# История и версии

**История CANopen** берёт начало в 1993 году, когда в рамках европейского проекта Esprit ASPIC под руководством компании Bosch началась разработка надстройки над протоколом CAN — CANopen. Целью было создать профиль связи для сетей производственных ячеек. В 1995 году организация CiA (CAN in Automation) выпустила переработанную спецификацию CANopen, определившую как структуру обмена данными, так и профили устройств и интерфейсов. Благодаря своей гибкости и адаптивности CANopen быстро стал стандартом в таких отраслях, как машиностроение, медицина и морские системы, обеспечивая эффективное и надёжное взаимодействие между встраиваемыми устройствами.

Первая спецификация CANopen была представлена организацией **CiA (CAN in Automation)** в **1995 году**. Эта версия стала основой для промышленного внедрения — в том числе в медицинских, морских, транспортных и машиностроительных системах.

Далее выходили новые редакции, в которых **уточнялись форматы сообщений**, **добавлялись новые профили устройств** (например, для датчиков, приводов, панелей управления), и улучшалась поддержка безопасности и диагностики.

Одним из ключевых обновлений стало введение стандарта **EN 50325-4**. Это европейский стандарт, официально утверждающий CANopen как промышленный протокол. Версия CANopen, соответствующая этому стандарту, обеспечивает полную совместимость между устройствами от разных производителей.

Сегодня CANopen выпускается и поддерживается как **набор профилей (CiA 301, 302, 401 и др.)**, каждый из которых отвечает за определённый аспект:

* **CiA 301** — основная спецификация протокола.
* **CiA 302** — управление сетью (например, загрузка конфигурации).
* **CiA 402** — профиль для систем управления приводами.
* **CiA 305** — CANopen по CANopen over Ethernet (поддержка туннелирования через TCP/IP).
* И так далее.

Наконец, с учётом появления расширенного формата передачи данных **CAN FD**, была разработана **новая версия: CANopen FD**, в которой обновлён механизм обмена сообщениями и увеличены размеры данных. Но классический CANopen остаётся актуальным — он надёжен, проверен временем и поддерживается огромным числом устройств.

# Скорость передачи данных

**Скорость передачи в классическом CANopen** задаётся физическим уровнем CAN (ISO 11898‑2) и напрямую влияет на длину шины, число узлов и циклическую производительность приложений.

**Ключевой вывод:** скорость CANopen влияет не только на пропускную способность, но и на надёжность связи и максимальную длину сети. Подбор битрейта — компромисс между быстродействием и физическими ограничениями шины.

# Описание форматов и алгоритмов передачи данных

В CANopen передача данных организована поверх базового протокола CAN и строго стандартизирована. Система обмена построена по модели "производитель–потребитель", где узлы сети взаимодействуют с использованием чётко определённых типов сообщений, называемых **объектами связи**.

Основу передачи данных в CANopen составляют так называемые **CAN-кадры**, каждый из которых содержит:

* **Идентификатор** (определяет приоритет и тип сообщения),
* **до 8 байт данных** в классической версии или до **64 байт** в CANopen FD,
* **CRC-контрольную сумму**, которая обеспечивает надёжность передачи.

С точки зрения логики протокола, основными форматами сообщений являются следующие:

1. **PDO (Process Data Objects)** Это короткие сообщения для быстрой передачи переменных — например, значений датчиков или управляющих команд.  
    PDO бывает двух типов: **TPDO** — передача, и **RPDO** — приём.  
    Их главное преимущество — **низкая задержка**, поэтому они применяются в реальном времени.
2. **SDO (Service Data Objects)** Используются для конфигурации и передачи больших объёмов данных — например, при загрузке параметров в устройство.  
    Передача идёт по блокам или сегментно, и требует подтверждения, поэтому SDO медленнее, но гибче.
3. **NMT (Network Management)** Управляют состоянием узлов сети: можно перевести устройство в рабочий режим, остановить его или перезапустить.  
    Команды NMT инициируются **мастер-узлом**.
4. **Heartbeat и Node Guarding** Это механизмы **мониторинга состояния узлов**.  
    Устройства регулярно отправляют сигналы "я жив", и если сигнал не пришёл вовремя — считается, что узел вышел из строя.
5. **SYNC и TIME** Служат для синхронизации действий в сети.  
    Сообщение SYNC, например, может инициировать одновременную отправку PDO от всех устройств.

Все сообщения проходят по **общей шине**, и их приоритет определяется **числовым значением идентификатора**: чем меньше ID, тем выше приоритет. Это позволяет передавать критически важные данные с минимальной задержкой.

# Физические интерфейсы, используемые в CANopen

Теперь немного о **физическом уровне**, то есть о том, **по каким проводам и с какими параметрами работает сеть CANopen**.

CANopen использует тот же физический уровень, что и классический CAN — он описан в стандарте **ISO 11898-2**. Это означает, что устройства соединяются по **двум проводам — CAN\_H и CAN\_L**, работающим в дифференциальном режиме. Такой способ передачи значительно **устойчив к помехам**, что особенно важно в условиях промышленности и транспорта.

Самый распространённый физический интерфейс — это **RS-485-подобная шина**, обычно с кабелем витая пара и общей длиной до **500 метров** при скорости 125 кбит/с. С увеличением скорости длина шины уменьшается:

* до 250 кбит/с — около 250 метров,
* до 1 Мбит/с — не более 40–50 метров.

На концах шины обязательно устанавливаются **терминирующие резисторы по 120 Ом**, чтобы предотвратить отражения сигнала.

Также важным элементом является **структура сети**: CANopen предполагает **топологию "общей шины"**, где все устройства подключаются к одной линии, желательно без длинных "ответвлений" (stubs), которые могут вызывать помехи.

По стандарту, к одной CAN-шине можно подключать **до 127 устройств**, каждое с уникальным идентификатором.  
 Однако на практике число активных узлов обычно ограничено **50–64**, чтобы сохранить стабильную работу и избежать перегрузки шины.

# Надежная передача данных: досталось от CAN

Во первых стандарт CAN описывает на физическом уровне надежность за счет использования двух противоположных по значению шин CAN\_H и CAN\_L. За счет такой топологии достигается возможность преодолевать электромагнитные помехи. Резисторы по краям шины по 120Ом уничтожают помехи которые могли бы быть вызваны отражением сигнала

Теперь рассмотрим пакет по стандарту CAN. Он содержит в себе 15 битную контрольную сумму CRC и механизм подтверждения: бит ACK. Это позволяет гарантировать правильность прочитанных данных.

# Надежная передача данных: в CANOpen

Для начала рассмотрим способ передачи данных - Service Data Objects. По сравнению с PDO он медленнее, работает по принципу клиент-сервер, требует больше сервисной информации. Но из плюсов - это надежный протокол, сродни TCP, ведь каждое сообщение требует подтверждение получения. Такой подход позволяет обнаружить потерю или искажение данных

Так же в SDO используется toggle-bit. При отправке каждого пакета значение toggle-бита инвертируется. Если оно осталось тем же, то прошлый пакет игнорируется, это помогает избежать дупликации данных

Так же в CANOpen применяются меры мониторинга сети:

Во-первых - это node guarding, что в переводе значит “защита узла”. Это циклический опрос состояния узла со стороны узла с повышенным приоритетом. Выполняется по очереди, и в ответ устройство должно ответить сообщением со своим состоянием

Во вторых - это heartbeat. Это сообщение, которое устройство должно циклически отправлять с некоторым промежутком времени. Содержит в себе toggle-бит и Состояние узла, которое может сообщать что устройство выключено, работает, либо же включается. Также с помощью heartbeat устройство говорит другим участникам сети о себе после запуска с помощью heartbeat “pre-operational”

# Роли устройств: Master/Slave

В любой момент времени только одно устройство в сети играет роль главного (Master). Все другие устройства сети считаются подчиненными (Slave). Master выдает запрос, и адресуемое (или адресуемые) устройства отвечают, если протокол подразумевает такое поведение. Такое распределение ролей используется когда в сети есть один NMT-master и много слейвов

# Роли устройств: Client/Server

В протоколе SDO (вспомним что он похож на TCP) применяется модель Client/Server. В ней клиент выдает запрос выгрузки или загрузки данных, что вызывает срабатывание сервера для выполнения задачи. После завершения задачи сервер отвечает подтверждением на запрос

В протоколе SDO мастер сети может прочитать словарь (Object Dictionary) подчиненного узла сети и, если требуется, изменить его

# Роли устройств: Producer/Consumer

Это стандартная взимосвязь типа генератор/потребитель. Вовлекает один генератор и множетсво потребителей. При модели push мы используем непотверждаемую службу по запросу отправителя. При модели pull служба подтверждаемая.

Взаимодействие producer/consumer применяется при синхронизации узлов сети (передача объектов SYNC) и при обмене прикладными данными PDO

# Особенности протокола

Теперь обсудим особенности протокола CanOpen

1. Во-первых CANopen использует концепцию объектного словаря – единой структуры, содержащей все параметры устройства (данные конфигурации, статус, диагностическую информацию). Объектный словарь стандартизирован, что упрощает интеграцию устройств разных производителей и облегчает разработку программного обеспечения для управления устройствами.
2. Стандарт определяет профили устройств (device profiles) для различных классов оборудования (например, приводы, датчики, дисплеи). Это позволяет создать предсказуемое и унифицированное поведение устройств, что упрощает их замену и модернизацию.
3. Протокол разделяет обмен данными на уровни:

– NMT (Network Management) обеспечивает централизованное управление сетью, позволяя переводить устройства между состояниями (инициализация, готовность, запуск и т.д.).

– SDO (Service Data Object) используется для доступа к объектному словарю устройств (чтение/запись параметров) с подтверждением передачи.

– PDO (Process Data Object) предназначен для передачи быстрых, циклических данных без подтверждения, что важно в реальном времени.

Такой принцип разделения позволяет эффективно настроить коммуникацию в зависимости от требований к скорости передачи и надёжности.

1. Использование механизмов heartbeat и node guarding позволяет периодически контролировать работоспособность узлов, обнаруживать сбои и предотвращать распространение ошибок. Протокол наследует надежность канального уровня CAN, включая проверку кадра с помощью CRC и подтверждение приёма (ACK).
2. CANopen поддерживает гибкое распределение ролей: устройства могут работать в режиме ведущего (мастер) для централизованного управления или в режиме ведомых (слейв) для исполнения команд.
3. Протокол позволяет легко добавлять новые узлы в сеть при минимальных изменениях конфигурации, что делает его пригодным для небольших систем и крупных комплексных установок.Возможность динамической перенастройки параметров и изменение топологии сети обеспечивает высокую адаптивность к меняющимся требованиям.

# Сферы использования

1. Промышленная автоматизация: На конвейерных линиях и в системах управления производственными процессами CANopen позволяет объединять многочисленные датчики, приводы и контроллеры для координированной работы.
2. Робототехника и управление приводами. Благодаря низкой задержке и предеачи сигналова в реальном времени, CANOpen широко применяется для управлениями электродвигателями, привода и систем позиционирования
3. Транспорт и мобильные системы. В автомобильной промышленности CANopen используется для управления подсистемами, например, для работы электроприводов, контроля за состоянием узлов, обмена данными между электронными блоками управления (ECU).
4. Автоматизация зданий и умные сети. CANopen может быть использован для управления системами климат-контроля, освещением, системами безопасности и другими инфраструктурными элементами зданий.