сигнала **xExecute**. После снятия **xExecute** поля автоматически обнуляются.