Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования
ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)

Д.Е. Болбуков
А.Е. Карелин
В.П. Коцубинский

ПРИМЕНЕНИЕ ПРОТОКОЛА ETHERCAT В ЗАДАЧАХ УПРАВЛЕНИЯ ДВИЖЕНИЕМ

методические указания по выполнению практических работ
и организации самостоятельной работы

Томск 2026

УДК 681.51, 681.58
ББК 32.965
Б79

Рецензент:
Рождественский Д.А., директор ООО «Автоматизация Производств»

Болбуков, Данил Евгеньевич
Б79 Применение протокола EtherCAT в задачах управления движением: методические указания по выполнению практических работ и организации самостоятельной работы / Д.Е. Болбуков, А.Е. Карелин, В.П. Коцубинский − Томск: Томск. гос. ун-т систем упр. и радиоэлектроники, 2026. – 60 с.

В методических указаниях изложены практические аспекты использования протокола EtherCAT для управления электроприводами: от принципа работы драйверов шаговых двигателей и устройств протокола EtherCAT до создания проекта управления движением на ПЛК в среде CODESYS на основе библиотеки SoftMoution, что позволяет самостоятельно разработать прототип системы управления движением на основе ПЛК.

Методические указания предназначены для студентов технических вузов, инженерно-технических специалистов.

Авторами данных методических указаний являются профессорско-преподавательский состав кафедры КСУП (Д.Е. Болбуков, А.Е. Карелин, В.П. Коцубинский), а также сотрудник профильного предприятия, директор ООО«НПП «Оптимум», г. Томск, А.Е. Карелин.

Одобрено на заседании кафедры КСУП, протокол № 8 от 12.05.2026 г.

УДК 681.51, 681.58
ББК 32.965

Оглавление

Введение

Сегодня в области промышленной автоматизации всё чаще применяют разнообразные электроприводы позиционирующие и перемещающие различные элементы технологического оборудования, так же широко применяются системы с программным управлением движением. Ключевую роль в применении электроприводов играет синхронное управление множеством приводов. Для решения этой задачи применяют специализированные драйверы двигателей с протоколом EtherCAT (Ethernet for Control and Automation Technology – Ethernet для контроля и автоматизации управления), позволяющие реализовывать синхронное управление множеством распределённых в пространстве приводов [1-3]. Однако данные драйверы бесполезны без системы управления движением выполняющей расчеты траектории движения и генерирующей команды для управления отдельными приводами, в качестве такой системы в промышленности часто применяют ПЛК. В данных методических указаниях рассматривается построение такой системы на основе ПЛК XINJE XSDH-60A32 в среде программирования CODESYS на основе библиотеки управления движением SoftMoution [4-7].

Целью методических указаний является получение практических навыков работы с протоколом EtherCAT и навыков управления движением в среде CODESYS: от первоначальной настройки и конфигурирования драйверов двигателей до построения графического интерфейса, программирования логики управления перемещением.

В методических указаниях последовательно рассматриваются следующие вопросы:

- создание и запуск проекта CODESYS;
- конфигурирование драйвера шагового двигателя XINJE DP3CL-705;
- работа с протоколом EtherCAT для управления движением в среде CODESYS;
- работа с библиотекой SoftMoution;
- настройка обработки отдельных регистров EtherCAT устройства, их запись и чтение, поиск нулевого положения оси SoftMoution;
- построение простого пользовательского интерфейса на основе визуализации CODESYS.

Все разделы сопровождаются подробными пошаговыми инструкциями и скриншотами, что позволяет обучающемуся выполнять практические работы в режиме «следуй за инструкцией». В результате освоения материала студент приобретает базовые компетенции, необходимые для самостоятельной работы с протоколом EtherCAT и программирования систем управления движением на основе ПЛК.

Методические указания предназначены для аудиторной работы под руководством преподавателя. Предполагается, что перед началом работы на компьютере установлена среда разработки CODESYS V3.5 SP17 Patch 3, а также у обучающегося имеются базовые навыки работы с операционной системой Windows.

1. РАЗРАБОТКА ПРОЕКТА УПРАВЛЕНИЯ ДВИЖЕНИЕМ В СРЕДЕ CODESYS

1.1.Создание проекта

Для того, чтобы создать проект необходимо запустить среду CODESYS. Для этого заходим в Пуск -> CODESYS -> CODESYS V3.5 SP17 Patch 3. Либо запускаем её по иконке на рабочем столе (рис. 1-2).

Рисунок 1. Запуск CODESYS

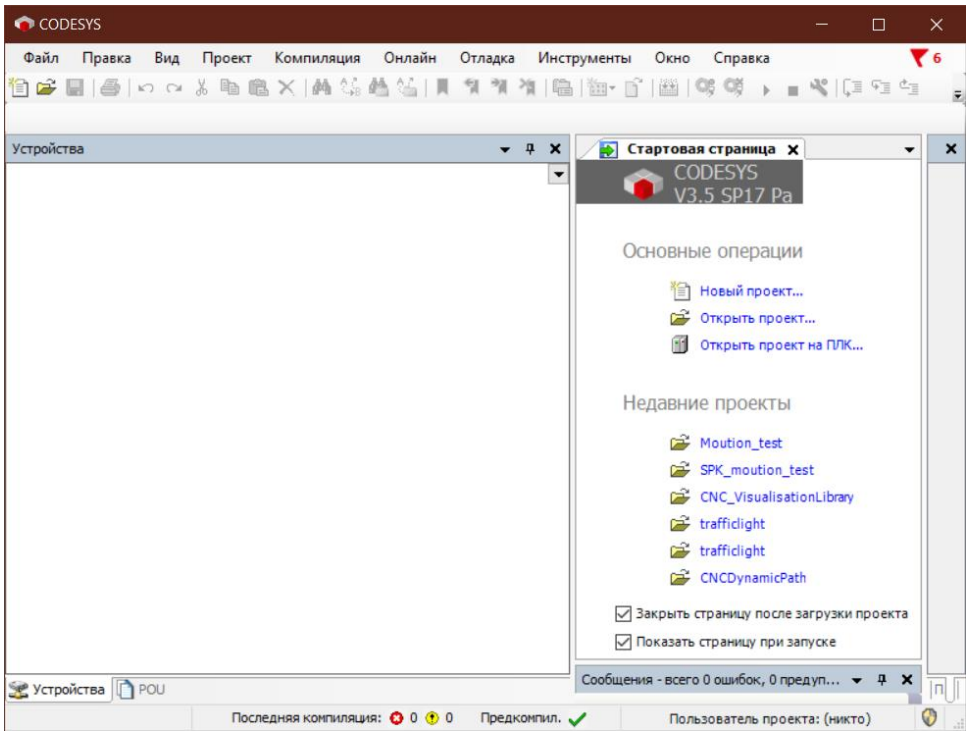


Рисунок 2. Среда программирования CODESYS

Создайте новый проект. Для этого нажмите на изображение чистого листа в панели инструментов или сочетание клавиш Ctrl + N.

Выберите Стандартный проект выберите расположение проекта и введите его название затем нажмите «ОК» (рис. 3).

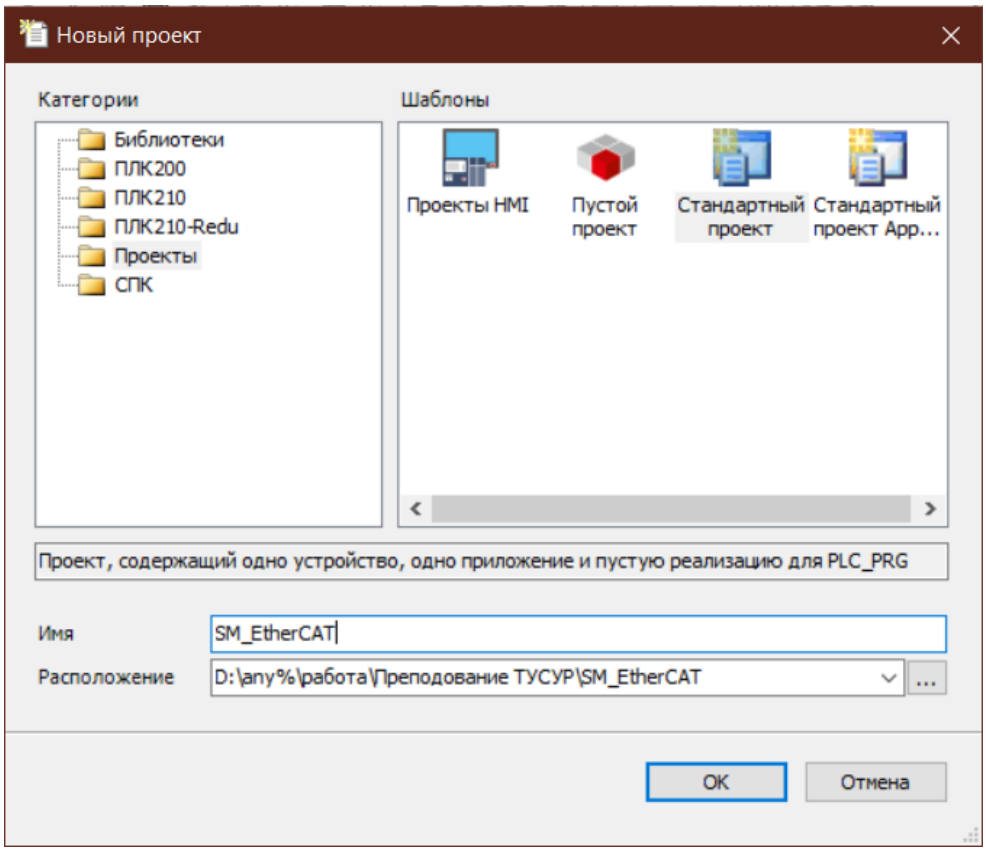


Рисунок 3. Создание нового проекта

В следующем окне выберите устройство XSDH-60A32 и язык PLC_PRG CFC. Нажмите «ОК» (рис. 4).

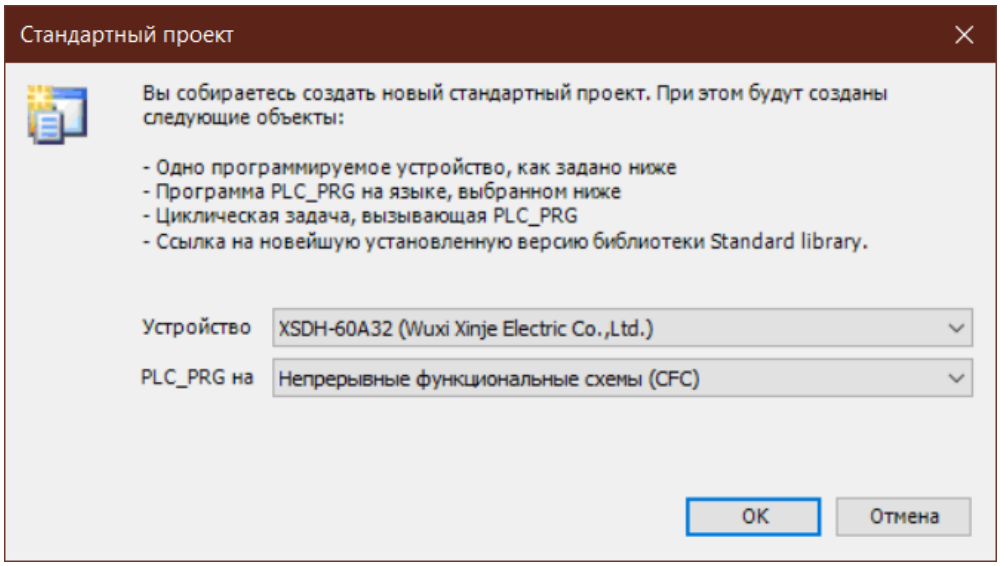


Рисунок 4. Выбор устройства проекта

После создания проекта откроется главное окно CODESYS, дважды кликните по PLC_PRG. В левой части окна расположено дерево проекта, здесь отображаются все файлы проекта, задачи, настроенные проекте устройства. В верхней части окна расположена панель инструментов, с неё открываются настройки проекта, запускается компиляция проекта, выполняется загрузка программы в ПЛК. В правой части окна расположены редактор файлов, разделённый на области переменных сверху и область программы снизу, в нижнем правом углу находится консоль вывода (рис. 5).

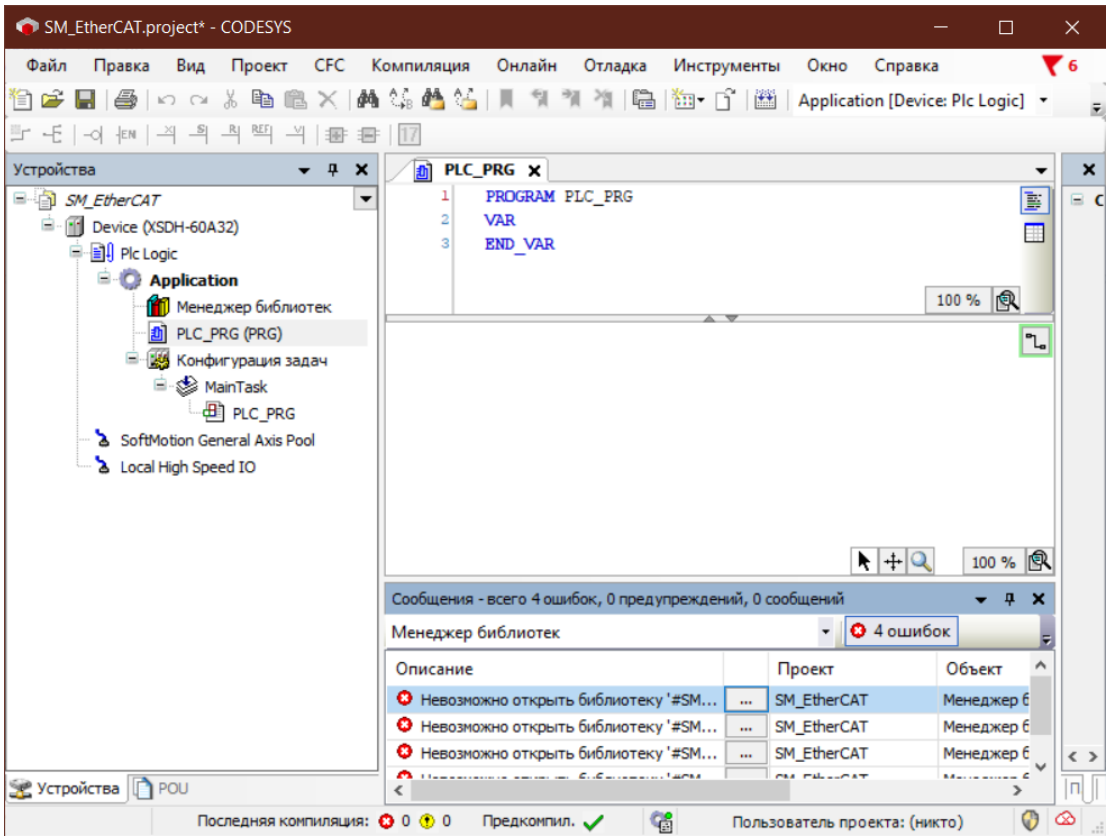


Рисунок 5. Проект CODESYS

Если в консоли отображаются ошибки, связанные с отсутствием библиотек, нажмите на три точки во втором столбце консоли и выберите «загрузить библиотеки», откроется следующее окно (рис. 6):

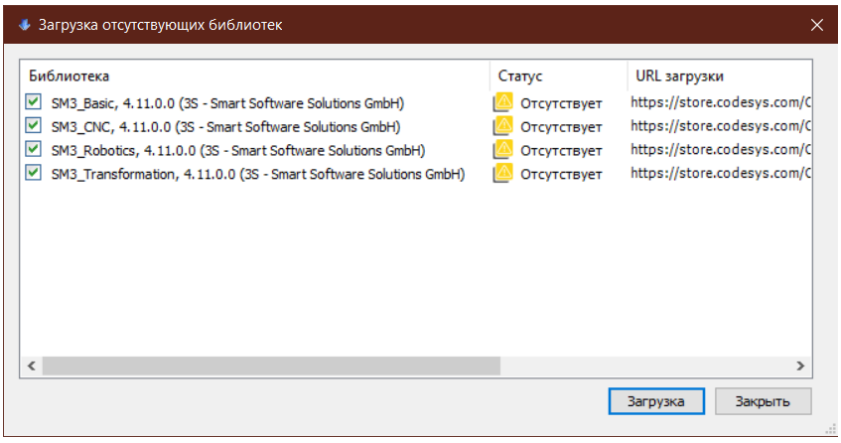


Рисунок 6. Загрузка библиотек

Нажмите «Загрузка» и дождитесь завершения загрузки, после чего закройте это окно, ошибки в проекте должны пропасть.

Следующим шагом выполним соединение с ПЛК, для этого подключите ПЛК к компьютеру при помощи интернет кабеля (на ПЛК используйте гнездо с подписью Enet, данное гнездо расположено выше гнезда Ecat), включите питание ПЛК, затем дважды кликните по объекту ПЛК «Device (XSDH-60A32)» в дереве проекта, откроется окно конфигурации ПЛК, на первой вкладке «Установка соединения» кликните «Сканировать сеть» в открывшемся окне кликните «Сканировать сеть» (рис. 7-8).

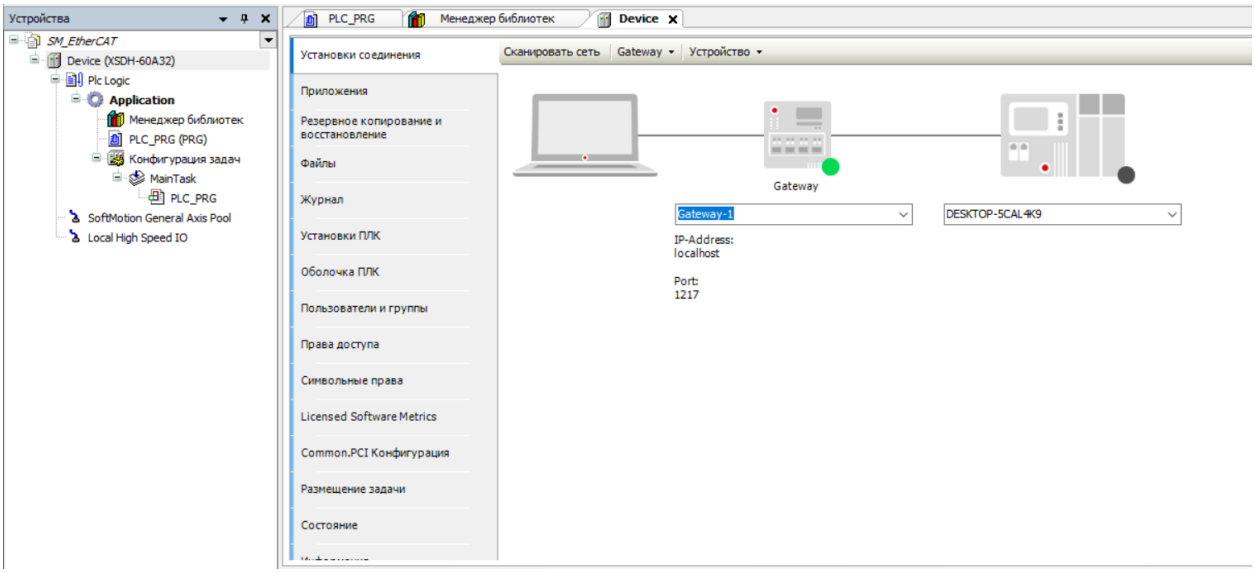


Рисунок 7. Конфигурация ПЛК

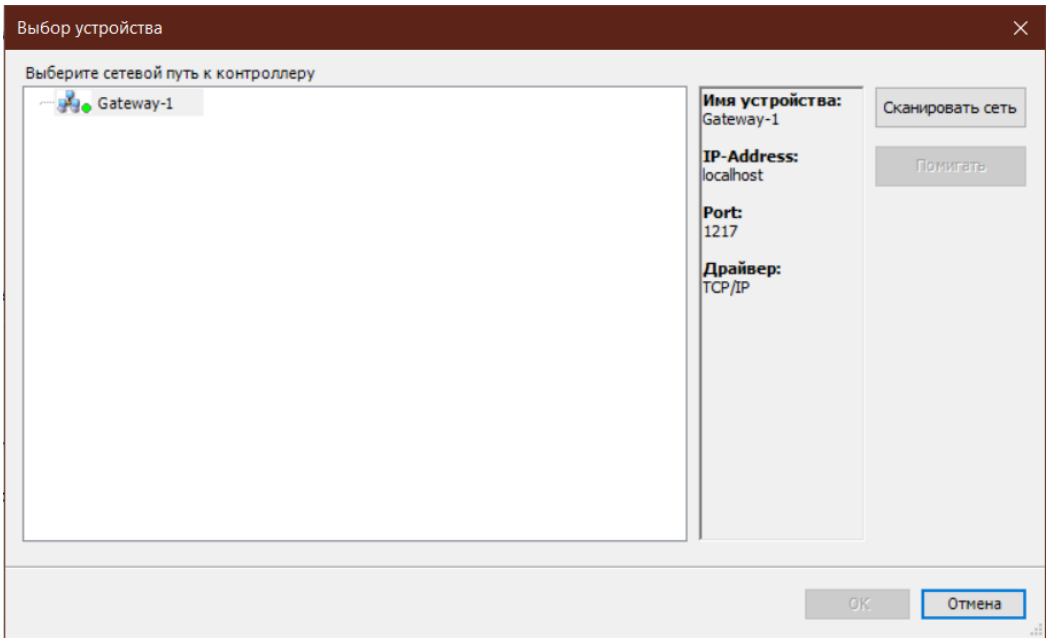


Рисунок 8. Окно сканирования сети

После запуска сканирования в окне должен отобразится ПЛК, как показано на рисунке 9, если этого не произошло перейдите к инструкции по настройке сетевой карты компьютера в конце данного подраздела.

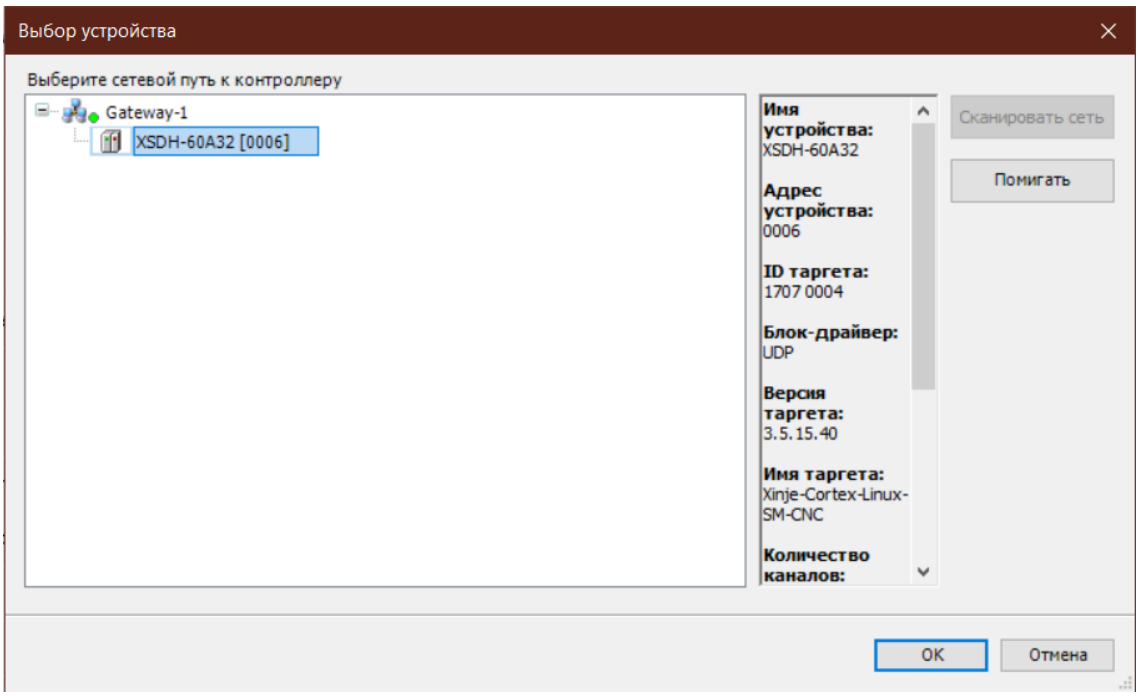


Рисунок 9. Обнаружение ПЛК

Выберите обнаруженный ПЛК и нажмите «ОК», после чего перейдите в главное окно CODESYS и на верхней панели выберите пункт «Логин» (иконка шестеренки с зелёным кружком на панели или онлайн -> Логин), после чего должно произойти соединение с ПЛК, в диалоговом окне подтвердите загрузку программы. На данном этапе мы создали и сконфигурировали проект CODESYS для дальнейшей работы [8-9].

Настройка сетевой карты (в случае если не удается обнаружить ПЛК).

Перейдите в Пуск –> Параметры –> Сеть и Интернет –> Настройка параметров адаптера –> Ethernet, откроется окно состояния подключения по Ethernet в нем нажмите «Свойства» (рис. 10-11).

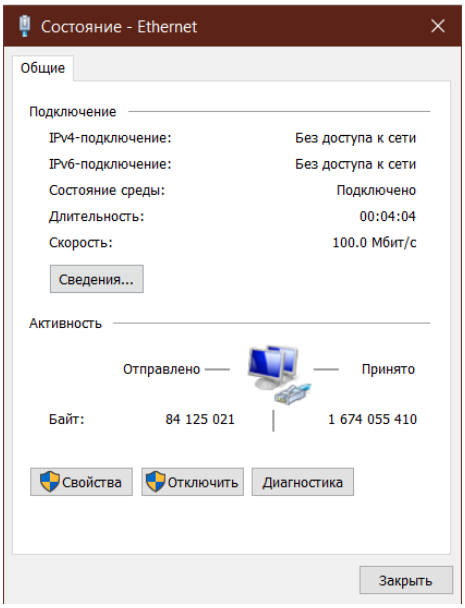


Рисунок 10. Состояние Ethernet

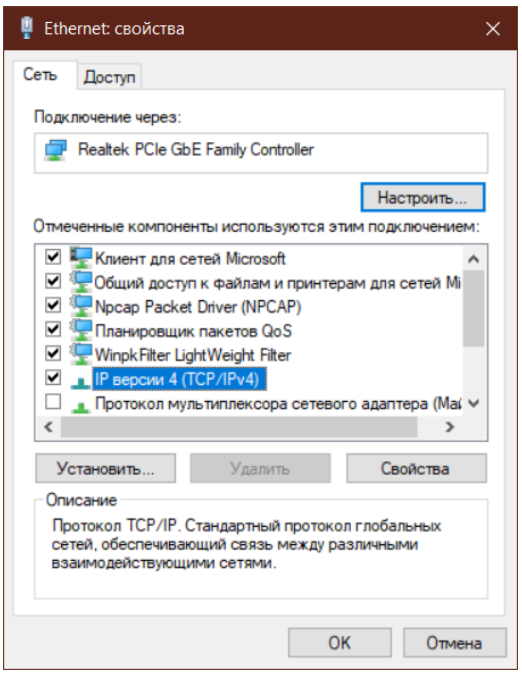


Рисунок 11. Свойства Ethernet

В окне свойств выберите «IP версии 4 (TCP/IPv4)» и нажмите «Свойства». В открывшемся коне установите параметры подключения как на следующем рисунке и нажмите «ОК». После чего повторите сканирование сети в CODESYS и продолжите выполнять работу (рис. 12).

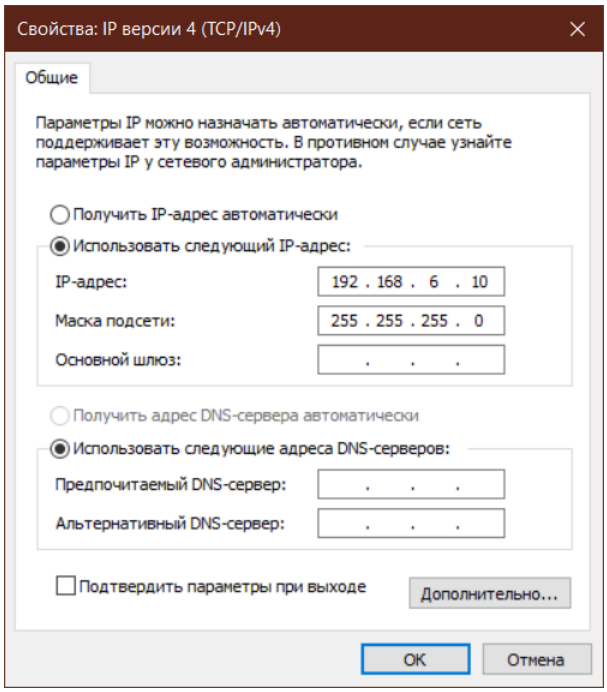


Рисунок 12. Настройки сетевой карты компьютера для подключения к ПЛК

Пояснение: IP адрес компьютера должен находится в той же подсети что и контроллер для того, чтобы CODESYS смог его обнаружить, так как по умолчанию IP ПЛК XINJE XSDH-60A32 статический и установлен как 192.168.6.6, сконфигурируем IP нашего компьютера на любой свободный адрес в подсети, например 192.168.6.10.

1.2.Конфигурирование драйвера шагового двигателя

Для дальнейшей работы необходимо сконфигурировать драйвер шагового двигателя, чтобы предотвратить перегрев и повреждение шагового двигателя, а также для установки параметров работы, которые потребуются при его добавлении в проект CODESYS.

Перед настройкой драйвера рассмотрим принцип работы шагового двигателя. Согласно принятой в России классификации электрических машин шаговый двигатель представляет собой синхронную бесщеточную явно полюсную электрическую машину с постоянными магнитами. Из этого определения следует что в нём нет скользящих контактов (щеток) поскольку на роторе двигателя отсутствуют обмотки, наличие магнитного поля ротора в таком двигателе обеспечивается постоянными магнитами, вращение ротора такого двигателя совпадает с вращением магнитного поля статора и является дискретным, то есть делиться на равные интервалы между полюсами машины.

Рассмотрим устройство типичного ШД, двигатель состоит из ротора с постоянными магнитами и статора на этих частях расположены продольные канавки и выступы необходимые для создания явных полюсов двигателя, как правило сегодня выпускают шаговые двигатели имеющие 200 или 400 шагов на оборот и угол между полюсами 1,8 или 0,9 градуса соответственно. На статоре расположены обмотки двигателя, создающие вращающееся магнитное поле. Ротор двигателя установлен в корпусе (как правило совмещен со статором) на подшипниках один из которых жестко закреплён в корпусе, а второй имеет осевой ход и подпружинен для компенсации изменения длинны вала при нагреве двигателя. Устройство представлено на рисунке 13.

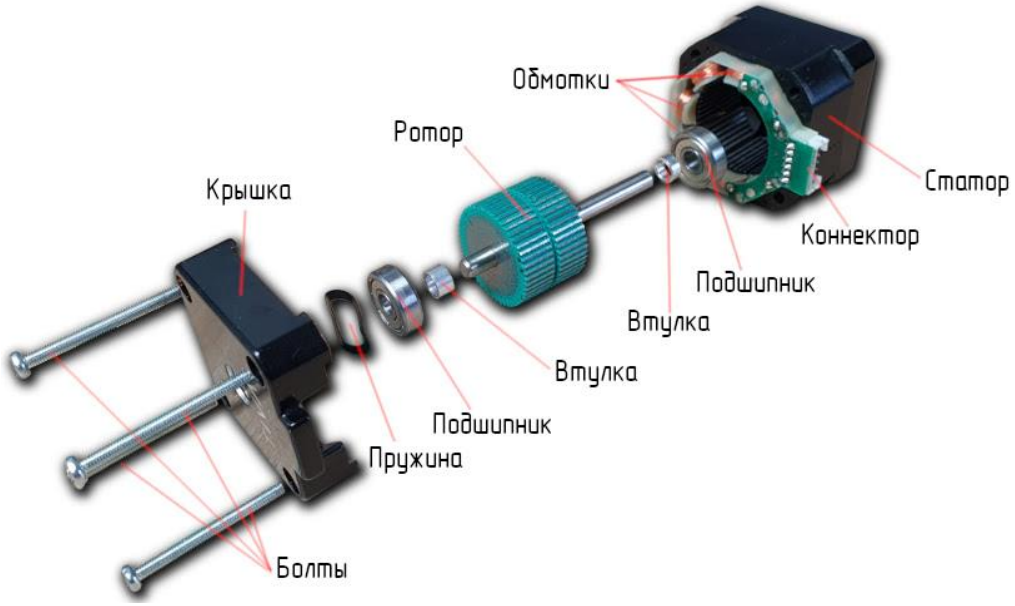


Рисунок 13. Устройство шагового двигателя

Современные шаговые двигатели имеют как правило две обмотки которые могут быть разделены на несколько катушек обмотки размещены вокруг ротора со смещением 90 градусов относительно друг друга. Электрически шаговые двигатели разделяют на биполярные и униполярные, их отличие заключается в способе подключения обмоток. В

униполярном двигателе выполнен отвод от средней точки обмотки и половины обмотки представлены двумя катушками, расположенными напротив друг друга вокруг ротора, благодаря этому для управления двигателем можно использовать всего четыре ключа как это показано на схеме, изменение направления магнитного потока катушки происходит попеременным включением половинок этой катушки. В биполярном же двигателе отводов от середины катушки нет, и соответственно для изменения направления магнитного потока необходимо изменять направление тока в катушки, для этого применяют мостовую схему, представленную на рисунке 14, открывая ключи попарно по диагонали моста Q1, Q4 и Q2, Q3, что позволяет управлять направлением тока в обмотке. Биполярные двигатели проще в изготовлении и как правило дешевле, однако требуют более сложного драйвера с большим числом ключей.

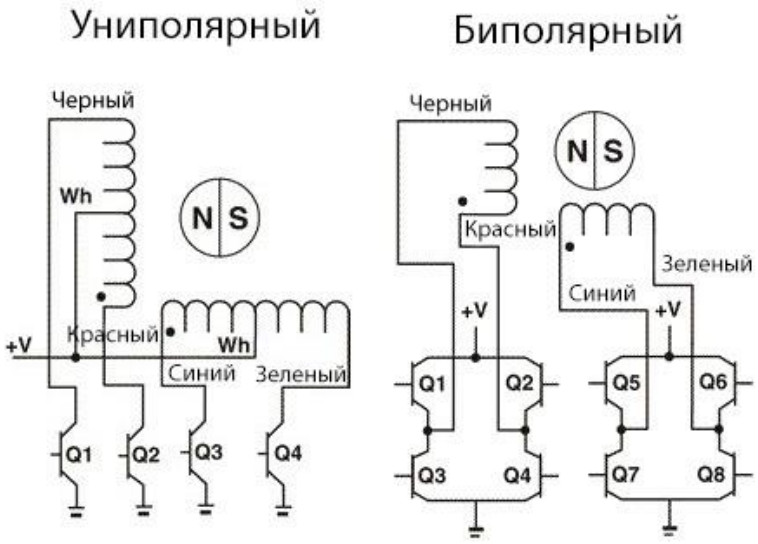


Рисунок 14. Устройство драйверов шагового двигателя

Мы рассмотрели, как устроен шаговый двигатель и как формируется ток в его катушках. Теперь рассмотрим каким образом необходимо управлять таком в катушках чтобы получить вращение двигателя.

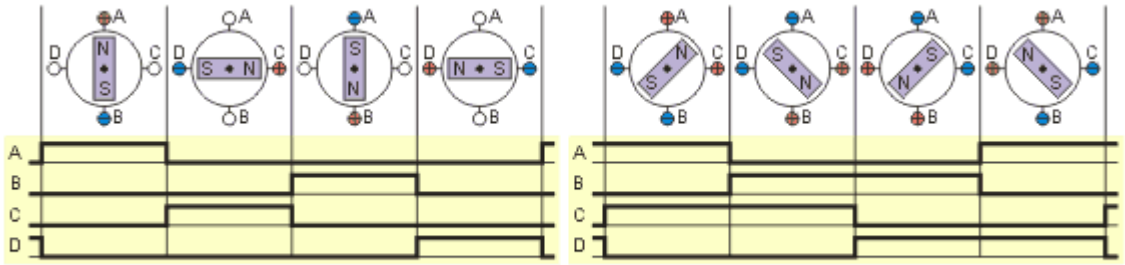
Первый способ обеспечивается попеременной коммутации фаз, при этом они не перекрываются, в один момент времени включена только одна фаза (а). Этот способ называют «one phase on» full step или «wave drive mode». Точки равновесия ротора для каждого шага совпадают с естественными точками равновесия ротора у не запитанного двигателя. Недостатком этого способа управления является то, что для биполярного двигателя в один и тот же момент времени используется 50% обмоток, а для униполярного - только 25%. Это означает, что в таком режиме не может быть получен полный момент.

Второй способ - управление фазами с перекрытием: две фазы включены в одно и то же время. Его называют «two-phase-on» full step или просто «full step mode». При этом способе управления ротор фиксируется в промежуточных позициях между полюсами статора (б) и обеспечивается примерно на 40% больший момент, чем в случае одной

включенной фазы. Этот способ управления обеспечивает такой же угол шага, как и первый способ, но положение точек равновесия ротора смещено на полшага.

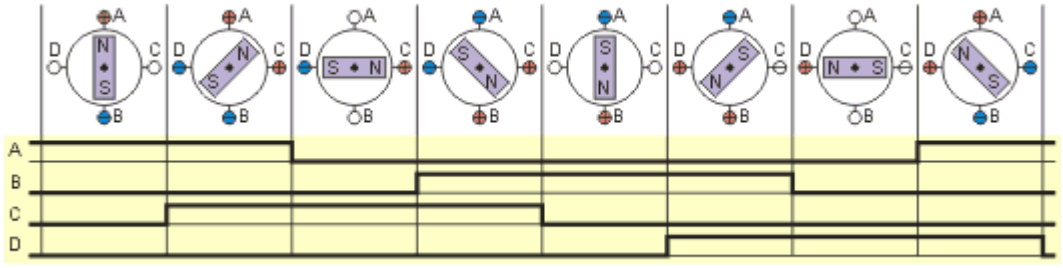
Третий способ является комбинацией первых двух и называется полушаговым режимом, «one and two-phase-on» half step или просто «half step mode», когда двигатель делает шаг в половину основного. Этот метод управления достаточно распространен, так как двигатель с меньшим шагом стоит как правило дороже и поэтому получить от шагового двигателя с 200 шагами на оборот 400 шагов на оборот может быть удобно. Каждый второй шаг запитана лишь одна фаза, а в остальных случаях запитаны две (в). В результате угловое перемещение ротора составляет половину угла шага для первых двух способов управления. Кроме уменьшения размера шага этот способ управления позволяет частично избавиться от явления резонанса. Полушаговый режим обычно не позволяет получить полный момент, хотя наиболее совершенные драйверы реализуют модифицированный полушаговый режим, в котором двигатель обеспечивает практически полный момент, при этом рассеиваемая мощность не превышает номинальной.

Еще один способ управления называется микрошаговым режимом или «micro stepping mode». При этом способе управления ток в фазах нужно менять небольшими шагами, обеспечивая таким образом дробление половинного шага на еще меньшие микрошаги. Когда одновременно включены две фазы, но их токи не равны, то положение равновесия ротора будет лежать не в середине шага, а в другом месте, определяемом соотношением токов фаз. Меняя это соотношение, можно обеспечить некоторое количество микрошагов внутри одного шага. Вместе с тем, для реализации микрошагового режима требуются значительно более сложные драйверы, позволяющие задавать ток в обмотках с необходимой дискретностью. Полушаговый режим является частным случаем микрошагового режима, но он не требует формирования ступенчатого тока питания катушек. Диаграммы отражающие все эти методы представлены на рисунке 15.



а) полношаговый режим, включена одна фаза, "one phase on" full step (wave drive mode)

б) полношаговый режим, включены две фазы, "two-phase-on" full step (full step mode)



в) полушаговый режим, "one and two-phase-on" half step (half step mode)

Рисунок 15. Диаграммы управления обмотками шагового двигателя

Следующим шагом подключим драйвер и источнику питания, а также шаговый двигатель к драйверу по схеме из официальной документации, представленной на рисунке 16. Для подключения драйвера к ПЛК используем интернет кабель, к ПЛК он подключается в гнездо Ecat, к драйверу в гнездо ECAT IN.

ВНИМАНИЕ! Перед подключением драйвера к регулируемому блоку питания, установите на блоке питания напряжение 24 В, и ограничение тока 3 А. Отключите блок питания и только после этого подключите драйвер!

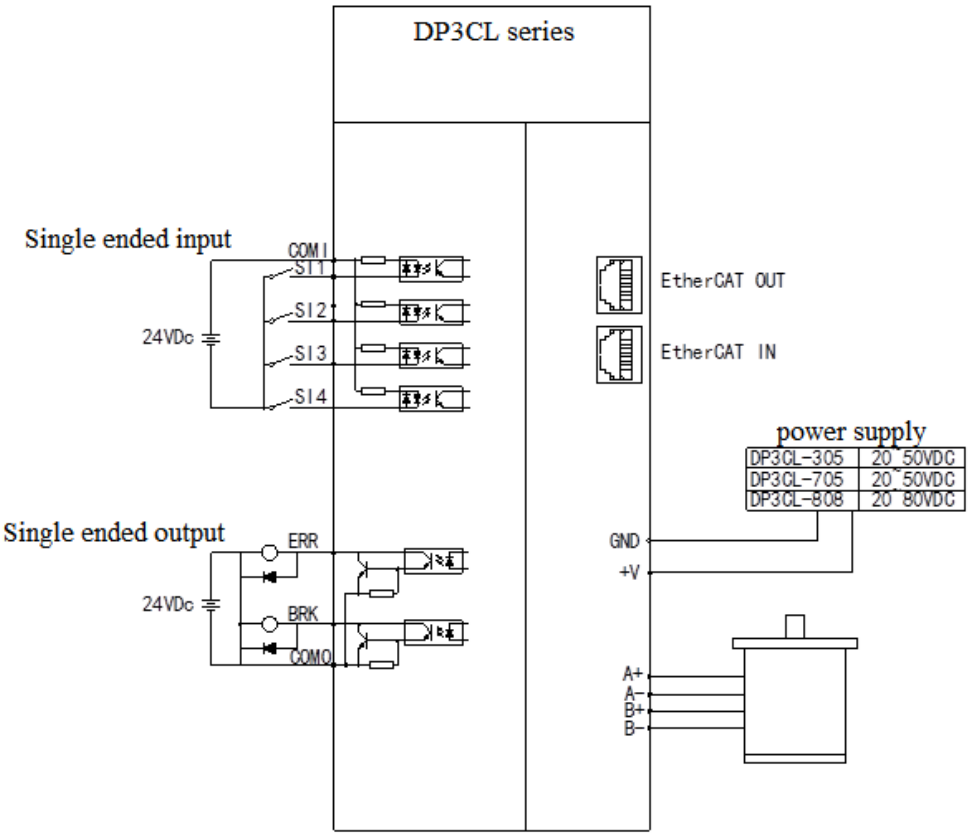


Рисунок 16. Схема подключения драйвера

Рассмотрев принцип работы и устройство шагового двигателя и собрав экспериментальную установку, перейдём непосредственно к настройке драйвера XINJE DP3CL-705 для управления шаговым двигателем Nema17-48мм [10-11].

Необходимо сконфигурировать следующие параметры.

1. Рабочий ток в обмотке двигателя – для используемого двигателя составляет 1,2 А, стоит обратить внимание на определение данного параметра из документации, поскольку в случае установки завышенного тока двигатель будет перегреваться и может выйти из строя, а в случае установки недостаточного тока, двигатель не будет создавать номинальный момент и это может приводить к пропуску шагов, что недопустимо в большинстве случаев. Так же важно разделять рабочий ток (среднеквадратическое значение тока, RMS, Root Mean Square) и пиковый ток (амплитудное значение тока) в обмотке двигателя. Как правило на драйвере шагового двигателя устанавливается именно рабочий ток.

2. Ток удержания – этот параметр выражается в процентах от рабочего тока и определяет ток, который подается в обмотки двигателя в режиме удержания (когда обмотки запитаны, но вал двигателя не вращается). Установка параметра зависит от применения двигателя, например для приводов, не испытывающих нагрузки в статическом состоянии, например, в случае если установлен редуктор, обеспечивающий самоблокировку, устанавливают 20-40%, в случаях, когда нагрузка на двигатель не пропадает после остановки, например, вертикально расположенная ось станка или подъемный механизм, устанавливают значение 70-100%. В текущей работе установим значение 30%.
3. Микрошаг – параметр, показывающий на сколько микрошагов будет разделятся физический шаг двигателя при помощи задания промежуточных уровней тока в обмотках двигателя. Данный параметр может быть представлен либо в виде дроби (1/4, 1/8, 1/16, 1/32 и т.д.) показывающей на сколько микрошагов делится один шаг, либо в виде числа шагов на оборот (800, 1600, 3200, 6400 и т.д.) исходящих из того что у двигателя 200 физических шагов на оборот. В используемом драйвере микрошаг задаётся вторым способом, установим микрошаг 1/16 или 3200 шагов на оборот.
4. Параметры парковки и назначения дискретных входов драйвера. Данные настройки мы рассмотрим в одной из следующих работ, когда будем настраивать парковку оси на основе данного драйвера шагового двигателя.

Перейдём к настройке, для этого откройте утилиту StepDriver, для этого перейдите Псук -> StepDriver, воспользуйтесь поиском в меню пуск или ярлыком на рабочем столе. Откроется окно, представленное на рисунках 17-18.

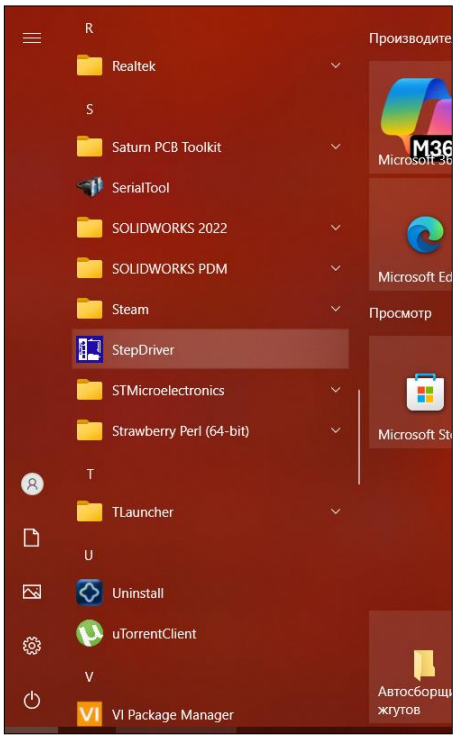


Рисунок 17. Запуск утилиты StepDriver

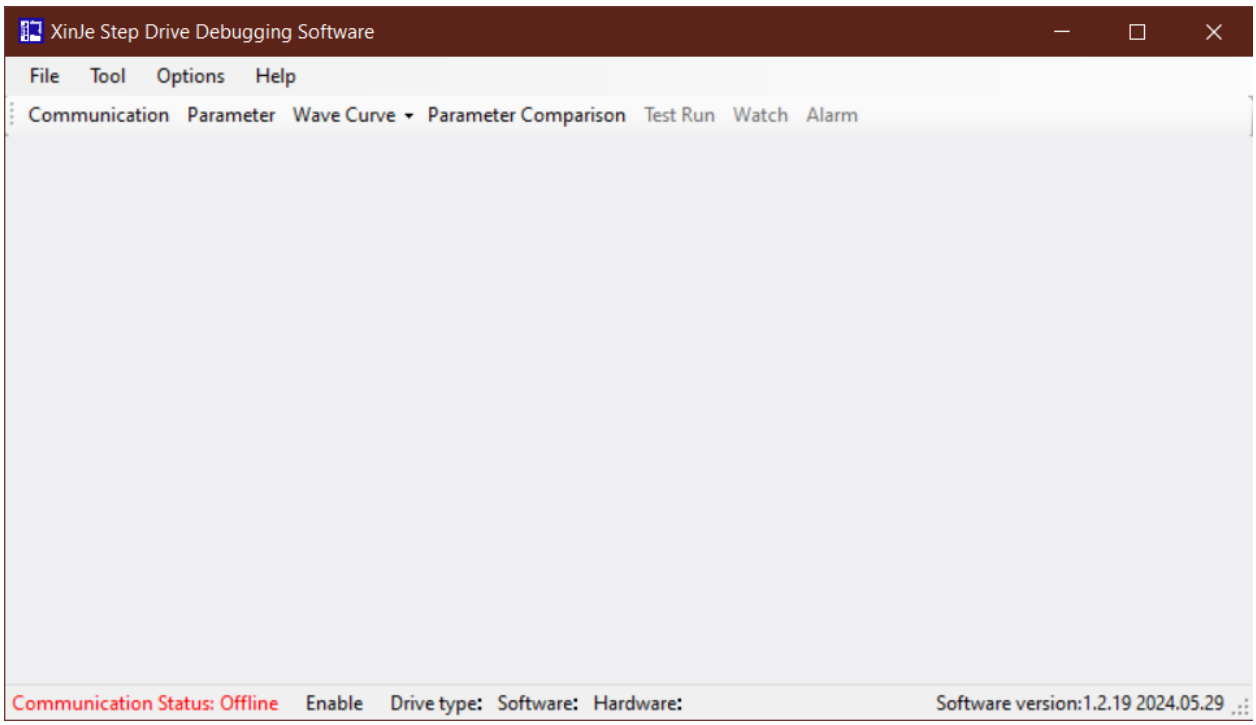


Рисунок 18. Главное окно StepDriver

В главном окне сверху расположена панель инструментов, снизу статус подключенного драйвера.

Драйвер конфигурируется через интерфейс RS-232, несмотря на то что на корпусе драйвера разъем MiniUSB. Данный интерфейс не поддерживает горячее подключение и отключение устройств, поэтому строго соблюдайте порядок работы, описанный далее.

Для подключения к драйверу необходимо сделать следующие действия именно в таком порядке, при отключении порядок действий обратный **(ВНИМАНИЕ! Несоблюдение порядка действий при подключении и отключении драйвера может привести к его повреждению!)**:

1. убедится в отсутствии питания на драйвере;
2. подключить переходник MiniUSB->COM к преобразователю COM->USB;
3. подключить разъем MiniUSB переходника к драйверу;
4. подключить разъем USB преобразователя к компьютеру;
5. включить питание драйера.

После подключения драйвера к компьютеру, необходимо нажать «Communication» на панели инструментов. Откроется окно соединения с драйвером (рисунок 19). В данном окне необходимо нажать кнопку «Auto search», подтвердить все появляющиеся окна, после чего утилита должна установить соединение с драйвером, как показано на рисунке 20. После успешного соединения нажмите «ОК».

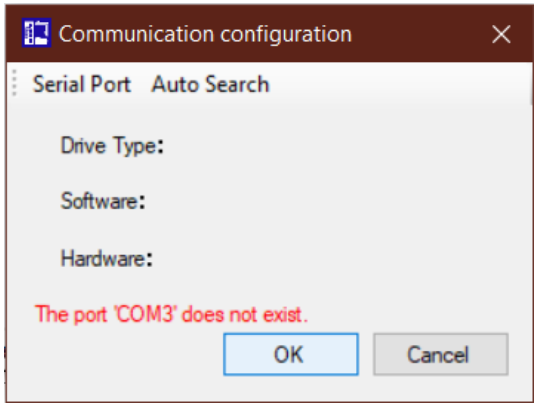


Рисунок 19. Окно соединения с драйвером

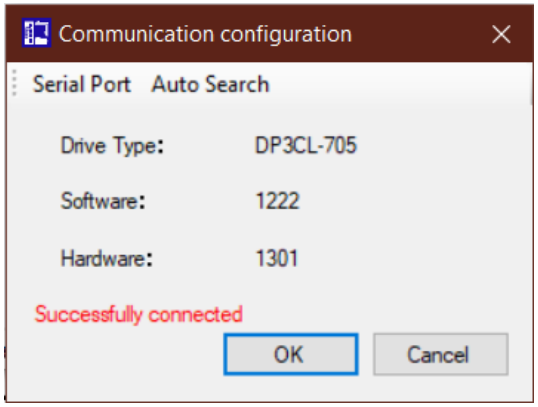


Рисунок 20. Успешное соединение с драйвером

После подключения к драйверу, утилита позволяет просматривать и редактировать параметры драйвера (рис. 21).

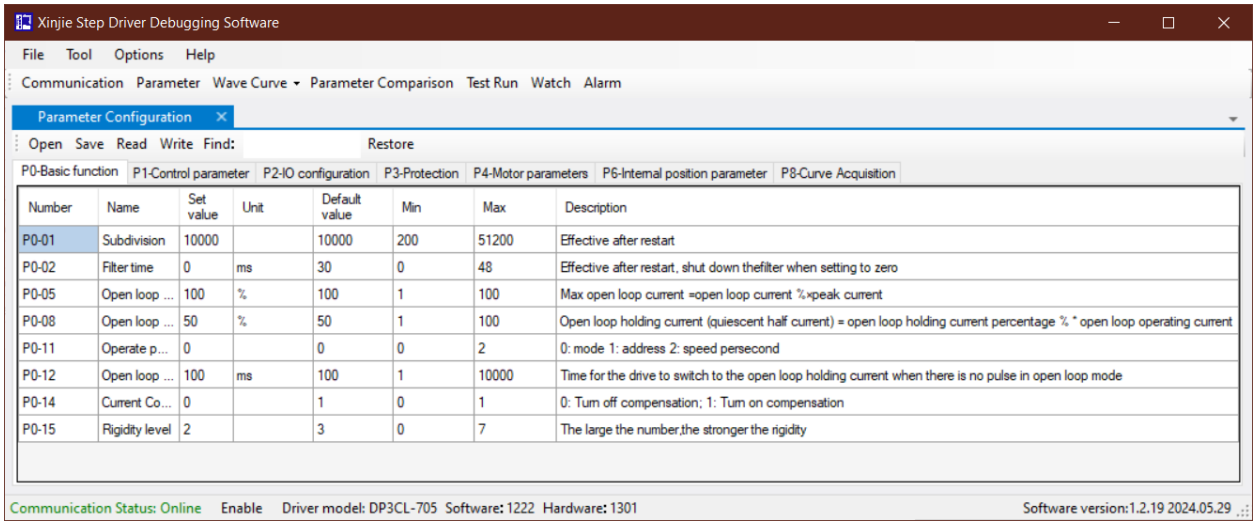
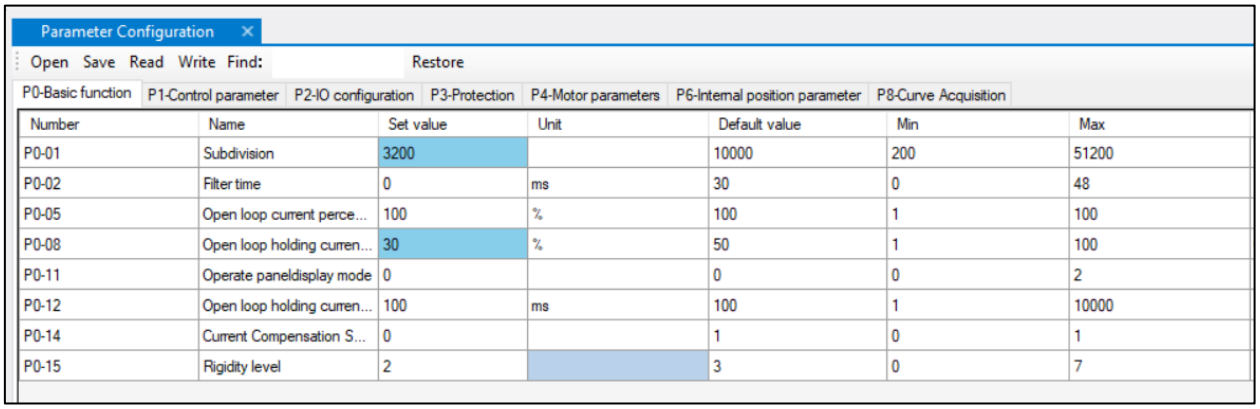


Рисунок 21. Параметры подключенного драйвера

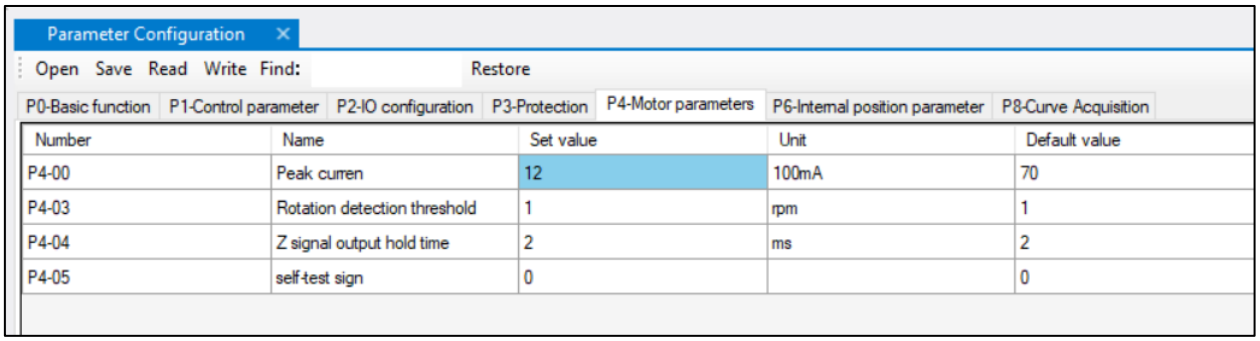
Сконфигурируем ранее рассмотренные параметры, при помощи утилиты. Для этого в столбце «Set value» изменим значения на необходимые. Установим: P4-00 – Peack current – рабочий ток двигателя – 12 (12 * 100 мА); P0-08 – Open loop honding current – ток

удержания – 30%; P0-01 – Subdivision – микрошаг – 3200 (3200 шагов на оборот, 1/16) (рис. 22-23).



Number	Name	Set value	Unit	Default value	Min	Max
P0-01	Subdivision	3200		10000	200	51200
P0-02	Filter time	0	ms	30	0	48
P0-05	Open loop current perce...	100	%	100	1	100
P0-08	Open loop holding curren...	30	%	50	1	100
P0-11	Operate paneldisplay mode	0		0	0	2
P0-12	Open loop holding curren...	100	ms	100	1	10000
P0-14	Current Compensation S...	0		1	0	1
P0-15	Rigidity level	2		3	0	7

Рисунок 22. Установка параметров P0-01, P0-08



Number	Name	Set value	Unit	Default value
P4-00	Peak curren	12	100mA	70
P4-03	Rotation detection threshold	1	rpm	1
P4-04	Z signal output hold time	2	ms	2
P4-05	self-test sign	0		0

Рисунок 23. Установка параметра P04-00

После установки параметров необходимо записать их в память драйвера, для этого нажмите кнопку «Write» на панели инструментов, откроется окно подтверждения, в котором необходимо подтвердить запись нажатием кнопки «Write» (рис. 24).

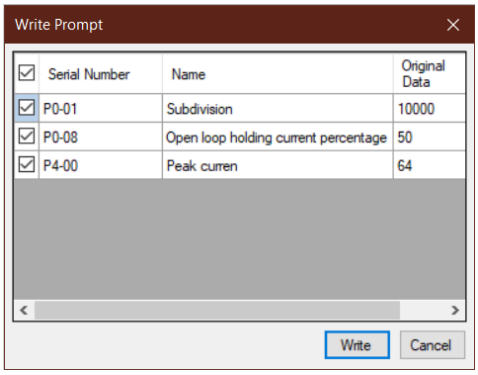


Рисунок 24. Подтверждение записи параметров

Успешная запись параметров завершится уведомлением (рис. 25):

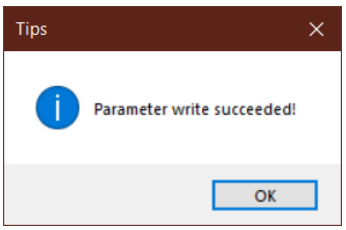


Рисунок 25. Успешная запись параметров

После записи параметров можно закрыть утилиту и отключить драйвер от компьютера. **(ВНИМАНИЕ! Несоблюдение порядка действий при подключении и отключении драйвера может привести к его повреждению!)**:

1. выключить питание драйвера;
2. отключить разъем USB преобразователя от компьютера;
3. отключить разъем MiniUSB переходника от драйвера;
4. отключить переходник MiniUSB->COM от преобразователя COM->USB.

На этом этапе мы закончили конфигурирование драйвера и подключили его к ПЛК, подготовив установку к работе с двигателем. Подробно ознакомится с параметрами драйвера и назначением остальных параметров можно в официальной документации (DP3CL series open loop bus stepping driver user manual).

1.3.Добавление драйвера шагового двигателя в проект CODESYS

Перед началом работы следует рассмотреть устройство интерфейса EtherCAT.

EtherCAT – это высокопроизводительный промышленный протокол связи на основе Ethernet с поддержкой различных топологий сети. Появился он в 2003 году, а с 2007 года стал международным стандартом.

Принцип работы протокола EtherCAT

Чтобы понять суть EtherCAT, нужно отойти от привычной картины, где Мастер (контроллер) по очереди шлет запросы каждому Ведомому (датчику/приводу). В классической схеме, например, в протоколах Modbus TCP, Profinet RT, кадр данных приходит на устройство, буферизируется, обрабатывается устройством, формируется ответ и отправляется обратно. Это похоже на почтовую службу: получил письмо, вскрыл, прочитал, написал ответ, запечатал, отправил.

EtherCAT работает по принципу «поезда, проходящего через станции без остановки». Представьте себе скоростной экспресс (EtherCAT-кадр), который проносится мимо платформ (устройств). Устройства не останавливают поезд, чтобы загрузить или выгрузить пассажиров. Они делают это «на лету». Такой механизм позволяет на высокой скорости управлять до 65 535 устройств в одной сети без ограничений в топологии сети: линия, шина, дерево, звезда или их комбинация. Причем для организации различных топологий не нужны хабы или коммутаторы, т.к. сами подчиненные устройства имеют несколько портов. Процесс обмена данными с 1000 распределенных цифровых входов-выходов EtherCAT занимает около 30 мкс, что является типичным для передачи 125 байт

на скорости 100 Мбит/с. Данные от 100 осей (драйверов шаговых двигателей или сервоприводов) могут обновляться со скоростью до 10 кГц. Схематически принцип работы EtherCAT показан на рисунке 26.

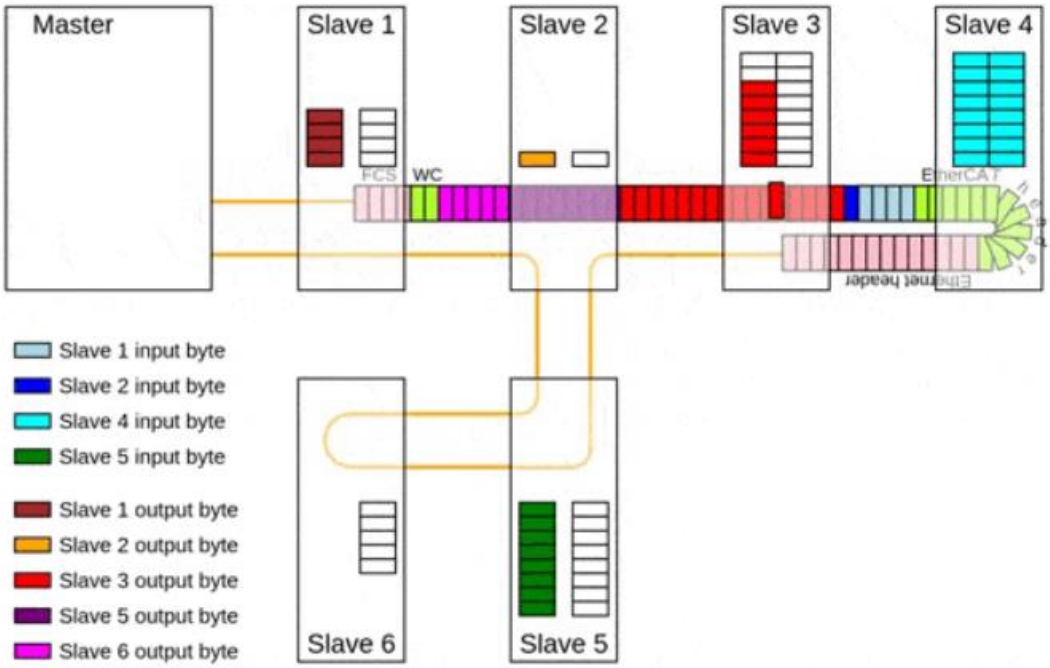


Рисунок 26. Движение кадра EtherCAT по сети

Архитектура «Суммирующего кадра» (Summation Frame) это ключевое отличие EtherCAT. Мастер отправляет всего один кадр на всю сеть. Этот кадр содержит данные не для одного устройства, а для всех устройств в сегменте одновременно. Кадр проходит последовательно через все устройства (Ведомые/Slaves) как по кольцу, но логически это часто выглядит как шина. Кадр представляет собой обычный пакет Ethernet, однако тип пакета на TCP и не UDP а 0x88A4, что означает кадр EtherCAT, внутри этого пакета находится кадр Ethercat, содержащий множество дейтаграмм, предназначенных для разных устройств. Устройство кадра показано на рисунке 27.

Детальный процесс обмена: «Чтение и запись на лету». Это самый важный раздел для понимания сути. Рассмотрим, что происходит на аппаратном уровне каждого Ведомого устройства (Slave Controller, ESC):

Представьте, что кадр (последовательность бит) входит в устройство через порт 0, а выходит через порт 1 (или обратно, в зависимости от топологии). Внутри устройства стоит специализированный интерфейсный контроллер (EtherCAT Slave Controller, ESC) или его IP-ядро в ПЛИС. Именно он реализует механизм «на лету», без участия основного процессора устройства.

Процесс обработки кадра выглядит следующим образом:

1. Обнаружение дейтаграммы. Ethernet-фрейм содержит внутри себя несколько EtherCAT-дейтаграмм (подкадров). Каждая дейтаграмма — это команда для определенной области памяти. Устройство «знает» адреса своей внутренней памяти (Register, FMMU — Fieldbus Memory Management Unit), которые ему выделил Мастер.
2. Аппаратная фильтрация. Аппаратный автомат ESC в реальном времени сравнивает адрес проходящей мимо дейтаграммы со своим списком адресов. Если адрес чужой — устройство никак не реагирует.
3. Чтение данных (Read). Если дейтаграмма помечена как «Команда чтения» и адрес совпал, ESC «врезается» в проходящий поток битов. Он копирует нужные байты из своей внутренней памяти (например, текущее положение энкодера) прямо в тело проходящего фрейма, заменяя пустые или старые данные. Основной процессор устройства (CPU) даже не знает, что это произошло именно в этот микросекундный момент.
4. Запись данных (Write). Если дейтаграмма помечена как «Команда записи», ESC выхватывает данные из проходящего фрейма и помещает их в свои выходные регистры (например, установка скорости двигателя).
5. Инкремент счетчика. После обработки каждой дейтаграммы устройство увеличивает специальное поле в фрейме — Working Counter (WKC). Это как расписка о получении.

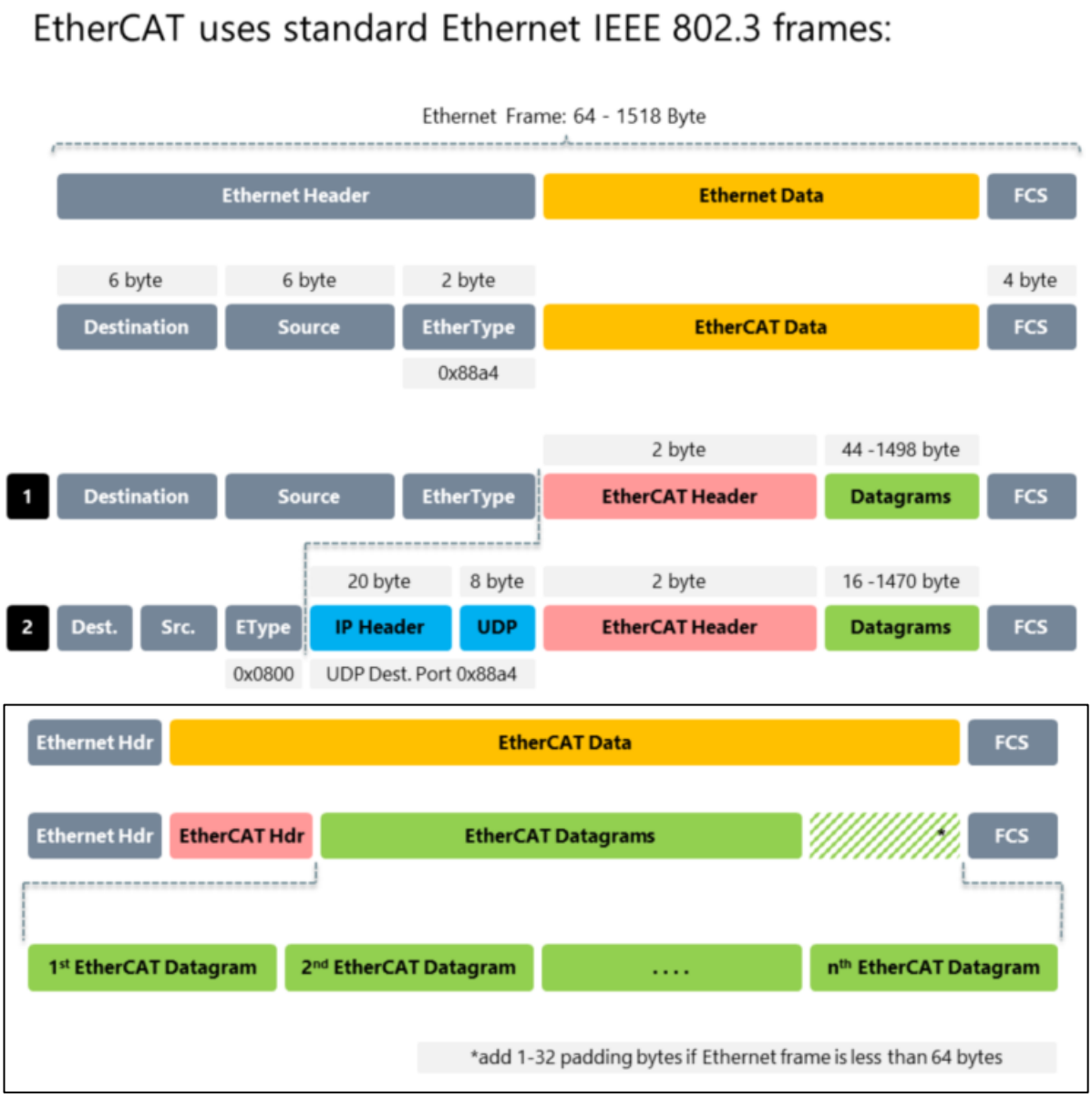


Рисунок 27. Структура EtherCAT кадра

Ключевой результат: задержка, вносимая каждым устройством, измеряется не миллисекундами, а наносекундами (задержка на чипе обычно составляет доли микросекунды — время прохождения сигнала через физический уровень и несколько тактов логики).

Как было сказано ранее каждое устройство содержит специализированный интерфейсный контроллер EtherCAT slave controller реализованный в виде отдельной интегральной микросхемы, который реализует логику работы интерфейса на аппаратном уровне, помимо этого он так же управляет направлением передачи кадра, каждый такой интерфейсный контроллер имеет четыре порта, и передает кадр по ним по очереди, при условии что к портам подключены устройства, если же устройства нет порт игнорируется, этот механизм позволяет строить разветвлённые сети без использования маршрутизаторов, используя только сами устройства. Логика работы такого чипа представлена на рисунке 28.

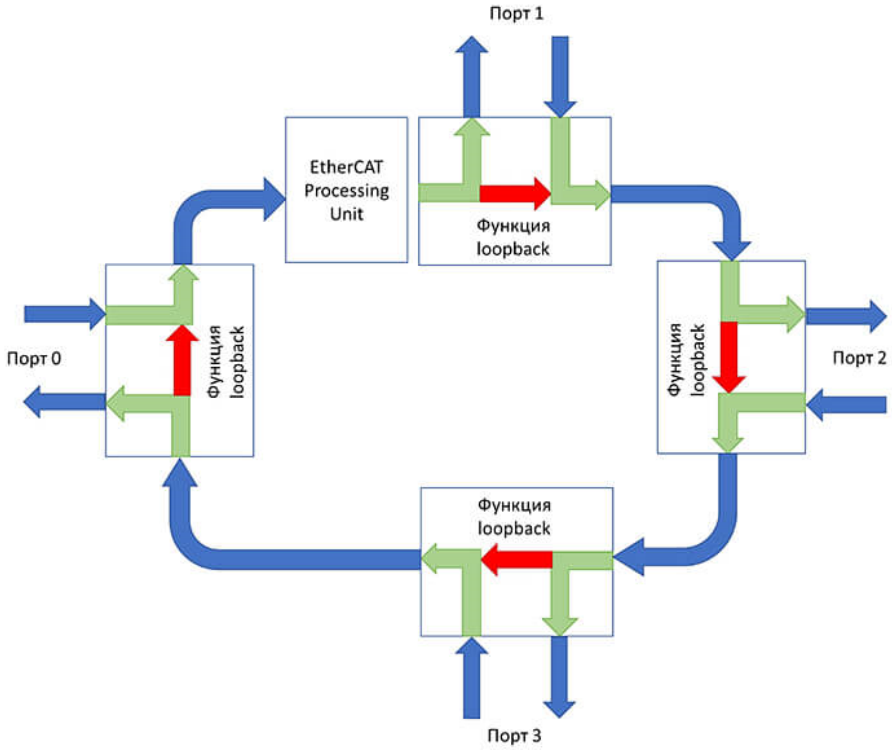


Рисунок 28. Структура EtherCAT slave controller

Пример обмена по протоколу EtherCAT

1. Мастер отправляет кадр: [Заголовок | Данные для привода_A (пока пусто) | Данные для датчика_B (пока пусто) | Данные для модуля_C | Конец].
2. Привод_A: принимает кадр. Видит команду "Read Position". Он не тормозит кадр, а «на лету» вписывает в ячейку «Данные для привода_A» свое текущее значение угла (Например, 100 градусов). Кадр уходит дальше.
3. Датчик_B: принимает кадр. Видит команду "Read Status" и "Write Config". Он вписывает свой статус в ячейку «Данные для датчика_B» и тут же забирает байты конфигурации из ячейки «Данные для модуля_C» (если там была часть данных для него).
4. Модуль_C: забирает оставшиеся данные, вписывает свои ответы.
5. Возвращение к Мастеру: Кадр проходит через конечную точку (или логическое кольцо) и возвращается к Мастеру. Мастер получает тот же самый кадр, но уже заполненный ответами от всех устройств.

Распределенные часы (Distributed Clocks — DC)

В системах управления движением (сервоприводы, роботы) критично, чтобы все оси выполнили команду с погрешностью менее 1 микросекунды. Обычного циклического обмена для этого недостаточно из-за джиттера (от англ. jitter — «дрожание») (нестабильности времени цикла). EtherCAT решает это механизмом распределенных часов (DC), мастер выбирает одни часы (обычно у первого Ведомого) за опорные, все остальные часы в сети аппаратно синхронизируются с опорными. Для синхронизации замеряются задержки прохождения сигнала до каждого устройства и по ним рассчитывается отклонение времени устройства от опорного, в результате все устройства имеют единое

системное время с точностью более 1 мкс. В процессе работы мастер отправляет команду: «Выполнить новое положение в 10:00:00.000000», все приводы начнут движение синхронно несмотря на то, что фрейм физически проходил через них последовательно и в разное время, благодаря тому что их время синхронизировано.

Завершим рассмотрение базовое рассмотрение интерфейса EtherCAT и перейдём к подключению драйвера шагового двигателя к ПЛК с его помощью.

Откройте проект, созданный в подразделе 1.1.

В первую очередь для работы с драйвером по протоколу EtherCAT необходимо в дерево проекта добавить устройство EtherCAT master SoftMoution, которое отвечает за взаимодействие с интерфейсом и совместимо с библиотекой SoftMoution.

Для добавления устройства кликните правой кнопкой мыши по объекту ПЛК в дереве проекта (Device), в выпадающем списке выберите «Добавить устройство», после чего в открывшемся окне выберите Промышленные сети -> EtherCAT -> Мастер -> EtherCAT master SoftMoution и нажмите кнопку «Добавить устройство» (рисунок 29).

Дважды кликните по добавленному устройству, и перейдите в раздел общее, где установить MAC адрес источника (интерфейс ПЛК) нажав на кнопку «Обзор» напротив пункта «Адрес источника», в открывшемся окне выберите интерфейс Eth1 и нажмите «ОК» (рисунок 30).

Отключите питание драйвера шагового двигателя, включите питание ПЛК и затем дождитесь загрузки ПЛК подайте питание на драйвер.

Выполните подключение к контроллеру (Онлайн -> Логин) как описано в подразделе 1.1. Согласитесь с загрузкой программы в всплывающем диалоговом окне.

ВНИМАНИЕ! В случае если подключится не удается проверьте настройки сетевой карты компьютера как это описано в конце подраздела 1.1.

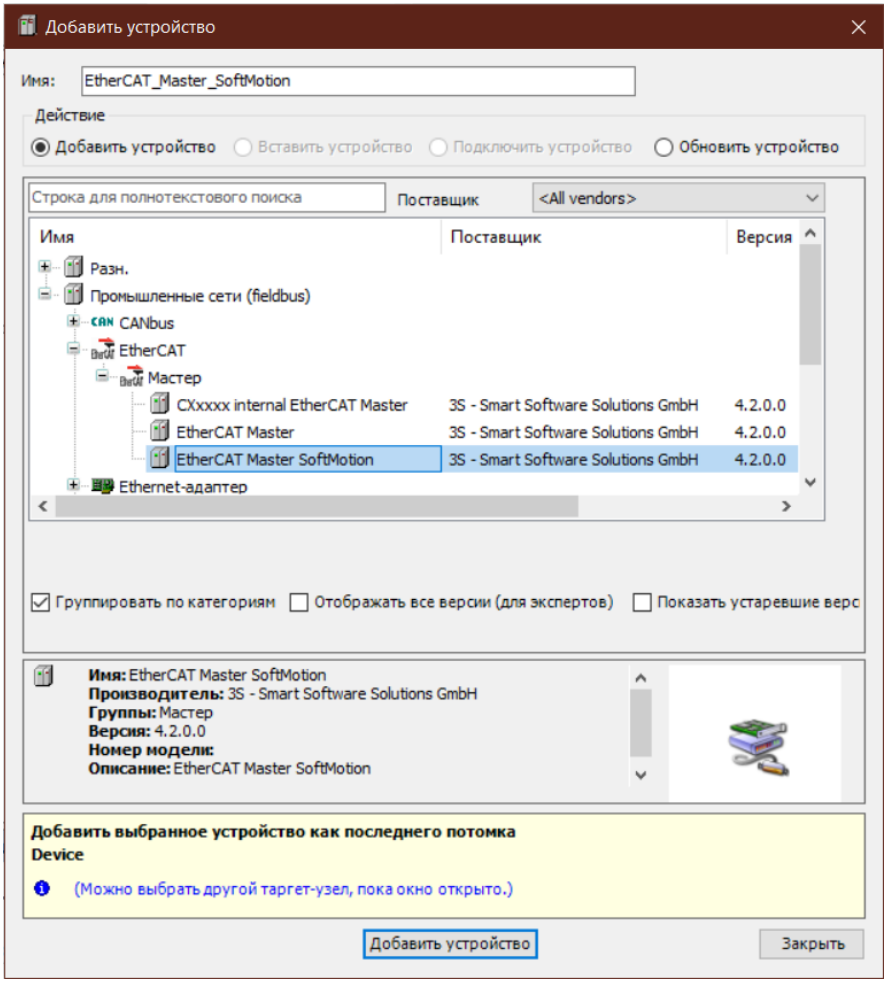


Рисунок 29. Добавление EtherCAT master

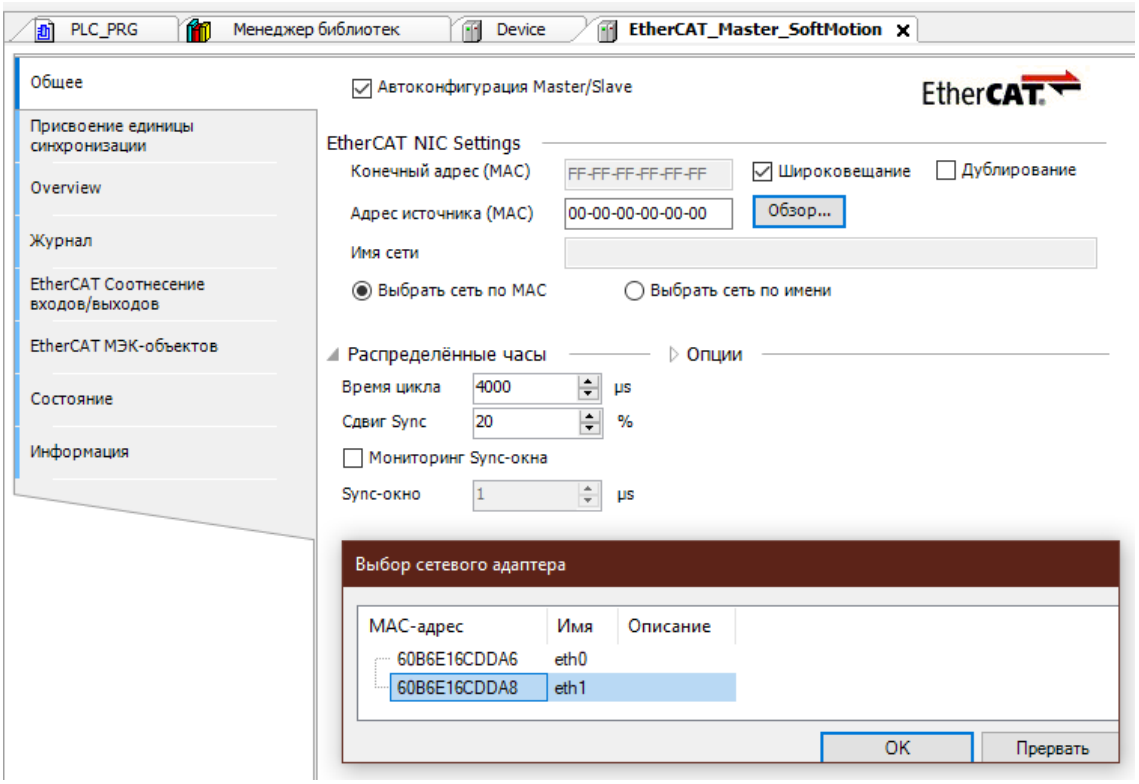


Рисунок 30. Выбор интерфейса для работы устройства

После успешного подключения к ПЛК, в режиме онлайн, нажмите правой кнопкой мыши на объект EtherCAT_master в дереве проекта, в выпадающем списке выберите «Поиск устройств», откроется окно следующего вида (рис. 31).

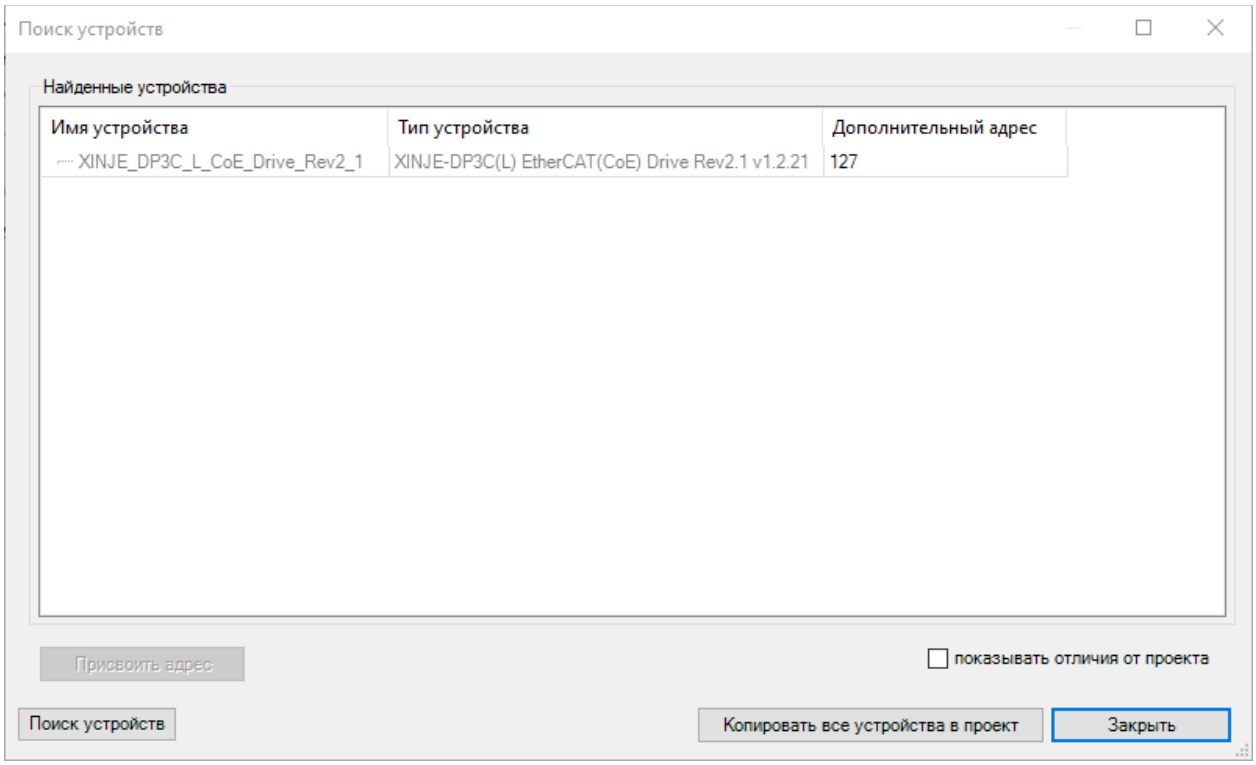


Рисунок 31. Поиск устройств EtherCAT

Выделите драйвер кликнув по нему и нажмите кнопку копировать в проект. В дереве проекта должен появиться объект драйвера как показано на рисунке 29, отключитесь от ПЛК (выйдите из режима онлайн в режим редактирования проекта).

ВАЖНО! При подключении нескольких драйверов по EtherCAT используйте режим адресации только по физическому номеру устройства на шине, при использовании псевдонима станции для адресации драйверов, в случае если драйверы идут не по порядку на шине, возможна нестабильная работа и проблемы с синхронизацией драйверов выражающаяся в виде звука ударов, рывков и дрожи отдельных приводов при движении.

Следующим шагом необходимо добавить ось SoftMoution для управления драйвером и шаговым двигателем, ось является объектом, который превращает миллиметры или градусы перемещения в безразмерные шаги двигателя и команды для драйвера, а также позволяет управлять двигателем при помощи специальных функциональных блоков.

Для того чтобы добавить ось кликните правой кнопкой мыши на объект драйера в дереве проекта и выберите пункт «Add SoftMoution CiA402 Axis» в выпадающем списке. CiA402 – это стандарт CAN in Automation, устанавливающий стандартные способы управления электроприводами при помощи ПЛК. После добавления оси дерево проекта должно выглядеть следующим образом:

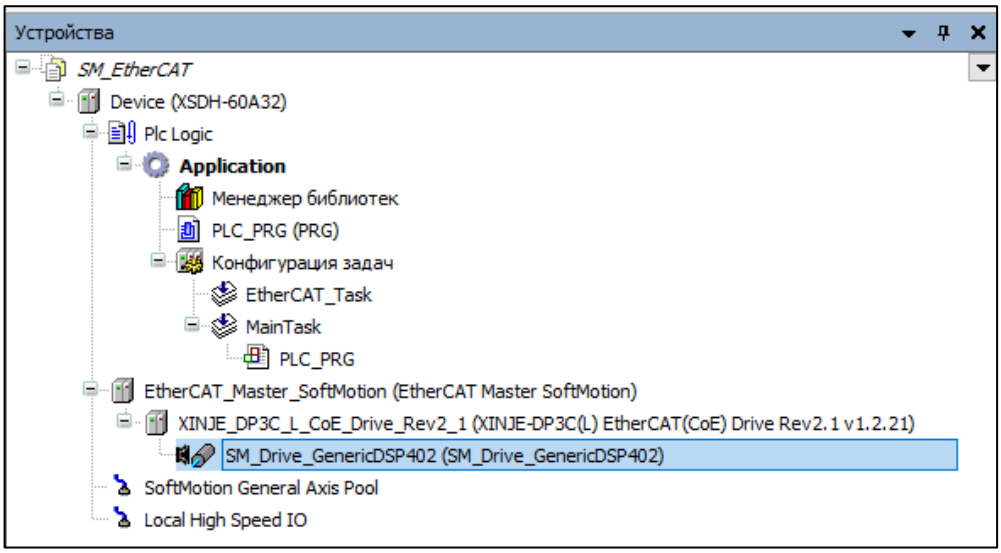


Рисунок 32. Добавление оси

Откройте двойным щелчком по объекту в дереве проекта ось, и перейдите во вкладку «General», на данной вкладке установим динамические ограничения оси, это velocity – максимальная скорость оси в единицах измерения в секунду, определим её как 360 градусов в секунду, что соответствует одному обороту двигателя; Acceleration и Deceleration – ускорение и замедление в единицах измерения в секунду в квадрате, определим как 500 градусов в секунду в квадрате, Jerk – рывок, то есть скорость изменения ускорения для профилей кроме Trapezoid (трапециевидный). Velocity ramp type – профиль ускорения, установим Trapezoid (рисунок 33), то есть скорость изменяется линейно, ускорение изменяется мгновенно, это простейший и наиболее распространённый профиль. Остальные профили так же обеспечивают плавное изменение ускорения, что полезно при управлении тяжелыми или крайне инертными в плане движения осями. В результате получим конфигурацию, представленную на рисунке 34.

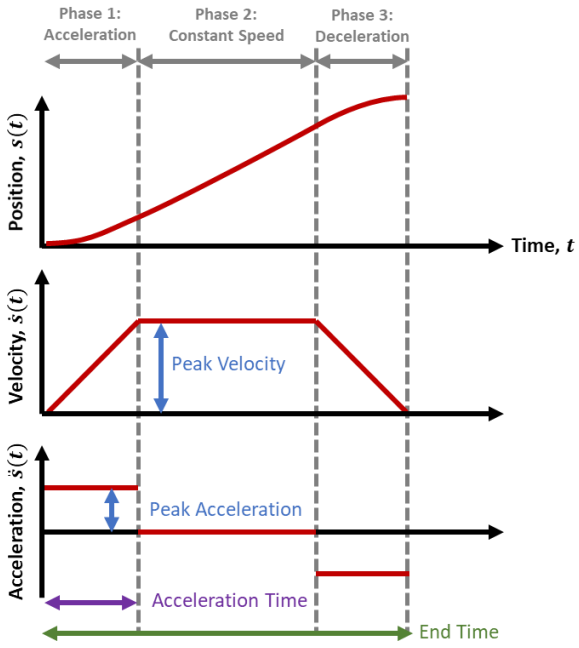


Рисунок 33. Трапециевидный профиль ускорения

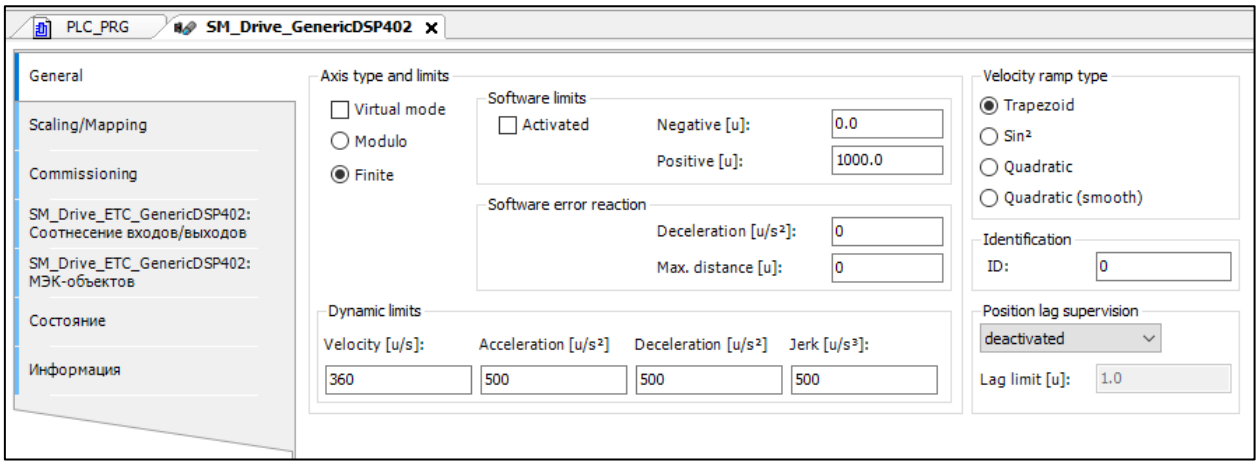


Рисунок 34. Настройки динамических ограничений оси

Перейдите на вкладку «Scaling/Mapping» и установите соотношение между шагами шагового двигателя и единицами измерения, поскольку к драйверу подключен двигатель без редуктора и мы далее будем рассматривать его как вращательную ось, на один оборот приходится 3200 шагов (настройка микрошага драйвера 1/16, 200 физических шагов на оборот двигателя), при этом в одном обороте 360 градусов, так как редуктора нет то число оборотов двигателя равно числу оборотов оси. Введём эти параметры в поля раздела «Scaling» (рис. 35).

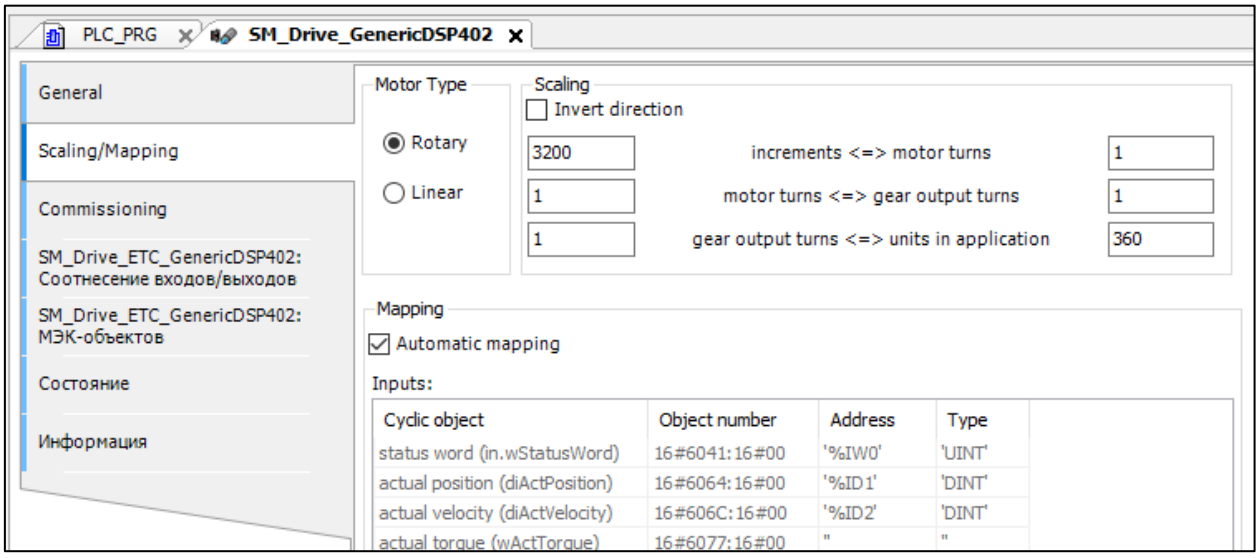


Рисунок 35. Настройка масштабирования оси

Для проведения проверки работы настроенной оси, прейдите в файл PLC_PRG. Для проверки работоспособности соединения между драйвером и ПЛК, добавим блок управляющий удержанием двигателя, то есть управляющий питанием обмоток двигателя. Для этого добавьте в область программы пустой «Элемент», кликните на вопросительны знаки на нем и нажмите на троеточие слева, откроется ассистент ввода выберите блок MC_Power как показано на рисунке 36, затем нажмите «ОК», в поле над блоком введите

название его экземпляра (объекта), например, Axis1Power. В открывшемся окне нажмите «ОК» (рис. 37).

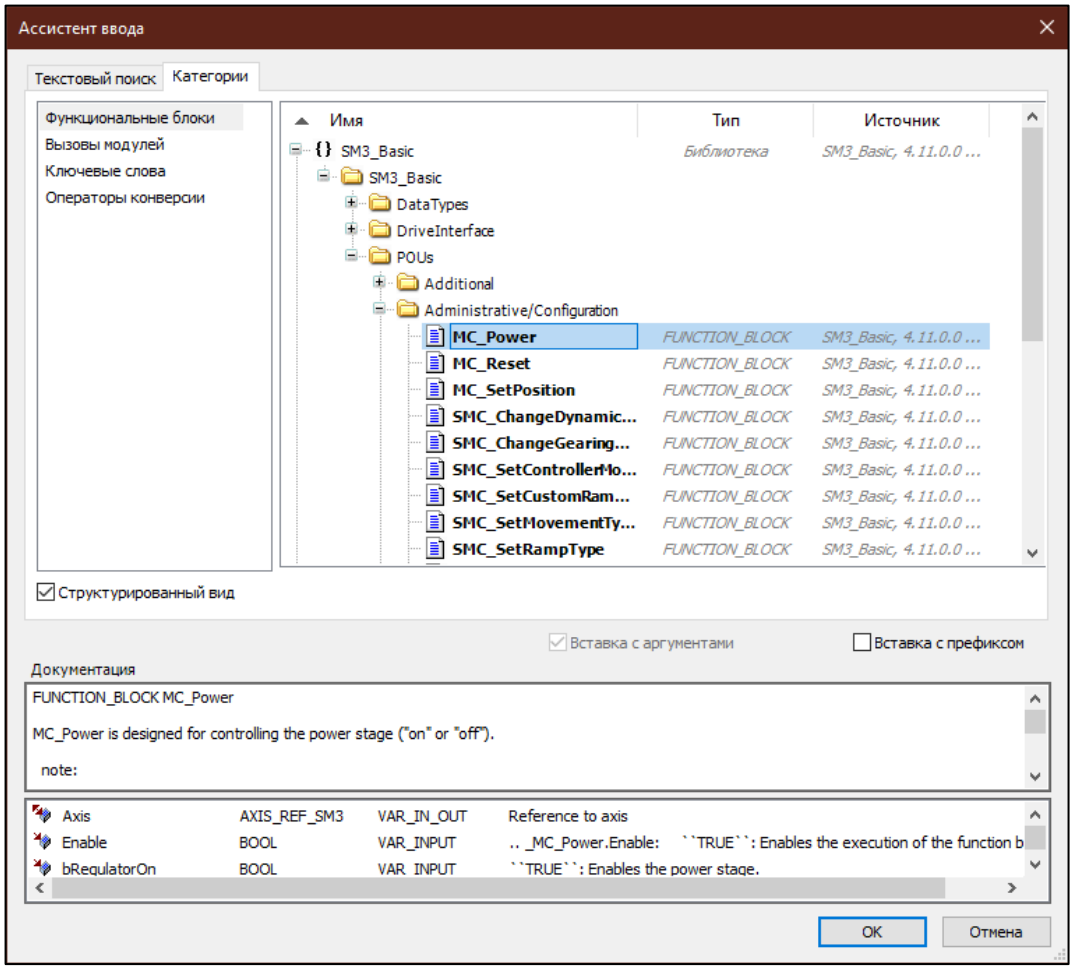


Рисунок 36. Выбор блока MC_Power

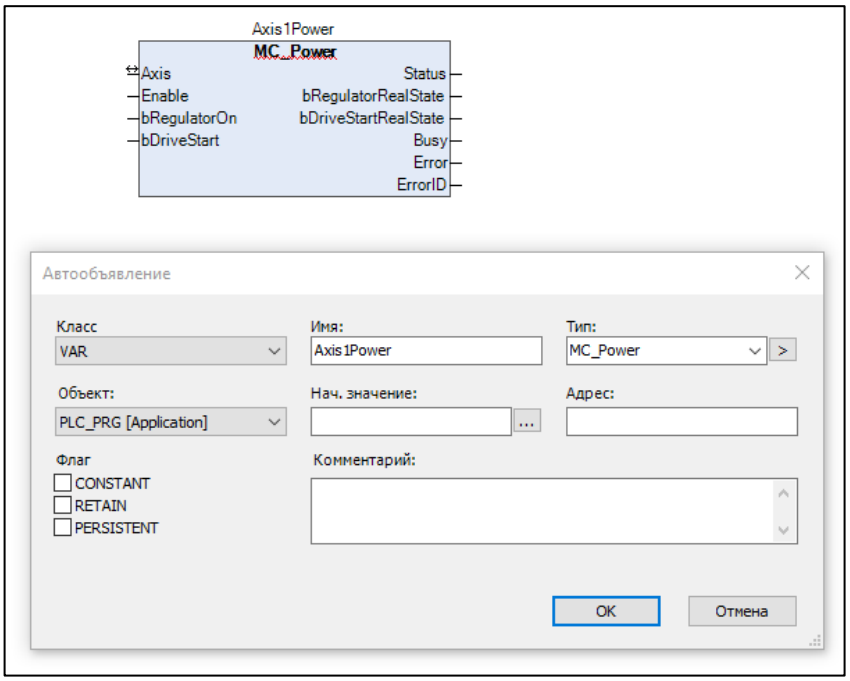


Рисунок 37. Создание экземпляра блока MC_Power

Последним шагом добавьте «Ввод» и уже знакомым способом вызовите ассистент ввода, в котором выберите ось, так же нажмите «ОК» (рисунок 38). Соедините «Ввод» с входом Axis блока MC_Power, протянув линию соединения мышью. В результате должна получиться программа, представленная на рисунке 39.

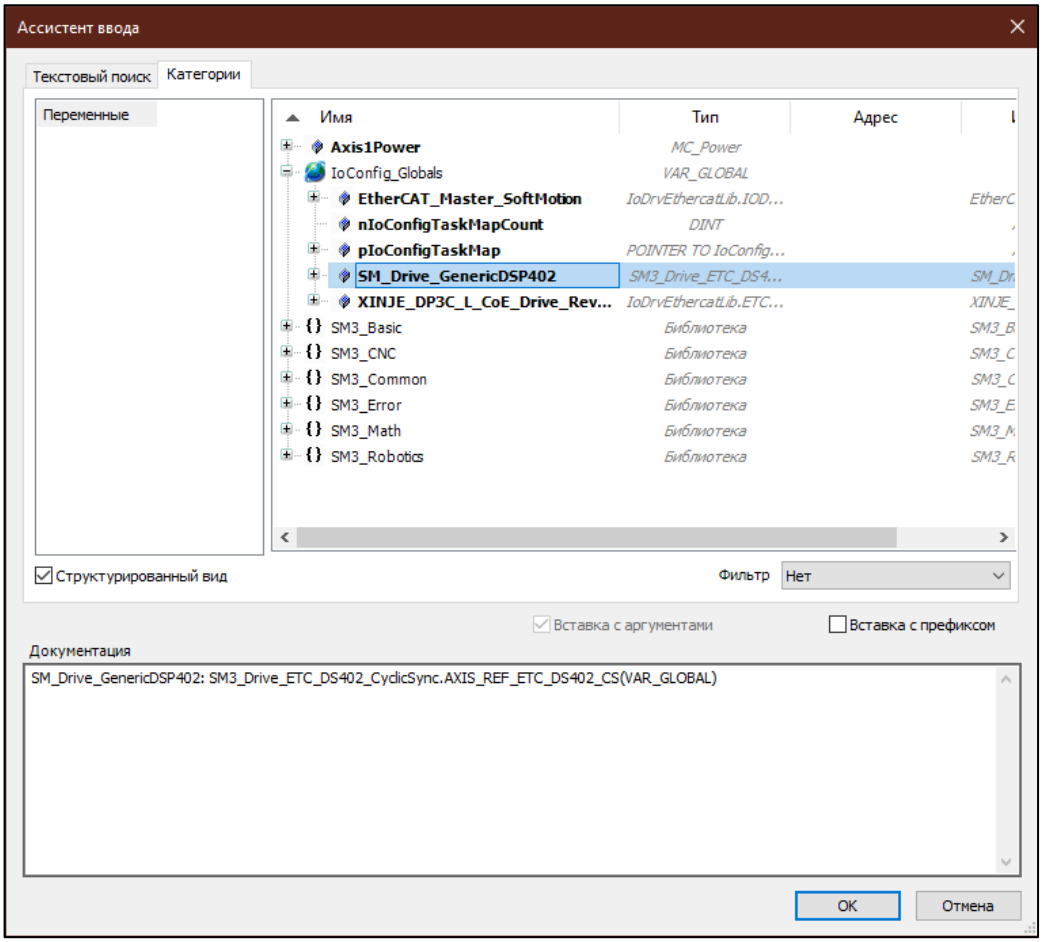


Рисунок 38. Выбор оси

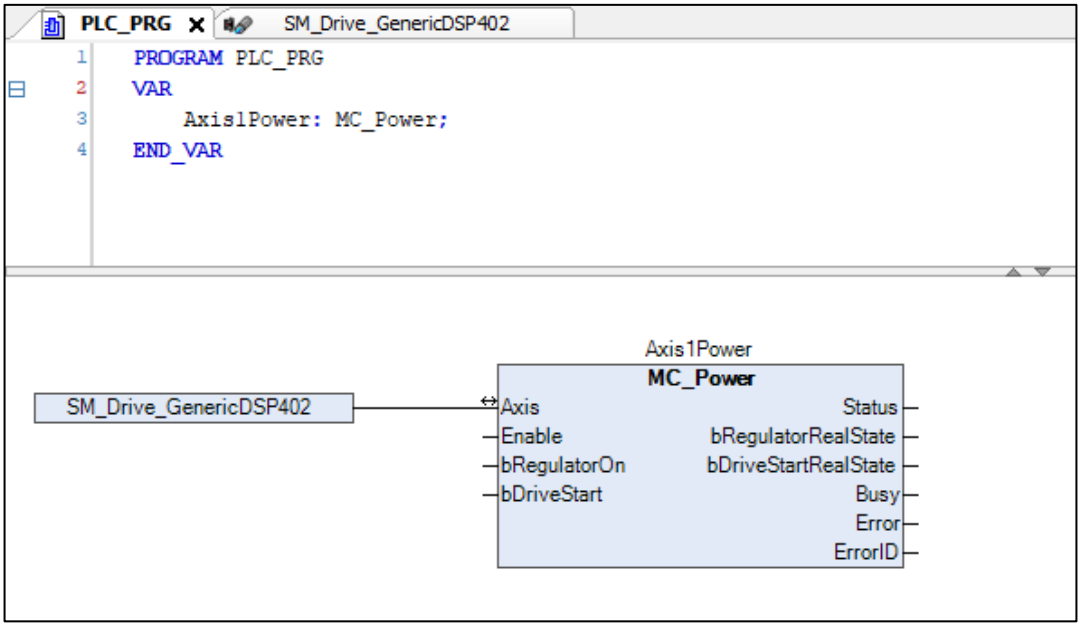


Рисунок 39. Программа управления питанием оси

Подключитесь к ПЛК, запустите программу нажав на треугольник рядом с кнопкой подключения к контроллеру на панели инструментов, после чего дождитесь инициализации драйвера и оси, напротив них должны появиться зелёные значки синхронизации (рис. 40).

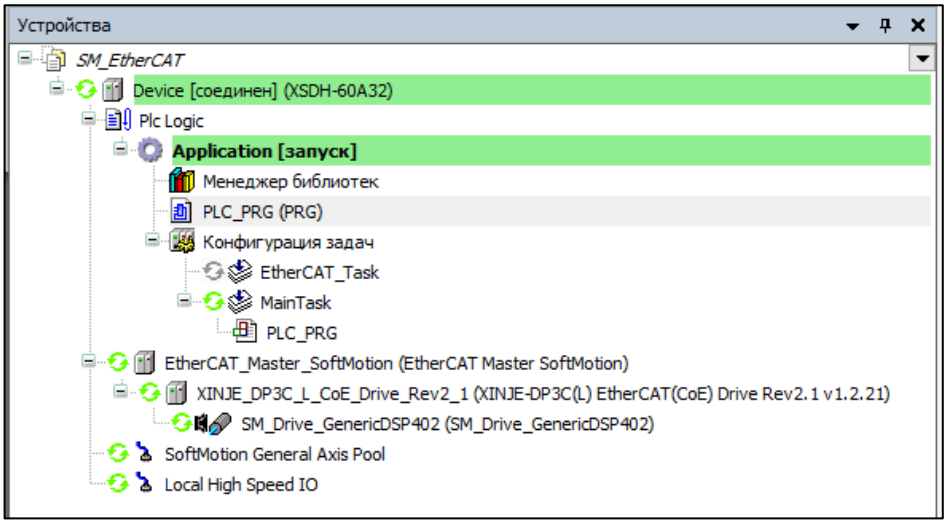


Рисунок 40. Успешная инициализация оси и драйвера

В области переменных PLC_PRG раскройте Axis1Power, нажав на значок плюс. Установите подготовленное значение как показано на рисунке 41, убедитесь, что вал двигателя свободно вращается, нажмите сочетание клавиш Ctrl+F7, это действие запишет подготовленные значения, которые запустят блок MC_Power и включат питание двигателя. Результат представлен на рисунке 42, проверьте вал двигателя, он должен перестать проворачиваться, поскольку двигатель перешел в режим удержания и ротор зафиксировался магнитным полем обмоток, так же на блоке питания должен возрасти потребляемый ток. Подробно про работу с блоками библиотеки SoftMoution будет рассказано в следующем подразделе.

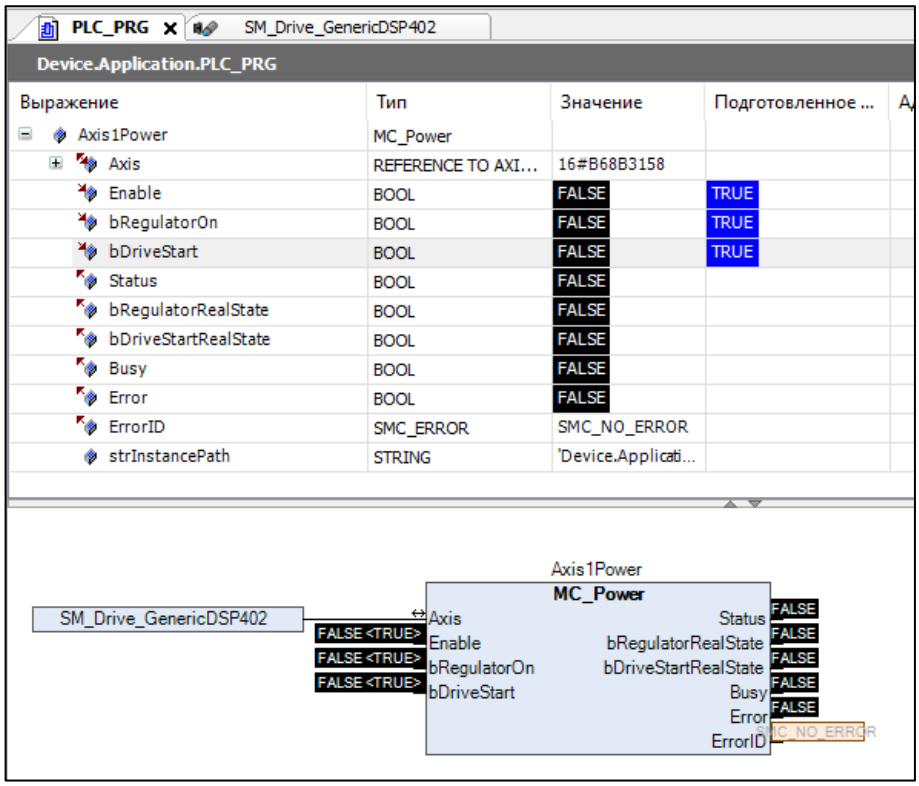


Рисунок 41. Подготовка значений для записи

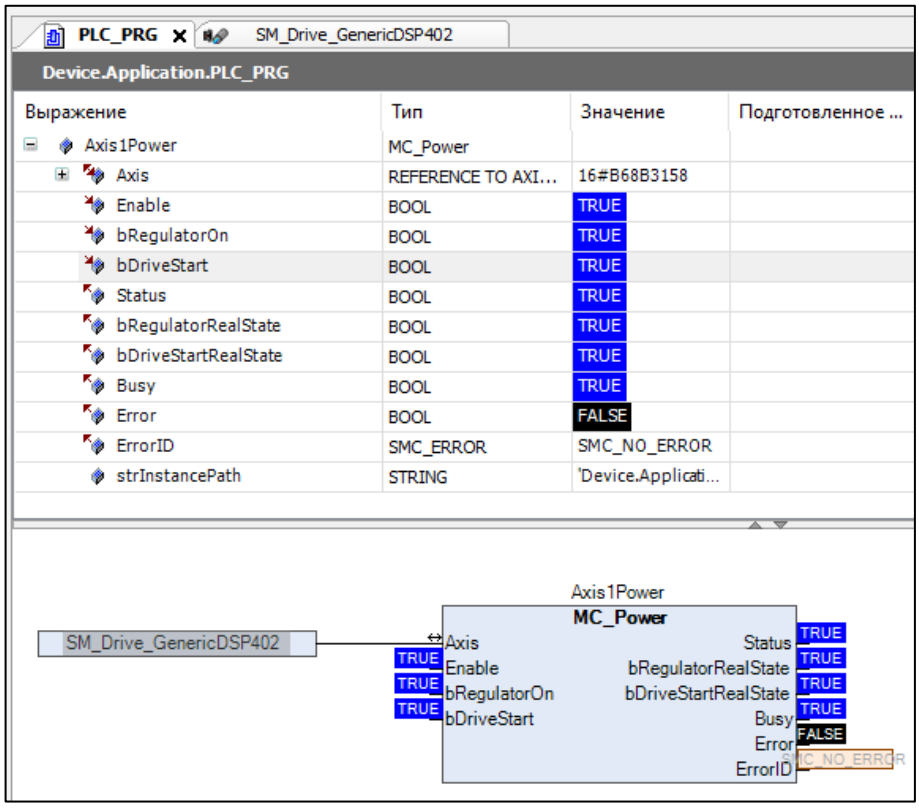


Рисунок 42. Записанные значения, включение режима удержания

1.4.Управление перемещением осей

Управление перемещением осей в библиотеке SoftMoution, выполняется при помощи функциональных блоков. Все блоки библиотеки управляющие движением имеют входы:

1. «Axis» на который необходимо подать ссылку, на ось которой управляет блок, в случае если ссылка не указана, при компиляции возникнет ошибка;
2. «Enable» который управляет исполнением блока, в зависимости от блока он либо исполняется пока на этом входе «Истина» или выполняется один раз по фронту сигнала.

Все блоки имеют выходы:

1. «Done», выдает значение «Истина», когда блок завершает выполнение своей функции, например, движение к указанной координате завершилось, значение «Истина» сохраняется до тех пор, пока на входе «Enable» значение «Истина»;
2. «Busy», выдает значение «Истина», когда блок выполняет свою функцию, то есть блок занят;
3. «Error», выдает значение «Истина», когда блок при выполнении своей функции перешел в состояние ошибки, например, при попытке перемещения оси оказалось, что ось выключена;
4. «ErrorID», номер ошибки в виде перечисления SMC_ERROR, данное перечисление содержит более тысячи ошибок, для расшифровки ошибки можно либо посмотреть её название в перечислении, что зачастую уже позволяет понять, что именно произошло, либо открыть документацию на официальном сайте helpme-codesys.com, и прочитать описание ошибки.

ВНИМАНИЕ! Блоки управляющие движением должны вызываться только в задаче, связанной с мастером EtherCAT, при асинхронном вызове из другой задачи вместо движения, ось постоянно будет выдавать ошибку SMC_AXIS_NOT_READY_FOR_MOUTION. Это так же означает что, если вам необходимо создать свою реализацию некоторого движения для повторного использования (функцию или функциональный блок), вы можете использовать только синхронные функции, программы, методы и действия, функциональные блоки не подойдут в связи с их асинхронностью.

Ось имеет встроенный конечный автомат, который управляет ей работой его диаграмма представлена на рисунке 43, на этом рисунке отмечены:

- Note 1: Из любого состояния, при возникновении ошибки.
- Note 2: Из любого состояния. MC_Power.bRegulatorOn = FALSE . Если нет ошибки.
- Note 3: MC_Reset и MC_Power.Status = FALSE
- Note 4: MC_Reset и MC_Power.Status = TRUE и MC_Power.bRegulatorON = TRUE
- Note 5: MC_Power.bRegulatorOn = TRUE и MC_Power.Status = TRUE
- Note 6: MC_Stop.Done = TRUE и MC_Stop.Execute = FALSE

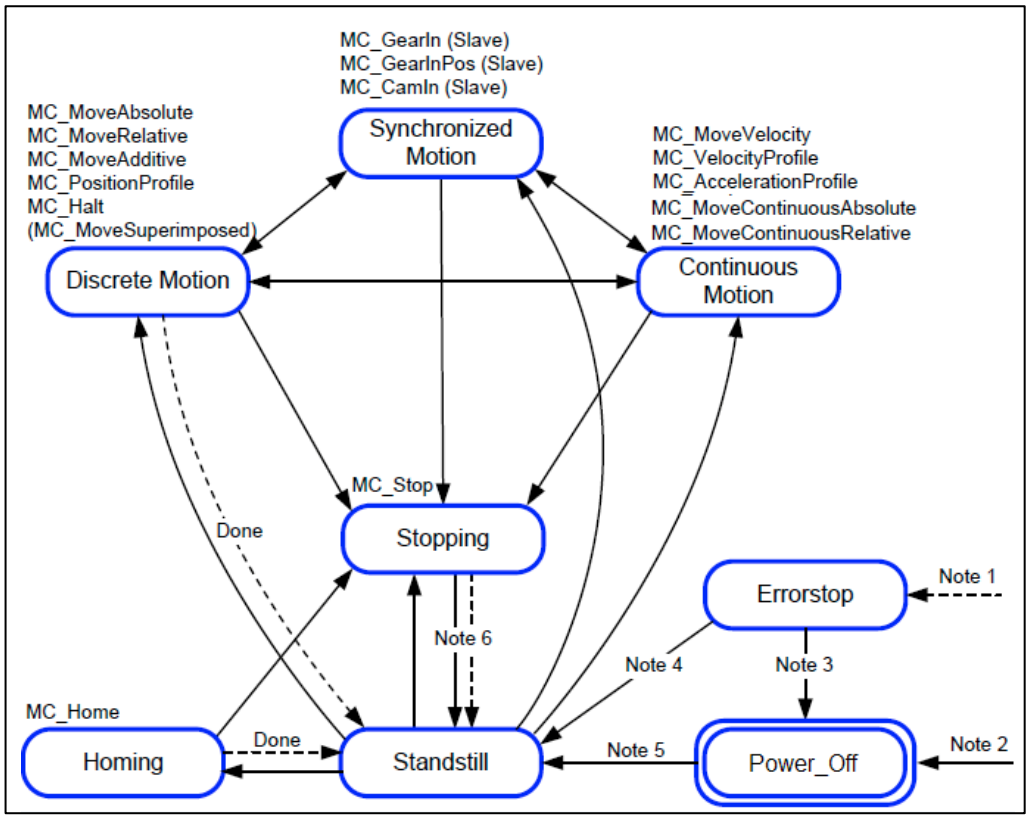


Рисунок 43. Диаграмма состояний оси

Перед началом движения необходимо включить питание оси, что было сделано в конце предыдущего подраздела. Однако, стоит дополнить информацию о блоке MC_Power. Данный блок имеет входы bRegulatorOn, значение «Истина» на котором включает питание двигателя оси и bDriveStart, значение «Истина» на котором отпускает физический тормоз привода, если он установлен, если тормоза нет, для запуска движения всё равное необходимо подать на данный вход значение «Истина».

Так же для сброса ошибок понадобится блок MC_Reset, данный блок при подаче сигнала «Enable» по его фронту сбрасывает ошибку оси.

Перейдём к управлению непосредственно движением оси. Ось в библиотеке SoftMoution может управляется четырьмя основными способами.

1. Относительное перемещение оси – выполняется при помощи блока MC_MoveRelative, выполняет перемещение оси в координату «Distance», заданную относительно текущего положения, то есть если ось находится в позиции 10 и получена команда двигаться 20, то итоговая координата оси после движения будет 30.
2. Абсолютное перемещение оси – выполняется при помощи блока MC_MoveAbsolute, выполняет перемещение оси в координату «Position», заданную относительно начала координат оси, то есть если ось находится в позиции 10 и получена команда двигаться 20, то итоговая координата оси после движения будет 20.
3. Движение оси с заданной скоростью – выполняется при помощи блока MC_MoveVelocity, данный блок запускает движение оси с заданной скоростью,

после начала движения, установка входа «Enable» в состояние «Ложь» не останавливает перемещение оси. Направление движения оси задается входом «Direction», движение может быть «positive», «negative», «current», что соответствует движению в положительном, отрицательном направлениях и направлении последнего движения оси перед вызовом блока (по умолчанию положительное). Для остановки оси необходимо использовать блок MC_Stop который останавливает движение и имеет максимальный приоритет над другими блоками, то есть после начала торможения нельзя его прервать и начать движение при помощи другого блока, или блок MC_Halt который так же останавливает движение оси, однако имеет более низкий приоритет, и его работа может быть прервана любым блоком запускающим движение. Блоки MC_Stop и MC_Halt начинают остановку оси по фронту входа «Enable». **ВАЖНО!** Нельзя запускать одновременно несколько экземпляров блоков MC_Stop работающих с одной осью, это приведёт к ошибке. MC_Halt такого ограничения не имеет.

4. Синхронное движение – данный способ применяется для более сложных ситуаций, например для реализации следования по криволинейной траектории, синхронного движения нескольких осей и т.п. В рамках данной работы этот тип не будет рассматриваться подробно.

Блоки управления движением имеют следующий входы:

1. «Velocity» - скорость движения оси;
2. «Acceleration» - ускорение разгона оси;
3. «Deceleration» - ускорение замедления оси;
4. «Jerk» - рывок, оси (скорость изменения ускорения), имеет значение только для профилей ускорения с плавным изменением ускорения;
5. «Direction» - для MC_MoveVelocity это направление движения, для остальных блоков при использовании физических приводов, не используется;
6. «BufferMode» - режим буферизации, указывает как обрабатывать одновременные вызовы нескольких блоков движения для одной оси, по умолчанию «Aborting», то есть новый блок отменяет действие предыдущего.

ВАЖНО! Перед запуском блока, необходимо задать скорость, ускорения и рывок на его входах, если этого не сделать, блок выдаст ошибку и движение не начнется.

Перейдём к практической работе с движением оси, для этого в первую очередь необходимо перенести исполнение программы «PLC_PRG» из задачи «MainTask» в задачу «EtherCAT_Task», для синхронного вызова блоков управления движением с работой EtherCAT master. Для этого в дереве проекта мышкой перетащите программу «PLC_PRG» в задачу «EtherCAT_Task» как показано на рисунке 44.

Далее добавьте в «PLC_PRG» блоки MC_Reset, MC_MoveRelative, MC_MoveAbsolute, MC_MoveAbsolute, MC_Halt, на вход «Axis» данных блоков подайте ссылку на ось как показано на рисунке 45. Добавление блоков аналогично добавленному ранее MC_Power.

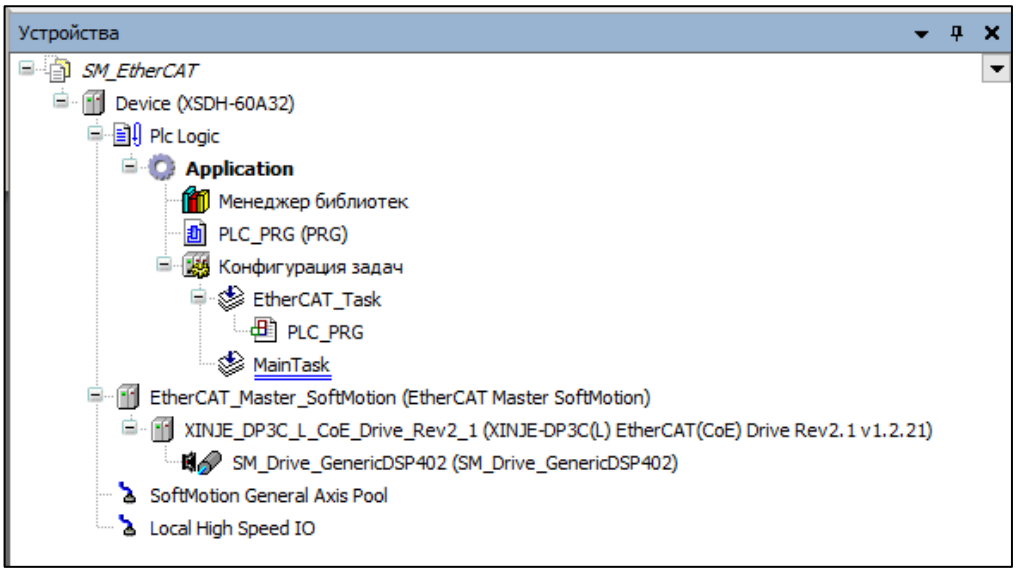


Рисунок 44. Перенос «PLC_PRG» в задачу «EtherCAT_Task»

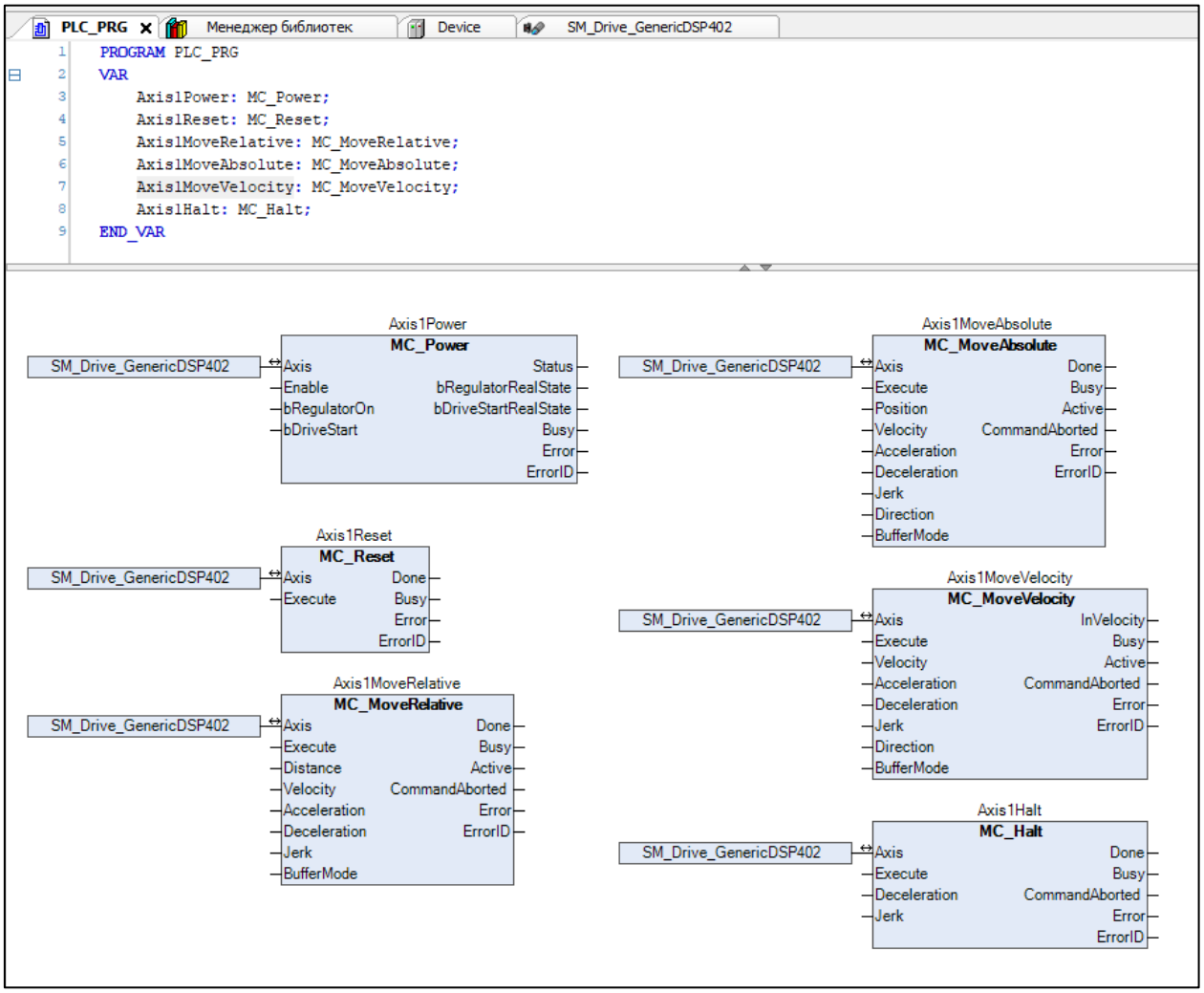


Рисунок 45. Добавление функциональных блоков управления движением

После добавления блоков, подключитесь к контроллеру и загрузите программу.

Дважды кликните по оси в дереве проекта в режиме онлайн, чтобы открыть её, на вкладке «General» в блоке «Online», вы можете отслеживать текущее положение оси, параметры её движения, состояние её конечного автомата, а также возникающие ошибки.

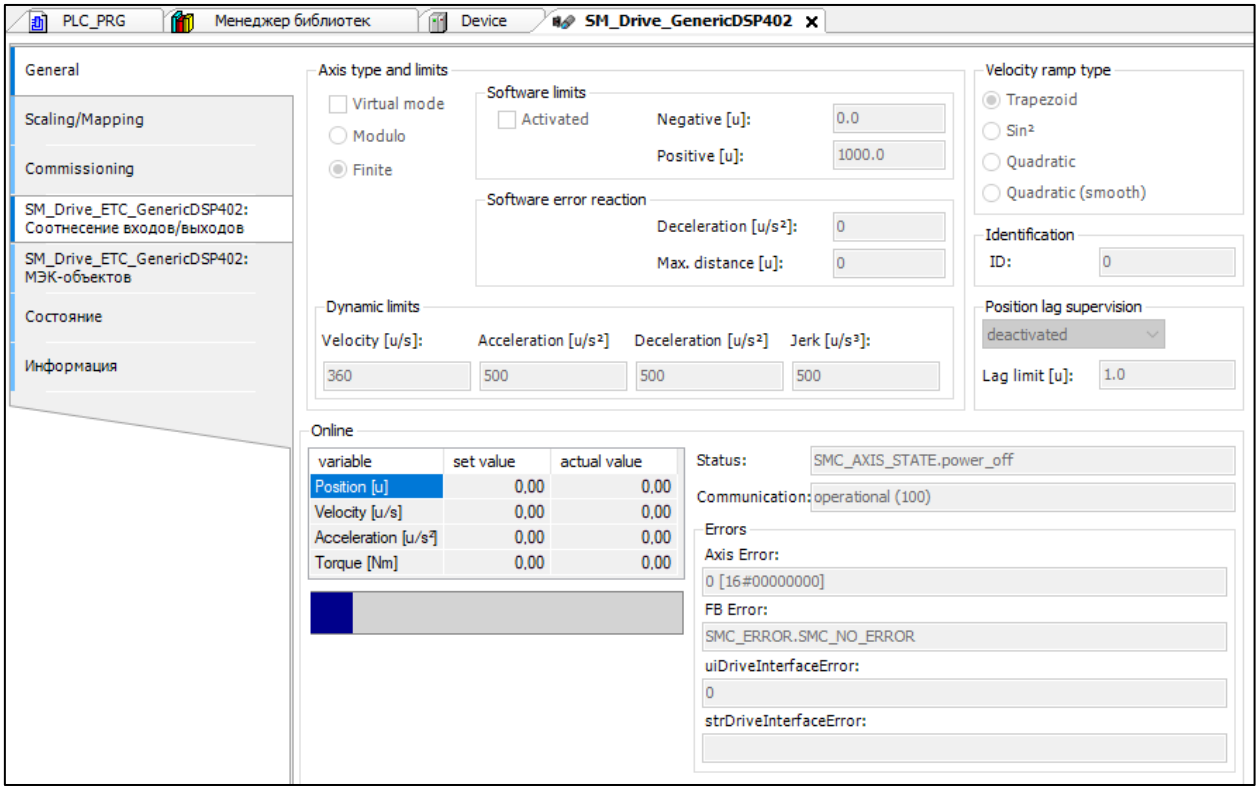


Рисунок 46. Параметры оси в режиме онлайн

Алгоритм запуска выполнения блока:

1. установите подготовленные значения для записи на входах блока «Enable» = «Ложь»;
2. запишите значения в контроллер (Ctrl+F7);
3. установите «Enable» = «Истина»;
4. запишите значения в контроллер (Ctrl+F7);
5. блок начнет выполнение, ось должна прийти в движение, остановится, сбросится ошибка и т.п.

При возникновении ошибки запустите блок MC_Reset для её сброса. Используйте скорость, ускорения и рывок в пределах от ноля до максимальных значений, заданных при настройке оси. Перед началом движения включите питание оси при помощи MC_Power.

Добейтесь перемещения оси при помощи ручного ввода значений на 10 градусов относительно текущего положения по часовой и против часовой стрелки, перемещения в ноль оси, а затем в координату 360 градусов, движения со скоростью 30 градусов в секунду по и против часовой стрелки и последующей остановки оси. В процессе движения наблюдайте за состоянием и параметрами движения оси.

1.5. Поиск начального положения оси

Поскольку шаговый двигатель при отключении питания не обеспечивает сохранение позиции и при этом не имеет обратной связи, перед началом работы после включения питания необходимо определить, где находится нулевая точка оси (начало координат). Для этого применяют концевые датчики различных типов, наиболее распространены механические и индуктивные концевые выключатели (рисунок 47).

Рисунок 47. Механический и индуктивный концевой выключатели

ВНИМАНИЕ! Перед началом подключения концевых выключателей, отключите питание драйвера шагового двигателя.

Начнем реализацию парковки с подключения концевых выключателей к драйверу, поскольку парковку (поиск нуля оси) будем реализовывать при помощи встроенной функции парковки драйвера шагового двигателя. Подключите два индуктивных концевых выключателя по схеме (рисунок 48), один выключатель будет отвечать за негативный предел перемещения (в направлении уменьшения координаты оси), а второй за позитивный (в направлении увеличения координаты оси), помимо парковки данные выключатели обеспечат безопасность оси от выхода за пределы рабочей зоны. По умолчанию позитивный предел назначен на вход SI2, а негативный на вход SI3.

После подключения концевых выключателей включите питание и убедитесь, что они работают, для этого подведите ось к концевому выключателю, на его торце должен загорается светодиод.

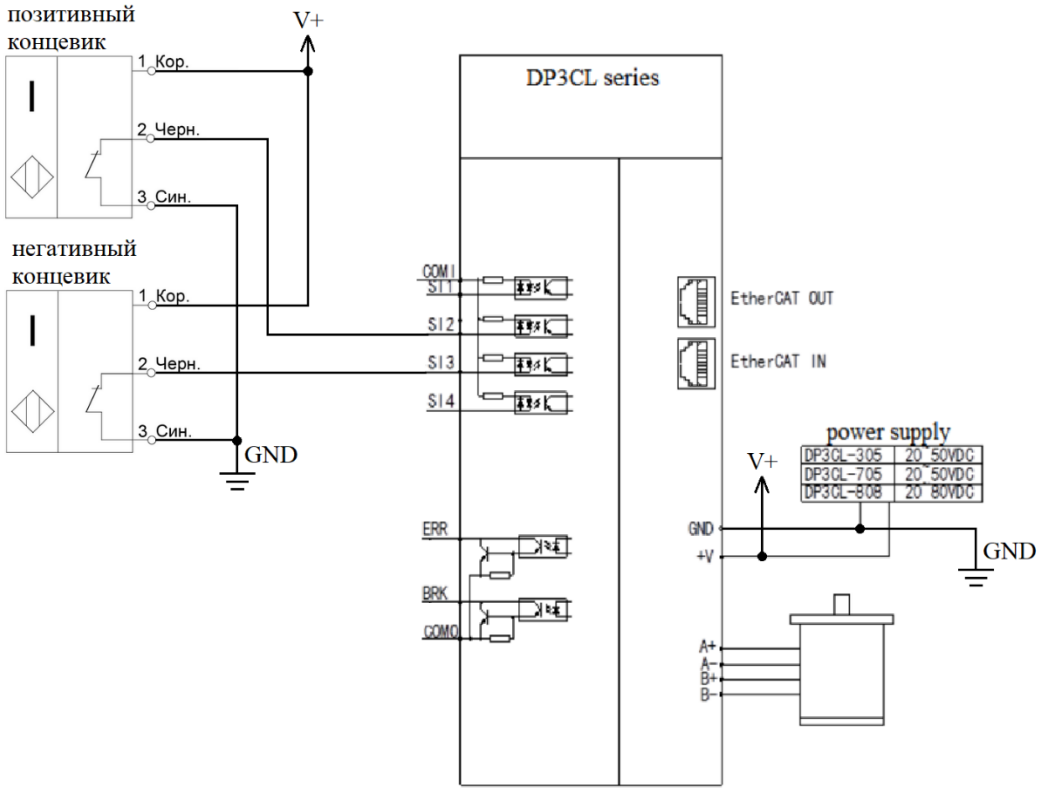


Рисунок 48. Электрическая схема подключения концевых выключателей

В документации драйвера описан алгоритм выполнения парковки, перевод этого алгоритма с пояснениями представлен далее. В квадратных скобках указаны адреса регистров.

1. Установите [режим управления: 6060h] в режим парковки (0x06).
2. Установите [режим парковки: 6098h], диапазон значений: 1~14, 17~30, 33, 34, 35, 37. Некоторые шаговые двигатели не имеют сигнала Z-фазы, поэтому внимательно выбирайте режим парковки. (Рисунок 4, режимы парковки, используемые в данной работе, другие режимы можно изучить в документации)
3. Установите [скорость парковки: 6099h, субиндекс 1] — определяет скорость поиска концевого датчика начала координат (ед. изм.: единиц/с).
4. Установите [скорость парковки: 6099h, субиндекс 2] — определяет скорость выхода на начало координат (ед. изм.: единиц/с).
5. Установите [ускорение парковки: 609Ah] — определяет ускорение при хоминге (ед. изм.: единиц/с²).
6. Установите [управляющее слово: 6040h] в (0x06 → 0x07 → 0x0F), чтобы включить драйвер и запустить двигатель.
7. Установите [управляющее слово: 6040h] в (0x0F → 0x1F), чтобы выполнить поиск концевого датчика и произвести операцию парковки.
8. Считайте [слово состояния: 6041h], чтобы узнать статус драйвера.

В библиотеке SoftMoution есть функциональный блок MC_Home, данный блок выполняет шаги 1, 6-8, автоматически, так же данный блок имеет вход «Position» который отвечает за положение оси которое будет установлено при срабатывании концевого

выключателя (по умолчанию ноль, то есть концевой выключатель находится ровно в точке ноль оси). Однако данный блок не выполняет шаги 2-5, поэтому эти значения необходимо установить вручную.

Обратившись к документации драйвера, рассмотрим назначения регистров, связанных с процедурой парковки оси (рисунок 49). Помимо ранее упомянутых регистров так же рассмотрим регистр 0x60FD, данный регистр содержит состояния концевых выключателей, как показано на карте регистров нулевой бит отвечает за негативный предел, а первый бит за позитивный предел.

Digital input (60FDh)

Index	Subindex	Name	Range	Data type	Accessibility	PDO	Op-mode
60FDh	00h	Digital inputs	0~4294967295	U32	ro	TxPDO	All

Indicates the theoretical input state of the external input signal.

Bit information

31	30	29	28	27	26	25	24
r							
23	22	21	20	19	18	17	16
r							
15	14	13	12	11	10	9	8
r							
7	6	5	4	3	2	1	0
R					hs	pls	nls

r = reserved (Not corresponding) pls= positive limit switch

nls = negative limit switch hs=home switch

Index	Name	Unit	Read/write
6040h	Controlword	-	RW
6041h	Statusword	-	RO
6060h	Modes of operation	-	RW
6061h	Modes of operation display	-	RO
6098h	Homing method	-	RW
6099h	Homing speed	command unit/s	RW
609A	Homing acceleration	command unit/s^2	RW

Рисунок 49. Электрическая схема подключения концевых выключателей

Рассмотрим методы парковки, которые будем использовать, наиболее простые методы парковки, обозначены режимами 17, 18, алгоритм парковки следующий, ось движется с начальной скоростью (регистр 0x6099-1) к концевому выключателю, затем при его срабатывании замедляется с ускорением (регистр 0x609A) и меняет направление движения а так же переходит на медленную скорость (регистр 0x6099-2), после чего в момент съезда с концевого выключателя (заднего фронта его сигнала) происходит обнуление позиции оси. Режимы 17 и 18 отличаются только тем, по какому выключателю будет происходить парковка по отрицательному или положительному. Диаграмма парковки из документации драйвера представлена на рисунке 50.

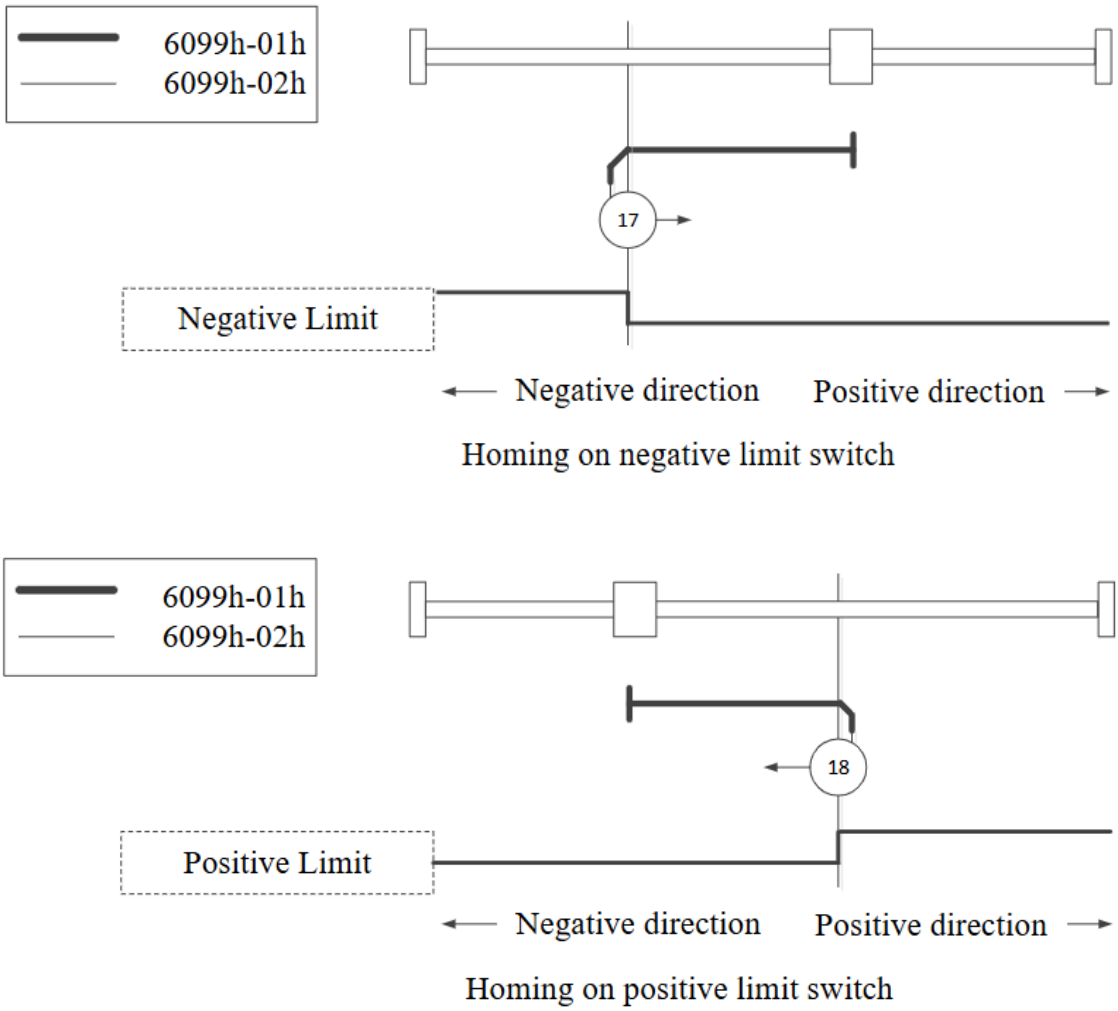


Рисунок 50. Диаграмма парковки драйвера

Для того чтобы записать параметры работы необходимо добавить их в список регистров, которые считываются и записываются через EtherCAT. Для этого откройте проект CODESYS и в нем откройте объект драйвера, в параметрах драйвера на вкладке «Общее» необходимо установить галочку «Экспертные установки», для того чтобы получить доступ к расширенным настройкам драйвера, в частности к настройке соотнесения регистров устройства (рисунок 51).

Далее перейдите в появившуюся вкладку настроек драйвера «Экспертные данные процесса» и в правой верхней четверти окна, выберите «1st RxPDO Mapping» (рисунок 52). Следующим шагом в правой нижней четверти окна, нажмите «Вставить», для того чтобы добавить новый регистр для записи в память драйвера, откроется окно выбора регистра (рисунок 53).

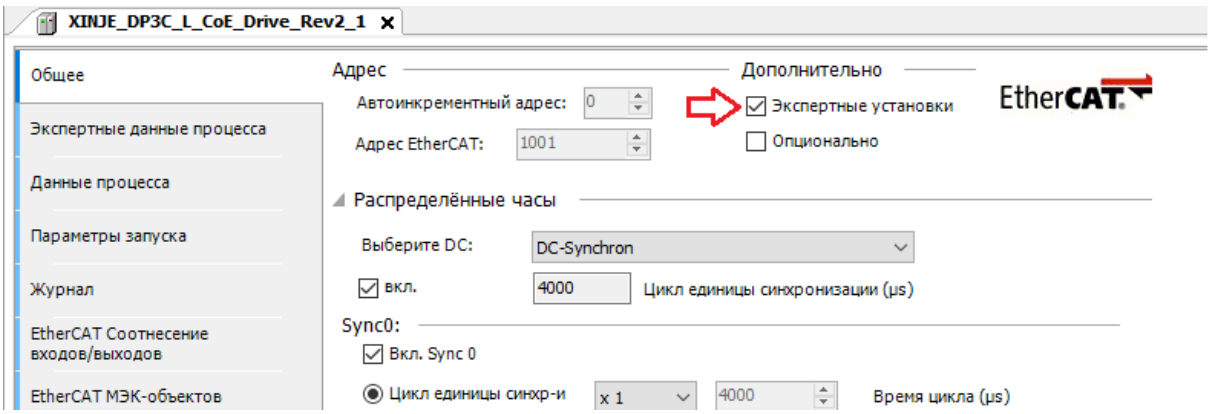


Рисунок 51. Активация режима расширенных настроек

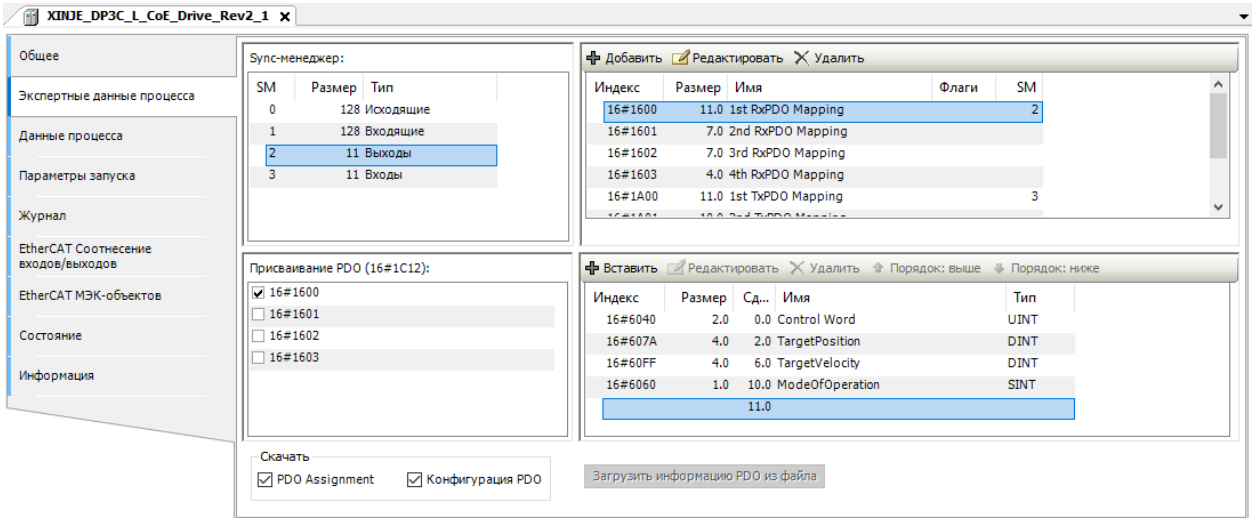


Рисунок 52. Вкладка «Экспертные данные процесса»

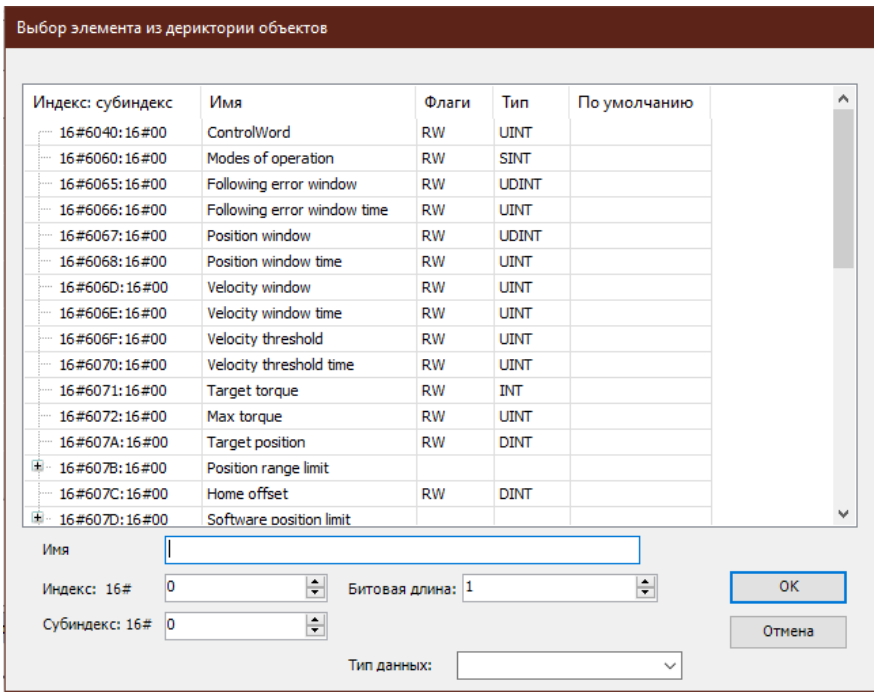


Рисунок 53. Окно выбора регистра

Начнем с добавления регистра отвечающего за выбор режима парковки (0x6068), для его добавления можно либо выбрать его из списка, либо ввести параметры адреса и названия вручную, затем нажмите «ОК» (рисунок 54).

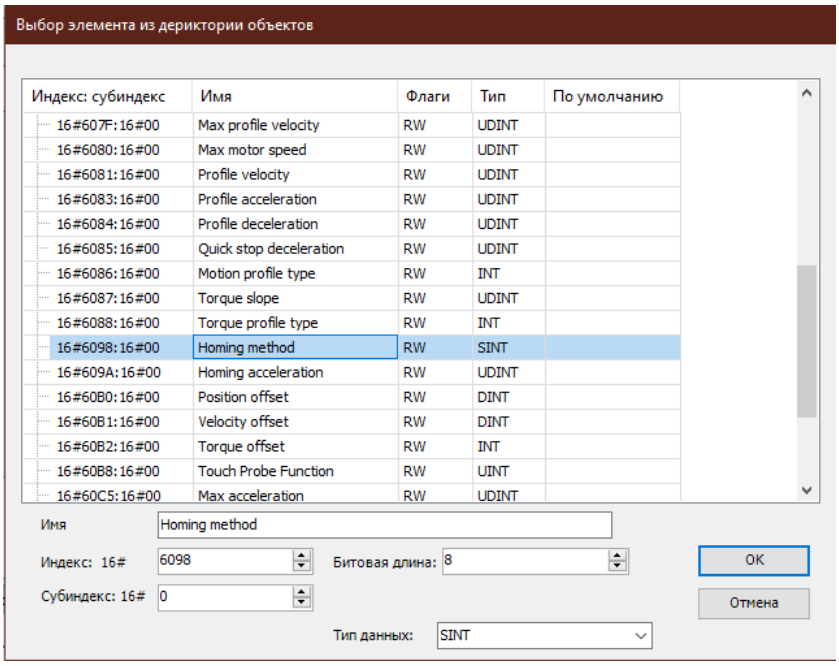


Рисунок 54. Выбор регистра режима парковки

Аналогично добавьте регистры, отвечающие за скорость, ускорение парковки (рис. 55-57).

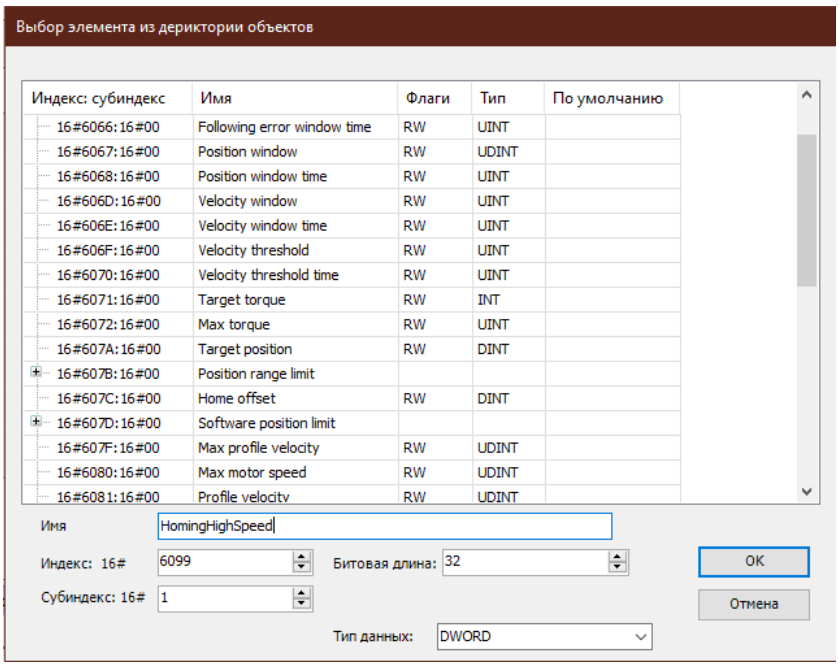


Рисунок 55. Выбор регистра скорости движения к концевому выключателю

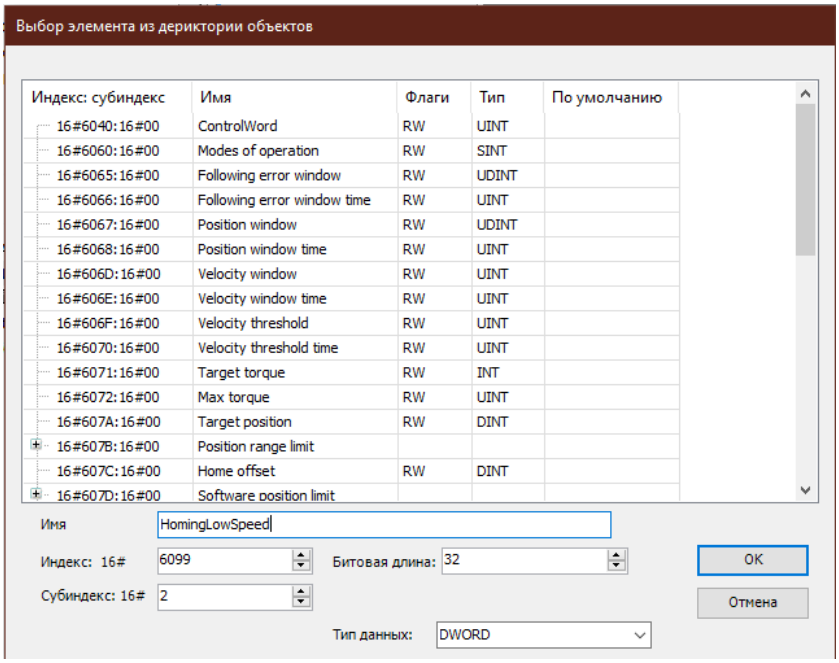


Рисунок 56. Выбор регистра скорости движения от концевого выключателя

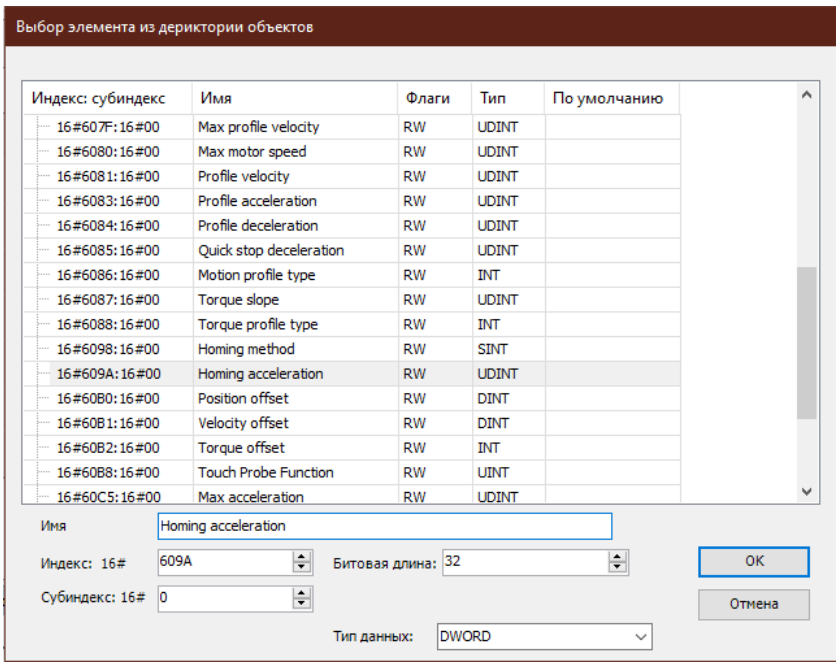


Рисунок 57. Выбор регистра ускорения при парковке

Добавим так же регистр, отвечающий за состояние концевых выключателей (0x60FD), для этого в правой верхней четверти окна, выберите «1st TxPDO Mapping», и по аналогии с предыдущими регистрами добавим его в список (рис. 58-59).

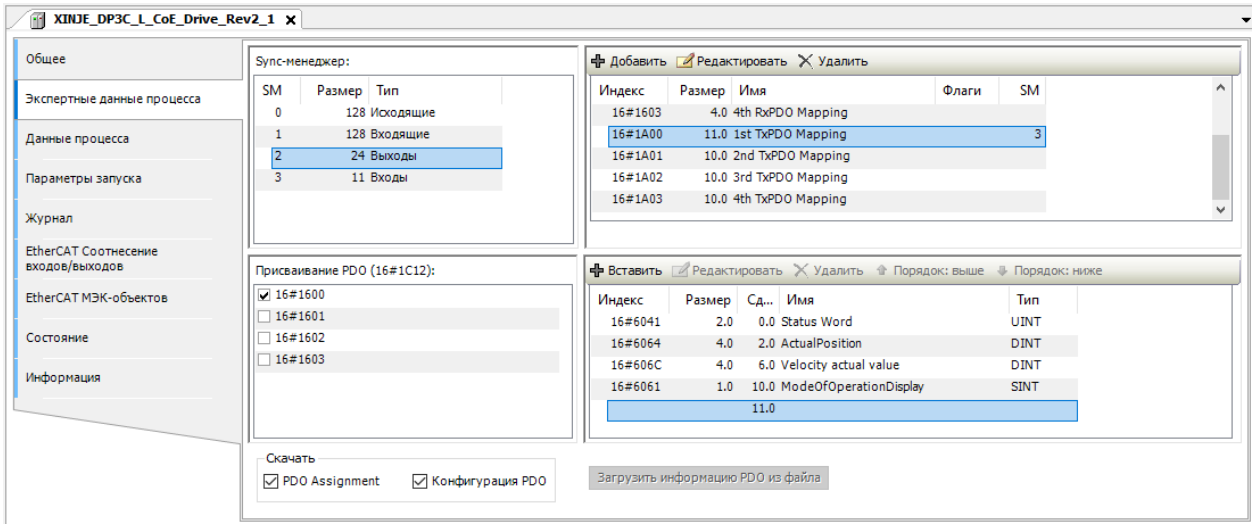


Рисунок 58. Вкладка «Экспертные данные процесса», «1st TxPDO Mapping»

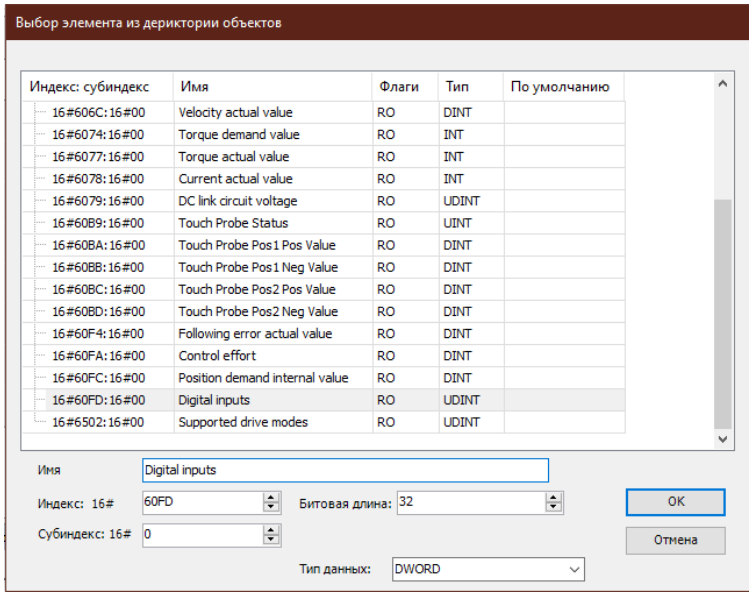


Рисунок 59 Выбор регистра состояния концевых выключателей

Перейдите в PLC_PRG и создайте переменные для хранения значений скорости парковки, ускорения, режима парковки. Поскольку драйвер получает значения в шагах в секунду, необходимо скорость и ускорение перевести из градусов в секунду в шаги в секунду, для этого нужно до множить её на число шагов на градус, то есть отношение числа шагов на оборот к числу градусов в обороте (3200 / 360 указанном в примере) (рис. 60).

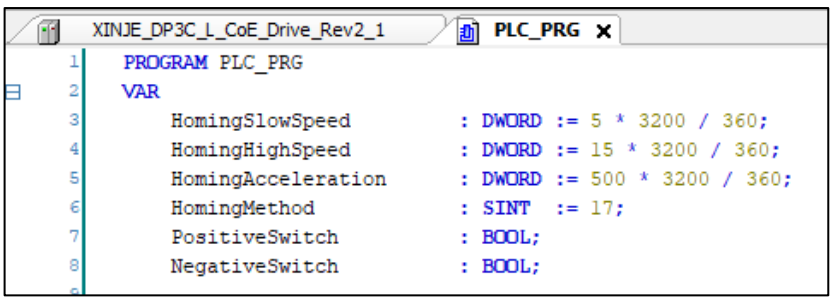


Рисунок 60. Создание переменных для хранения параметров парковки

Следующим шагом необходимо связать переменные с регистрами, для этого откройте настройки драйвера и перейдите на вкладку «EtherCAT Соотнесение входов/выходов» (рис. 61).

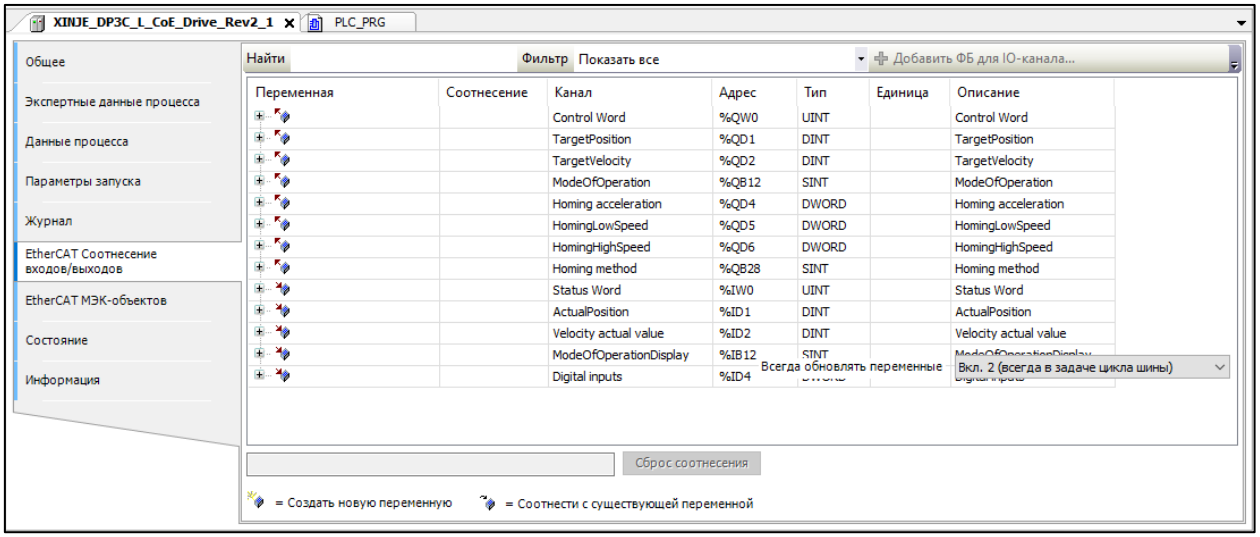


Рисунок 61. EtherCAT Соотнесение входов/выходов

Для того чтобы привязать переменную, дважды клините по полю «Переменная» откройте ассистент ввода нажатием на три точки и с его помощью выберите переменную созданную в «PLC_PRG», повторите эти действия для остальных переменных, созданных ранее (рис. 62-63).

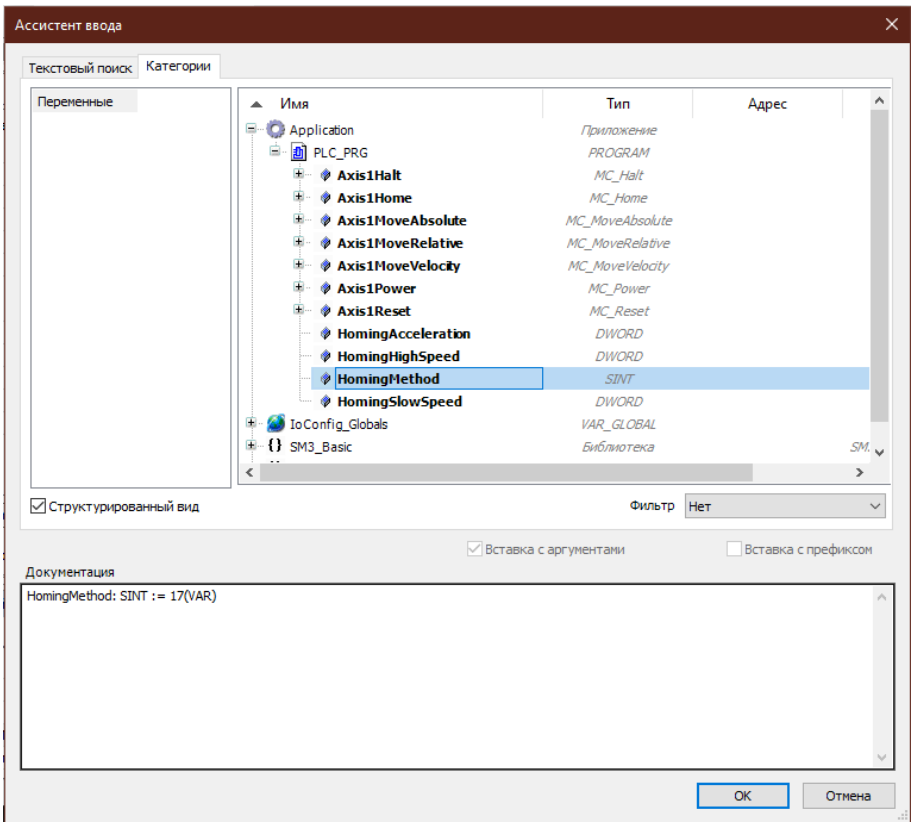


Рисунок 62. Выбор переменной для соотнесения

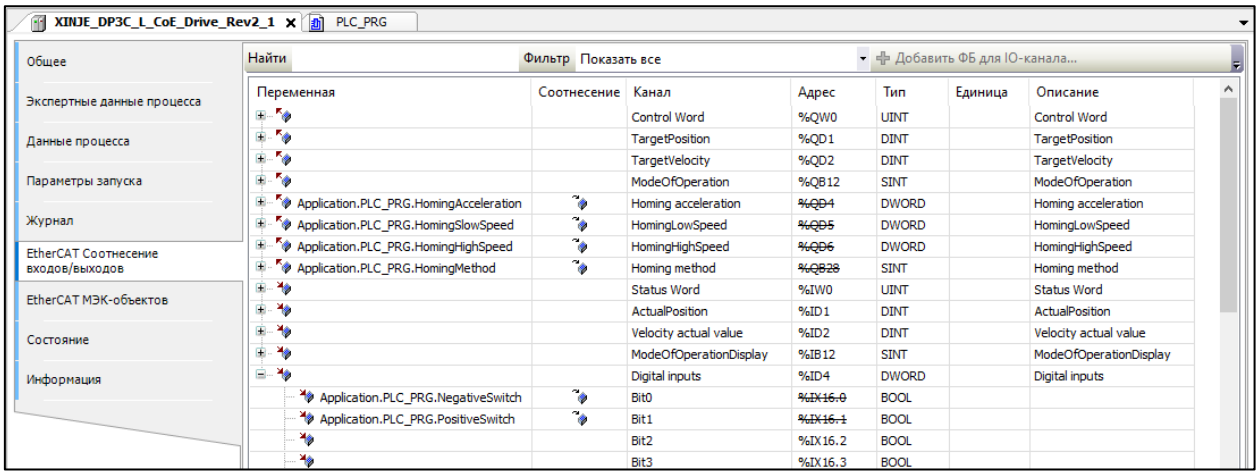


Рисунок 63. Соотнесение переменных

Следующим шагом необходимо добавить блок MC_Home в программу «PLC_PRG», как это было сделано в предыдущей работе с другими блоками (рис. 64).

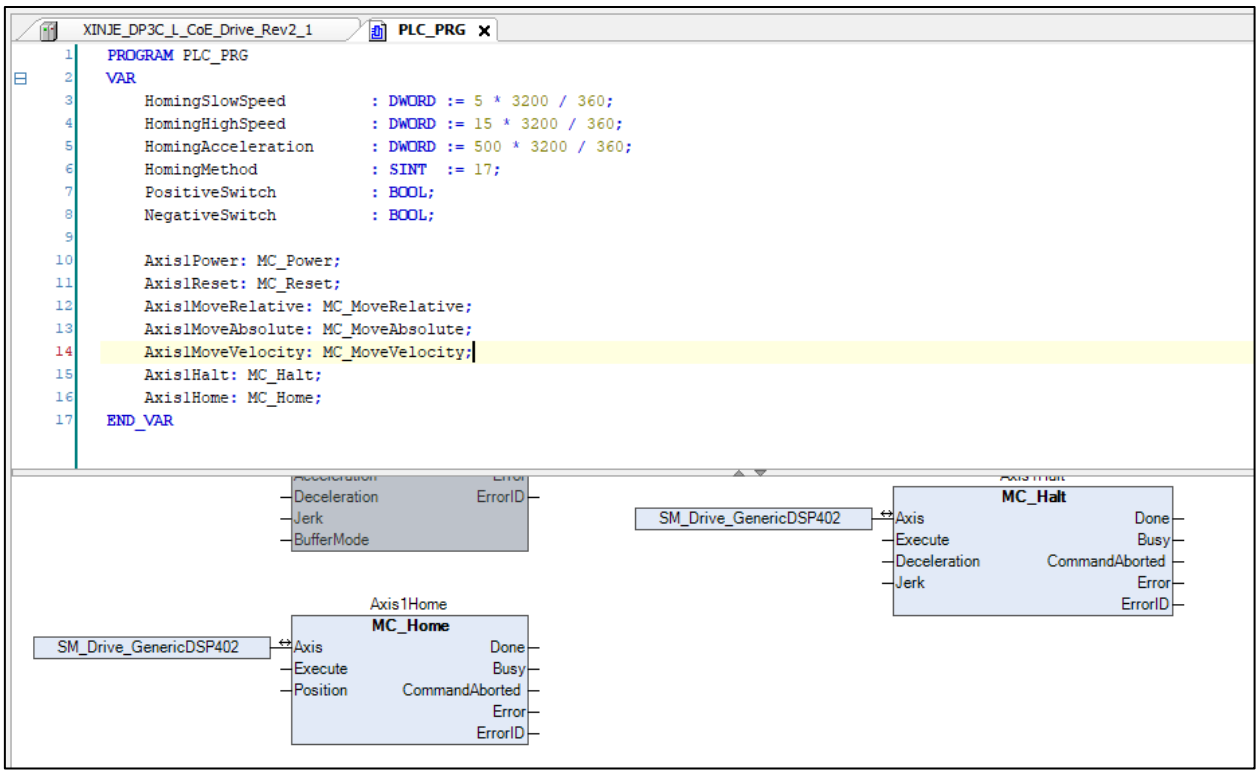


Рисунок 64. Функциональный блок MC_Home

Подключитесь к ПЛК, выполните загрузку программы и запустите контроллер. Проверьте что состояние переменных «PositiveSwitch» и «NegativeSwitch» изменяется при срабатывании концевых выключателей.

При помощи записи значений, как в предыдущем подразделе, запустите парковку при помощи блока MC_Home, предварительно подав питание на двигатель при помощи MC_Power. Ось должна дойти до концевого выключателя, съехать с его и обнулить свое положение.

Изменяя скорость парковки, ускорение и режим в проекте, оцените как эти параметры влияют на процесс парковки. Установите высокую и низкую скорость парковки, попробуйте уменьшить ускорение, проверьте работу режима парковки по позитивному пределу.

1.6. Пользовательский интерфейс

CODESYS имеет возможность создания визуализаций, которые позволяют легко взимодействовать с ПЛК. Для создания визуализации нажмите правой кнопкой мыши по объекту «Application», в выпадающем списке выберите «Добавление объекта» и затем выберите «Взуализация», откроется окно в котором нажмите «ОК» (рисунок 65).

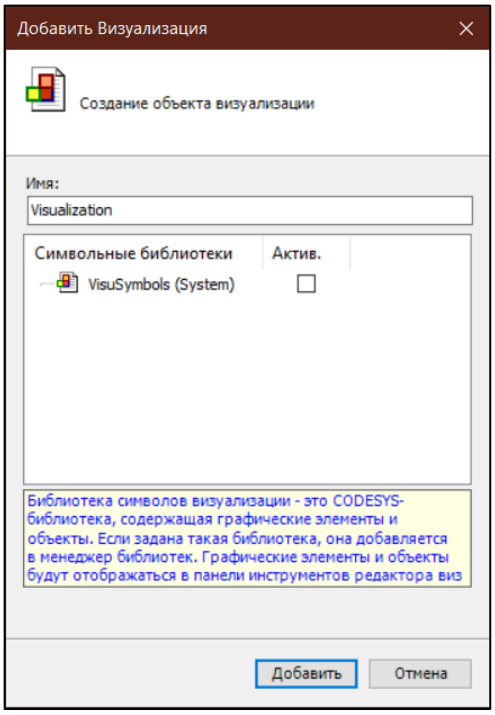


Рисунок 65. Добавление визуализации

Откройте созданную визуализацию. Рассмотрим интерфейс редактирования визуализации (рисунок 66). По центру расположено рабочее поле, на котором располагаются элементы визуализации, справа находится панель инструментов визуализации, на данной панели в нижней части есть вкладки, наиболее используемые вкладки — это «Панель инструментов визуализации», которая содержит объекты визуализации (кнопки, примитивы, и т.п.), вторая вкладка — это «Свойства», она отображает все параметры выделенного на рабочем поле объекта. Для добавления объектов, достаточно просто перетащить их мышью с правой панели на рабочее поле.

Для визуализации, которая позволит управлять шаговым двигателем, необходимо реализовать ввод и вывод числовых параметров и бинарных сигналов. Для ввода и вывода бинарных сигналов, в стандартной библиотеке сигналов есть стандартные элементы, такие как кнопки и индикаторные лампы (рисунок 67), расположенные в разделе «Lamps/Switches/Bitmaps».

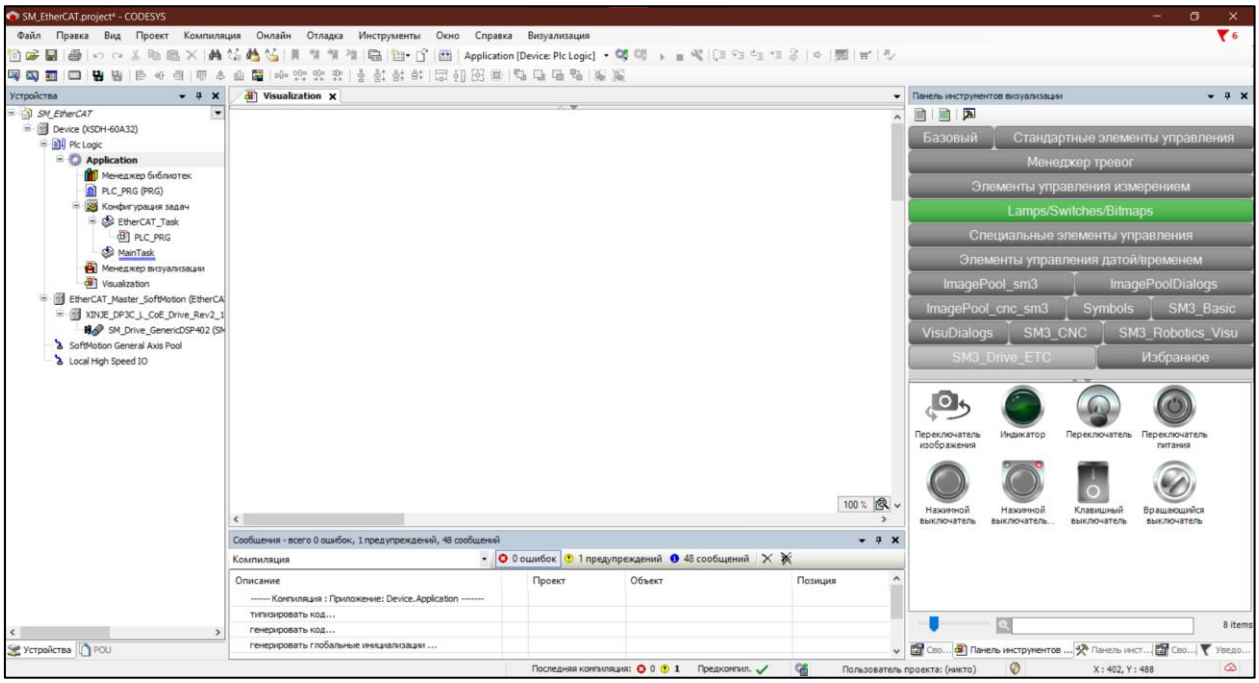


Рисунок 66. Интерфейс редактора визуализации

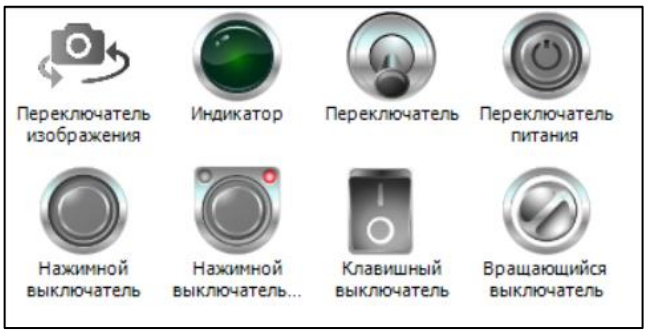


Рисунок 67. Элементы ввода/вывода бинарных сигналов

Для того чтобы получить сигнал с кнопки или передать его в лампу, необходимо выбрать элемент управления и в свойствах необходимо в поле «Переменная» привязать переменную, с которой он будет работать. Например, для индикации процесса парковки оси, можно создать индикатор и привязать к нему выход «Busy» блока MC_Home, а для запуска парковки привязать его вход «Enable» к кнопке (рисунки 68, 69).

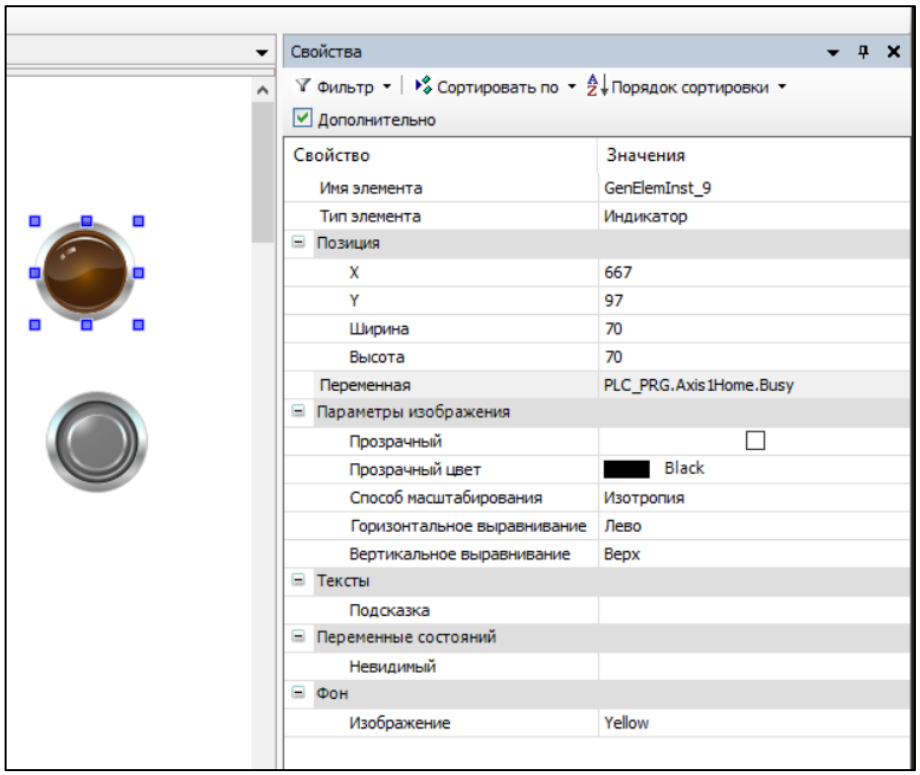


Рисунок 68. Индикатор процесса парковки оси

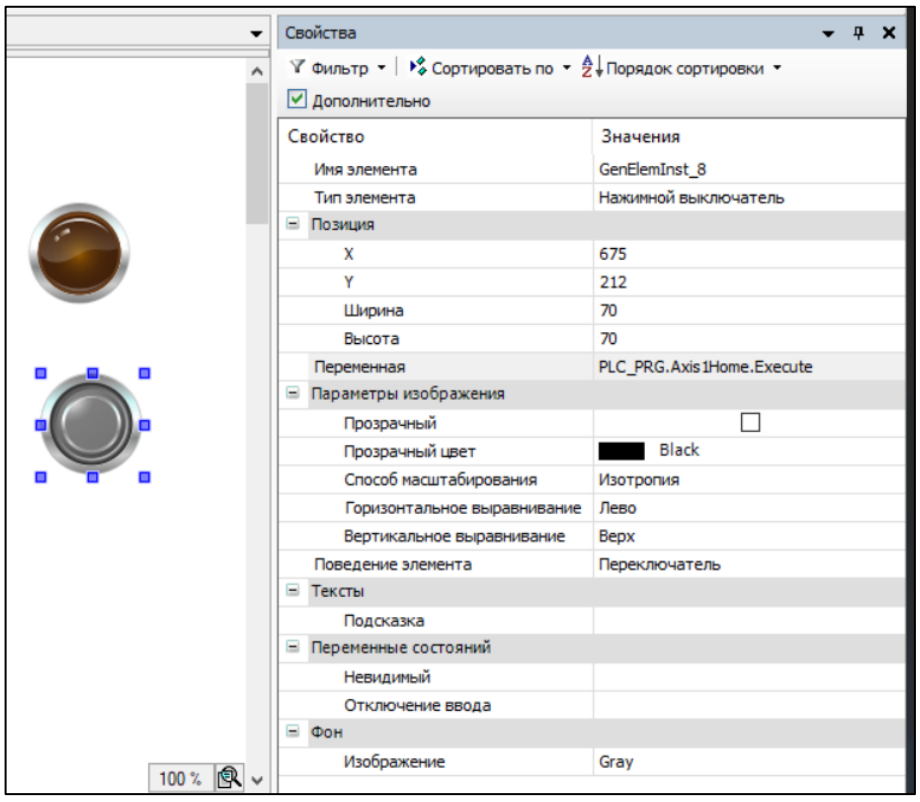


Рисунок 69. Кнопка запуска парковки оси

Для ввода и вывода числовых значений, например скорости перемещения оси, можно использовать либо стандартные элементы ввода вывода такие как «Бегунок», «Индикатор выполнения» и т.п. с вкладки «Стандартные элементы управления» или элементы с вкладки «Элементы управления измерением», принцип работы их

идентичный, необходимо привязать переменную и элемент будет записывать свое значение в неё. Так же у элементов есть настройки дискретности изменения значения и пределов значения.

Например, используя элемент «Управление вращением» реализуем установку скорости движения оси, для этого привяжем переменную и зададим пределы изменения значения скорости (рис. 70).

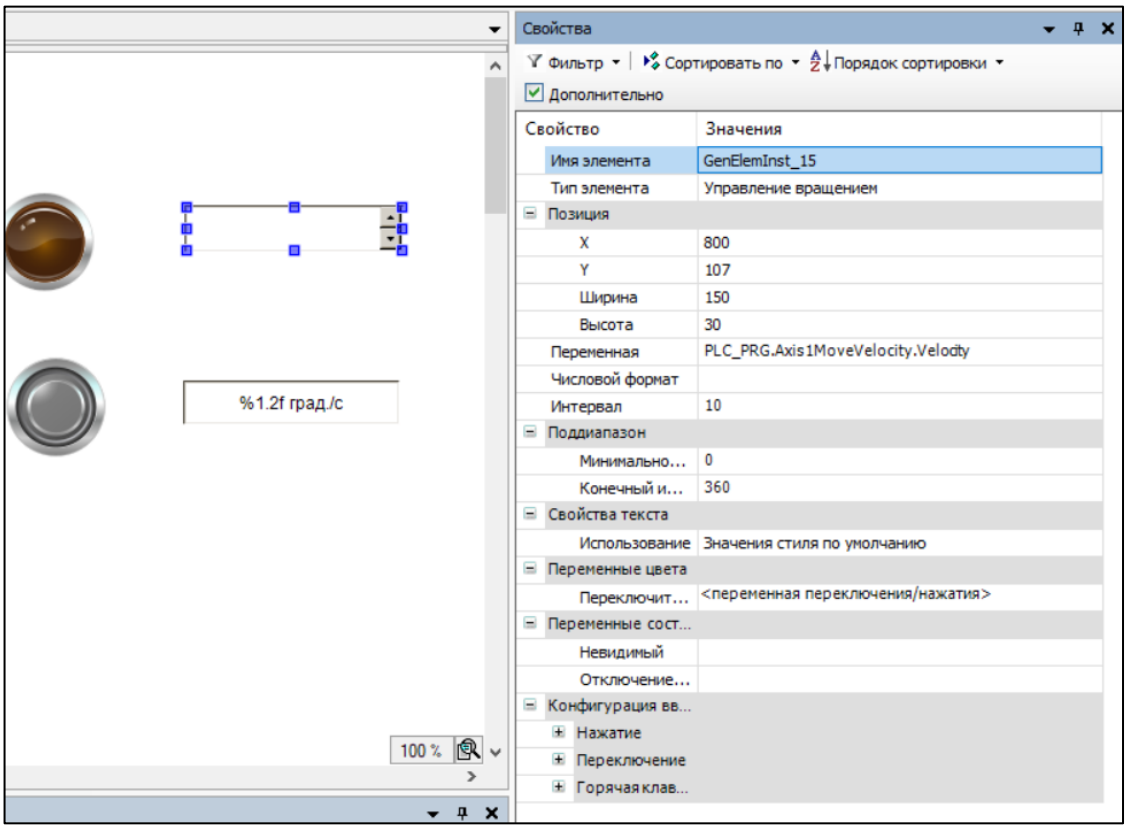


Рисунок 70. Ввод скорости движения оси

Для индикации, например, текущего положения оси, добавим элемент «Текстовое поле», так же привяжем переменную и в поле «Текст» запишем строку форматирования аналогично строкам форматирования языка С (рис. 71-72).

Таблица 1. Спецификаторы форматирования С-строк

Спецификатор	**Тип данных**	**Вывод**
%d или %i	int	десятичное целое со знаком
%u	unsigned int	десятичное без знака
%o	unsigned int	восьмеричное
%x / %X	unsigned int	шестнадцатеричное (нижний/верхний регистр)
%f	double / float	число с плавающей точкой (десятичная запись)
%F	double	то же, но inf/nan в верхнем регистре
%e / %E	double	научная нотация (1.2e+3)
%g / %G	double	автоматически %f или %e (короче)
%c	char	один символ
%s	char*	строка (до \0)

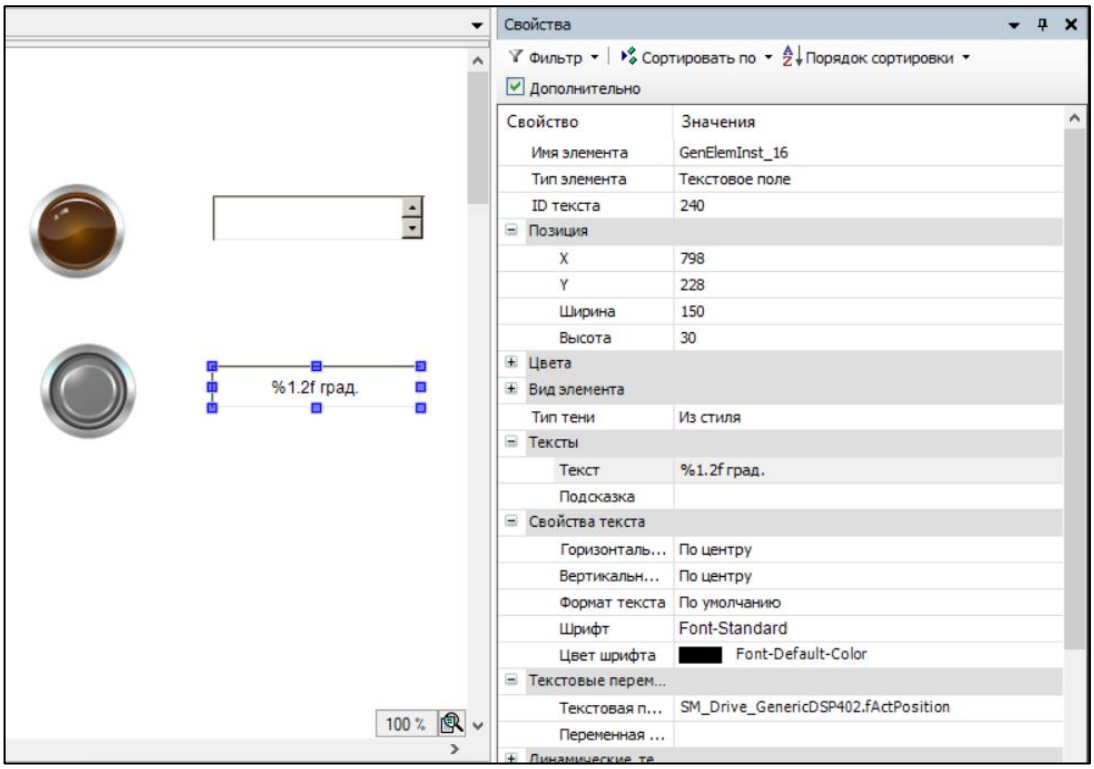


Рисунок 71. Вывод текущего положения оси

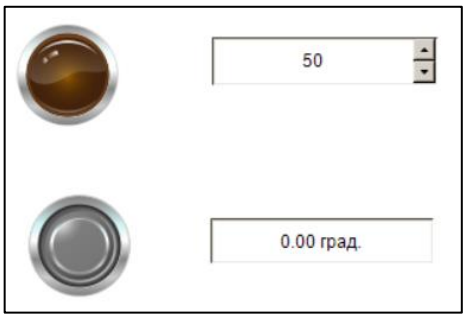


Рисунок 72. Отображение числовых значений

В случае если необходимо использовать одно значение, для нескольких блоков, например скорость для всех трёх блоков движения, необходимо создать переменную в программе «PLC_PRG», связать её с элементом визуализации, а затем передать на вход блока аналогично ссылке на ось.

Используя описанные элементы визуализации, выполните интерфейс, который позволяет перемещать ось в относительном, абсолютном режиме, запускать движение с заданной скоростью и останавливать движение, управлять питанием и сбрасывать ошибку, а также отображающий состояние концевых выключателей.

2. ПРАКТИЧЕСКИЕ ЗАДАНИЯ И ВОПРОСЫ ДЛЯ САМОКОНТРОЛЯ

Практическая работа №1. Создание и настройка проекта в CODESYS

Цель: научиться запускать среду CODESYS, создавать новый стандартный проект, выбирать целевое устройство и язык программирования, устанавливать соединение с ПЛК и загружать программу.

Задания:

1. Запустите среду CODESYS через меню «Пуск» или с помощью ярлыка на рабочем столе. Дождитесь полной загрузки главного окна программы.
2. Создайте новый стандартный проект, используя кнопку чистого листа на панели инструментов или сочетание клавиш Ctrl + N. Задайте произвольное расположение и имя проекта (на английском языке, без пробелов и спецсимволов).
3. В окне выбора устройства укажите модель XSDH-60A32, а в качестве языка для программного компонента PLC_PRG выберите CFC.
4. Дважды кликните по элементу PLC_PRG в дереве проекта. Изучите структуру главного окна.
5. Если в консоли появились ошибки об отсутствии библиотек, нажмите на многоточие во втором столбце, выберите пункт «загрузить библиотеки» и нажмите кнопку «Загрузка». Дождитесь исчезновения ошибок.
6. Дважды кликните по объекту «Device (XSDH-60A32)» в дереве проекта и перейдите на вкладку «Установка соединения». Нажмите кнопку «Сканировать сеть», в открывшемся окне повторно нажмите «Сканировать сеть».
7. Убедитесь, что ПЛК отобразился в списке найденных устройств. Выберите его и закройте окно. На главной панели инструментов выберите «Логин» и подтвердите загрузку программы в диалоговом окне.

Вопросы для самоконтроля:

1. Для чего предназначен ярлык в виде чистого листа на панели инструментов и какое сочетание клавиш его дублирует?
2. Перечислите основные элементы главного окна CODESYS после создания проекта. Где именно отображаются программы и настроенные устройства?
3. Что необходимо сделать, если в консоли вывода отображаются ошибки, связанные с отсутствием библиотек?
4. Какое физическое гнездо на ПЛК следует использовать для соединения с компьютером?
5. Опишите последовательность действий для обнаружения ПЛК в сети.
6. Почему важно настроить IP-адрес компьютера в той же подсети, что и у контроллера? Какой IP-адрес по умолчанию у ПЛК XINJE XSDH-60A32?

Практическая работа №2. Конфигурирование драйвера шагового двигателя

Цель: изучить принцип работы и устройство шагового двигателя и методы управления обмотками, научиться подключать и настраивать драйвер шагового двигателя XINJE DP3CL-705.

Задания:

1. Изучите устройство типичного шагового двигателя. Определите, каким образом в биполярном двигателе изменяется направление магнитного потока в катушке, и чем принципиально отличается схема подключения обмоток униполярного двигателя от биполярного.
2. Подключите шаговый двигатель, драйвер и источник питания согласно схеме из официальной документации. Убедитесь, что на регулируемом блоке питания установлено напряжение 24 В и ограничение тока 3 А, после чего отключите блок питания и только затем выполняйте физическое подключение драйвера.
3. Подключите драйвер к ПЛК с помощью интернет-кабеля, используя гнездо Ecat на ПЛК и гнездо ECAT IN на драйвере. Убедитесь в правильности соединения.
4. Соблюдая строгую последовательность (убедиться в отсутствии питания, подключить переходник MiniUSB->COM к преобразователю COM->USB, подключить MiniUSB к драйверу, подключить USB к компьютеру, включить питание драйвера), подготовьте драйвер к соединению с утилитой StepDriver.
5. Запустите утилиту StepDriver и в открывшемся окне выполните автоматический поиск драйвера кнопкой «Auto search». Убедитесь в успешном соединении.
6. В таблице параметров найдите и измените параметры драйера: P4-00 (Peack current) – установите значение 12 (что соответствует рабочему току 1.2 А); P0-08 (Open loop holding current) – установите 30%; P0-01 (Subdivision) – установите значение 3200 (микрошаг 1/16).
7. Запишите измененные параметры в память драйвера, нажав кнопку «Write» и подтвердив запись в появившемся окне. Убедитесь в появлении уведомления об успешной записи.
8. Выполните безопасное отключение драйвера от компьютера в обратной последовательности: выключите питание драйвера, отключите USB от компьютера, отключите MiniUSB от драйвера и отсоедините переходники друг от друга.

Вопросы для самоконтроля:

1. Почему шаговый двигатель относят к синхронным бесщеточным машинам? За счет чего создается магнитное поле ротора?
2. Что такое полушаговый и микрошаговый режим (half step mode) и какие преимущества они дают по сравнению с полношаговым режимом?
3. Объясните принцип микрошагового режима. Каким образом ток в фазах влияет на положение равновесия ротора?
4. Почему для настройки драйвера важно различать рабочий (RMS) и пиковый ток двигателя? Какой из них задается на драйвере?
5. Для чего нужен параметр «ток удержания»?
6. Почему при подключении драйвера по интерфейсу RS-232 необходимо строго соблюдать порядок подачи питания и коммутации разъемов?

Практическая работа №3. Подключение драйвера по EtherCAT и настройка оси

Цель: изучить принцип работы протокола EtherCAT, освоить добавление устройства EtherCAT Master в проект CODESYS, научиться выполнять поиск и конфигурирование подчиненных устройств, добавлять и настраивать ось SoftMotion.

Задания:

1. Откройте проект, созданный ранее. Добавьте в дерево проекта устройство EtherCAT master SoftMotion: кликните правой кнопкой мыши по объекту ПЛК (Device), выберите «Добавить устройство», перейдите в раздел Промышленные сети → EtherCAT → Мастер → EtherCAT master SoftMotion и нажмите «Добавить устройство».
2. Дважды кликните по добавленному устройству, перейдите в раздел «Общее» и назначьте MAC-адрес источника. Для этого нажмите кнопку «Обзор» напротив пункта «Адрес источника», выберите интерфейс Eth1 и подтвердите выбор.
3. Отключите питание драйвера шагового двигателя, включите питание ПЛК, дождитесь загрузки ПЛК, затем подайте питание на драйвер. Выполните подключение к контроллеру (Онлайн → Логин) и согласитесь с загрузкой программы.
4. Находясь в режиме онлайн, нажмите правой кнопкой мыши на объект EtherCAT_master в дереве проекта и выберите пункт «Поиск устройств». В открывшемся окне выделите обнаруженный драйвер и нажмите кнопку «Копировать в проект». Убедитесь, что объект драйвера появился в дереве проекта, после чего выйдите из режима онлайн в режим редактирования.
5. Добавьте ось SoftMotion: кликните правой кнопкой мыши на объект драйвера в дереве проекта и выберите пункт «Add SoftMotion CiA402 Axis».
6. Откройте двойным щелчком добавленную ось и перейдите на вкладку «General». Установите динамические ограничения: Velocity — 360 градусов в секунду, Acceleration и Deceleration — 500 градусов в секунду в квадрате, Velocity ramp type — Trapezoid. Сохраните настройки.
7. Перейдите на вкладку «Scaling/Mapping» и настройте масштабирование: в разделе «Scaling» укажите, что на один оборот двигателя приходится 3200 шагов (учитывая микрошаг 1/16 и 200 физических шагов на оборот) и 360 градусов на оборот.
8. В файле PLC_PRG добавьте в область программы блок MC_Power через ассистент ввода. Задайте имя экземпляра блока (например, Axis1Power). Добавьте «Ввод» и через ассистент ввода выберите созданную ранее ось, после чего соедините этот ввод с входом Axis блока MC_Power.
9. Подключитесь к ПЛК и запустите программу нажатием на треугольник рядом с кнопкой подключения к контроллеру. Дождитесь успешной инициализации драйвера и оси (появления зелёных значков синхронизации). Раскройте Axis1Power в области переменных PLC_PRG, установите подготовленные значения, нажмите Ctrl+F7 для записи и убедитесь, что вал двигателя зафиксировался в режиме удержания.

Практическая работа №4. Управление перемещением оси с помощью функциональных блоков SoftMotion

Цель: изучить назначение функциональные блоки библиотеки SoftMotion, освоить способы управления движением оси (относительное, абсолютное, с заданной скоростью), научиться отслеживать состояние оси в режиме онлайн, запускать и останавливать движение, сбрасывать ошибки.

Задания:

1. Перенесите исполнение программы PLC_PRG из задачи MainTask в задачу EtherCAT_Task.
2. Добавьте в программу PLC_PRG функциональные блоки: MC_Reset, MC_MoveRelative, MC_MoveAbsolute, MC_MoveVelocity, MC_Halt. Для каждого блока задайте имя экземпляра (например, Axis1Reset, Axis1MoveRelative и т.д.). На вход Axis всех добавленных блоков подайте ссылку на ранее созданную ось.
3. Изучите общие для всех блоков входы (Axis, Enable) и выходы (Done, Busy, Error, ErrorID).
4. Подключитесь к контроллеру и загрузите программу. Дважды кликните по оси в дереве проекта в режиме онлайн и откройте вкладку «General». Найдите блок «Online» и ознакомьтесь с отображаемыми параметрами: текущее положение оси, параметры движения, состояние конечного автомата, активные ошибки.
5. Используя алгоритм запуска выполнения блока (установка Enable = «Ложь» → запись Ctrl+F7 → установка Enable = «Истина» → запись Ctrl+F7), выполните следующие действия с осью:

- Включите питание оси через блок MC_Power, предварительно убедившись, что на входах bRegulatorOn и bDriveStart установлено значение «Истина».
- Выполните относительное перемещение на 10 градусов по часовой стрелке и на 10 градусов против часовой стрелки с помощью блока MC_MoveRelative, изменяя знак параметра Distance.
- Выполните абсолютное перемещение в нулевую координату (Position = 0), а затем в координату 360 градусов с помощью блока MC_MoveAbsolute.
- Запустите движение с постоянной скоростью 30 градусов в секунду в положительном направлении с помощью блока MC_MoveVelocity, затем остановите ось с помощью блока MC_Halt. Повторите то же самое для отрицательного направления.

6. В процессе выполнения каждого движения наблюдайте за изменением состояния и параметров движения оси в окне «Online» на вкладке «General». Сравните поведение выходов Done, Busy и Error до, во время и после завершения движения.

Вопросы для самоконтроля:

1. Перечислите общие входы и выходы всех функциональных блоков управления движением библиотеки SoftMotion. Для чего предназначен каждый из них?
2. Почему блоки управления движением должны вызываться только в задаче, связанной с мастером EtherCAT?
3. Опишите назначение входов bRegulatorOn и bDriveStart блока MC_Power. Какую функцию выполняет каждый из них?

4. Чем отличается работа блока MC_MoveRelative от MC_MoveAbsolute? Приведите пример: в какой итоговой координате окажется ось, находящаяся в позиции 50, при выполнении MC_MoveRelative, Distance = 25 и MC_MoveAbsolute, Position = 25 соответственно.
5. В чем разница между блоками MC_Stop и MC_Halt? Какой из них имеет более высокий приоритет и не может быть прерван другим блоком движения?
6. Что произойдет, если перед запуском блока движения не задать значения скорости, ускорения и рывка на его входах?

Практическая работа №5. Настройка и выполнение процедуры парковки (поиска нуля оси) шагового двигателя

Цель: научиться конфигурировать драйвер шагового двигателя для выполнения парковки оси, освоить ручное редактирование опрашиваемых регистров EtherCAT.

Задания:

1. Изучите электрическую схему, выполните подключение концевых выключателей к драйверу шагового двигателя.
2. Используя описание из теоретической части работы, добавьте регистры отвечающие ща парковку оси в список регистров опрашиваемых при помощи EtherCAT.
3. Создайте переменные параметров парковки, свяжите их с регистрами и задайте для них значения как описано в теоретической части работы.
4. Добавьте блок MC_Home.
5. Подключитесь к контроллеру и выполните парковку оси, изменяя параметры ускорения, скорости парковки и режима, проанализируйте их влияние на процесс парковки оси.

Вопросы для самоконтроля:

1. Почему перед началом подключения концевых выключателей необходимо обязательно отключать питание драйвера?
2. В каких единицах измерения задаются скорость и ускорение парковки в регистрах драйвера? Как перевести значения в эту величину?
3. Какие шаги алгоритма парковки функциональный блок MC_Home выполняет автоматически, а какие требуют ручной установки через регистры?
4. Что происходит в момент схода оси с концевого выключателя (задний фронт сигнала) при использовании режимов 17 и 18?
5. Каким образом в CODESYS получить доступ к расширенным настройкам драйвера для добавления регистров в PDO-маппинг?
6. В чём разница между RxPDO Mapping и TxPDO Mapping при добавлении регистров? Приведите пример регистра для каждого типа маппинга из данной работы.

Практическая работа №6. Разработка визуализации для управления шаговым двигателем в CODESYS

Цель: изучить интерфейс редактора визуализации CODESYS, освоить создание элементов ввода/вывода бинарных сигналов и числовых значений, научиться связывать элементы визуализации с переменными программы ПЛК.

Задания:

1. Изучите интерфейс редактора визуализации.
2. Используя описанные элементы визуализации, выполните интерфейс, который позволяет:

- перемещать ось в относительном,
- перемещать ось в абсолютном режиме,
- запускать движение с заданной скоростью,
- останавливать движение,
- управлять питанием двигателя,
- сбрасывать ошибку оси,
- просматривать состояние концевых выключателей.

3. Протестируйте и отладьте работу интерфейса. Оформите интерфейс так чтобы он был читаемым и аккуратным.

Вопросы для самоконтроля:

1. Каким способом элементы визуализации добавляются на рабочее поле редактора?
2. В какой вкладке правой панели отображаются все параметры выделенного объекта визуализации?
3. Какой спецификатор форматирования (%d, %f, %s) необходимо использовать для вывода целого числа, а какой для числа с плавающей точкой в текстовом поле?
4. Зачем выполняется привязка переменной к элементу ввода или вывода?
5. Какими параметрами необходимо управлять из визуализации для полноценного управления шаговым двигателем?

Заключение

В ходе выполнения практических работ, описанных в методических указаниях, обучающийся последовательно прошёл все основные этапы создания проекта управления движением в среде CODESYS: от инициализации проекта и настройки связи с ПЛК, и работы с протоколом EtherCAT до построения пользовательского интерфейса с возможностью управления шаговым двигателем и выполнения процедуры парковки.

В результате освоения материала были сформированы следующие практические навыки и компетенции:

1. Создание и конфигурирование проекта – разработка стандартного проекта для целевого устройства XSDH-60A32 на языке CFC; подключение к ПЛК по промышленному протоколу Ethernet; диагностика и устранение ошибок на этапе загрузки библиотек; настройка сетевого взаимодействия между средой разработки и контроллером.
2. Настройка и интеграция исполнительных устройств – понимание принципов работы и устройства шагового двигателя, конфигурирование параметров драйвера XINJE DP3CL-705 через специализированную утилиту StepDriver.
3. Работа с промышленным протоколом EtherCAT – понимание архитектуры и механизма обработки данных, добавление и настройка EtherCAT Master SoftMotion; автоматический поиск ведомых устройств в сети и их копирование в проект; конфигурирование оси SoftMotion.
4. Программирование логики управления движением – применение функциональных блоков библиотеки SoftMotion освоение алгоритмов запуска/останова перемещения.
5. Настройка опроса регистров EthercCAT – добавление и привязка специфических регистров, реализация процедуры поиска начального положения с анализом влияния параметров на поведение оси.
6. Разработка человеко-машинного интерфейса (HMI) средствами CODESYS – создание и конфигурирование визуализации, использование стандартных элементов ввода/вывода (кнопки, индикаторы, бегунки, текстовые поля), привязка переменных к графическим элементам для управления.

Полученные знания являются фундаментом для дальнейшего изучения более сложных аспектов промышленной автоматизации на базе CODESYS: реализации многокоординатного синхронного движения, подключения и конфигурирования сервоприводов по стандарту CiA402, организации сложных пользовательских интерфейсов с анимацией и разграничением прав доступа, а также использования дополнительных промышленных протоколов в единой системе управления.

Рекомендуемая литература

1. EtherCAT Technology Group [Электронный ресурс] // сайт www.ethercat.org URL: https://www.ethercat.org/default.htm (дата обращения 20.05.2026).
2. Сапожников А. От классической полевой шины (fieldbus) к EtherCAT // Современные технологии автоматизации. 2010. №3. С. 16-18. URL: https://www.cta.ru/cms/f/434755.pdf (дата обращения 20.05.2026).
3. Обзор протокола EtherCAT и устройств на его основе [Электронный ресурс] // сайт ipc2u.ru URL: https://ipc2u.ru/articles/prostye-resheniya/obzor-protokola-ethercat-i-ustroystv-na-ego-osnove/ (дата обращения 20.05.2026).
4. XSDH-60A32 hardware manual [Электронный ресурс] // сайт xinje.com URL: https://xinje.com.ru/XS_series_PLCopen_controller_hardware_manual_2024ver.pdf (дата обращения 20.05.2026).
5. XSDH-60A32 software manual [Электронный ресурс] // сайт xinje.com URL: https://cdn.xinje.com.ru/files/1/4272/38449328/original/XS_software_manual.pdf (дата обращения 20.05.2026).
6. XS series moution control manual [Электронный ресурс] // сайт xinje.com URL: https://cdn.xinje.com.ru/files/1/4275/38449331/original/XS_motion_control_manual_.pdf (дата обращения 20.05.2026).
7. CODESYS SoftMotion [Электронный ресурс] // сайт helpme-codesys.com URL: https://content.helpme-codesys.com/en/CODESYS%20SoftMotion/ _sm_start_page.html (дата обращения 20.05.2026).
8. CODESYS V3.5 первый старт [Электронный ресурс] // сайт owen.ru URL: https://ftp.owen.ru/CoDeSys3/11_Documentation/03_3.5.11.5/CDSv3.5 _FirstStart_v3.0.pdf (дата обращения 20.05.2026).
9. Шпаргалка по CODESYS V3.5 [Электронный ресурс] // сайт owen.ru URL: https://ftp.owen.ru/CoDeSys3/99_ForumFiles/CodesysCheatSheet.pdf (дата обращения 20.05.2026).
10. DP3CL-705 user manual [Электронный ресурс] // сайт xinje.com URL: https://cdn.xinje.com.ru/files/1/4435/38449491/original/DP3CL-open-bus-step-driver-manual.pdf (дата обращения 20.05.2026).
11. DP3CL-705 configuration utility [Электронный ресурс] // сайт xinje.com URL: https://xinje.com.ru/product/dp3cl-705 (дата обращения 20.05.2026).