Изменения

01.04.2003	1.0	Базовая версия
16.06.2003	1.1	Уточнены правила работы с объектом идентификации Добавлена команда SHUTDOWN в объекте идентификатора Исправлены орфографические ошибки Добавлены команды работы с объектом данных физического узла контроллеров MA444R2.
02.07.2003	1.1a	Добавлен новый поддерживаемый узел в сети MA004
11.12.2003	1.1c	Добавлено описание контроллеров MA004R3, MA444R3 и MD846R3
16.12.2003	1.2	Внесены изменения в функциональность объекта идентификации. Возможна незначительная несовместимость с версиями сети 1.1. Описаны команды доступа к интерфейсу SSI (Stream Storage Interface).
12.01.2004	1.21	Расширена функциональность команды открытия сессии доступа к SSI интерфейсу. Появилась возможность индивидуального добавления контроллеров в сессию. Ответ узла о текущем уровне заполнения хранилища содержит идентификацию узла. Уточнено содержимое неиспользованных полей EXT-пакетов объекту идентификации.
25.08.2004	1.22	Добавлено описание контроллера CSC4.
25.06.2013	1.23	Добавлено описание контроллера MA844.

Содержание

1 АРХИТЕКТУРА СЕТИ 3

2 МОДЕЛЬ ПОТОКОВ ДАННЫХ 4

2.1 Фазы обмена сообщениями 4
2.2 Структура сообщения 6
2.3 Объекты данных 7

3 ПРОТОКОЛ 9

3.1 Упаковка данных 9
3.2 Extended CAN ID 9
3.3 Объект ошибок 10
3.4 Объект общих данных 10
3.5 Объект групповых данных 11
3.6 Объект конфигурации 12
3.7 Объект идентификации 12
3.7.1 STD-сообщения объекта идентификатора 14
3.7.2 EXT1-сообщения объекта идентификации 15

4 ВЕДУЩИЙ УЗЕЛ 21

4.1 Модель ведущего узла 21
4.2 Обработка сообщений ведущим узлом 22
4.2.1 Сообщения объекту ошибок 22
4.2.2 Сообщения объекту данных 23
4.2.3 Сообщения объекту конфигурации 23
4.2.4 Сообщения объекту идентификации 24
4.3 Инициализация ведущего узла 24

5 ВЕДОМЫЙ УЗЕЛ 26

5.1 Модель ведомого узла 26
5.2 Состояния ведомого узла 27
5.3 Обработка сообщений ведомым узлом 29
5.3.1 Сообщения объекту ошибок 30
5.3.2 Сообщения объекту данных 31
5.3.3 Сообщения объекту конфигурации 32
5.3.4 Сообщения объекту идентификации 34
5.4 Инициализация ведомого узла 34

6 ВИДЫ СЕТЕВЫХ УЗЛОВ ПРОЕКТА КТЖ-2 36

6.1 Идентификация периферийных контроллеров 36
6.2 Контроллер MCN-3 37
6.3 Контроллер MA444 / MA444R2 / MA444R3 / MA004 / MA004R3 37
6.3.1 Физический узел контроллера MA444/MA444R2/MA444R3/MA004/MA004R3 39
6.3.2 Логические узлы контроллера MA444/MA444R2/MA004 41
6.3.3 Формальное описание контроллера MA444 42
6.3.4 Формальное описание контроллера MA444R2/MA444R3 43
6.3.5 Формальное описание контроллера MA004/MA004R3 43
6.4 Контроллер MD846 / MD846R2 / MD846R3 43
6.4.1 Физический узел контроллера MD846/MD846R2/MD846R3 44
6.4.2 Логические узлы контроллера MD846/MD846R2/MD846R3 44
6.4.3 Формальное описание контроллера MD846 45
6.4.4 Формальное описание контроллера MD846R2/MD846R3 45

1 Архитектура сети

В разделе описываются основные положения сети CANPro+. Сеть обеспечивает обмен данными между ведущим узлом и множеством узлов ввода/вывода различного типа. В общем случае сеть обеспечивает работу нескольких ведущих узлов с одним множеством устройств ввода/вывода. Сеть обеспечивает подключение, конфигурирование, отключение отдельных узлов сети (в том числе и ведущих) без нарушения работы отдельных узлов и целиком сети.

Физически сеть состоит из узлов и среды передачи, соединяющей узлы. Узлы могут быть двух типов: ведущие и ведомые. Ведущие узлы руководят процессом обмена данными, ведомые осуществляют непосредственный распределенный ввод/вывод, с возможностью выполнения локальных алгоритмов обработки данных и принятия решений. Сеть опирается на CAN 2.0B, поэтому многие характеристики определяются возможностями CAN. К таковым относятся:

- Топология шины
- Скорость передачи данных
- Максимальное расстояние между узлами сети
- Устойчивость сети к помехам

Сетевой протокол основан на событийной модели. Для всех узлов сети (ведущих и ведомых) событиями могут служить как принятые сетевые сообщения, так и изменение собственного состояния (внешних условий). В нормальном состоянии обмен данными инициируют только ведущие узлы. Однако в рамках логики работы сети предусмотрена возможность асинхронного оповещения всех узлов сети об исключительных ситуациях в ведомых узлах.

Защита данных и восстановление из сбойных ситуаций при передаче сообщений осуществляется с использованием встроенных возможностей CAN.

Каждый узел сети имеет возможность обмениваться данными от 16 объектов. Сеть обеспечивает 16 стандартных типов обменов данными с каждым из объектов. В текущей реализации сети определяются следующие объекты:

- Объект ошибок
- Объект общих данных
- Объект групповых данных
- Объект конфигурации
- Объект идентификации

Узел не обязан реализовывать все типы объектов, но функциональность реализованных должна быть обеспечена в полном объеме.

Для ведущих узлов регламентируются перечень и функциональность объектов, которые доступны для обмена данными, событийная модель поведения ведущего узла. Конкретные структурные и временные ограничения могут зависеть от реализации ведущего узла.

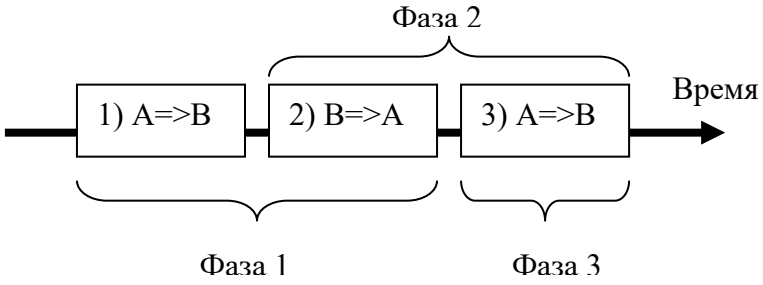
Для ведомых узлов регламентируются перечень и функциональность объектов, которые доступны для обмена данными, событийная модель поведения ведомого узла, логическая организация сложного устройства ввода/вывода.

2 Модель потоков данных

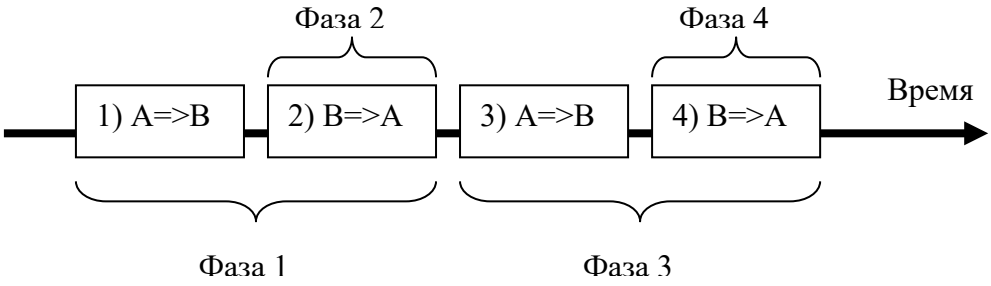
Обмен данными в сети осуществляется между одноименными объектами ведущего и ведомого узлов. В зависимости от типа узла, принятого сообщения, состояния узла и т.д. предпринимаются определенные действия, и формируется ответное сообщение. Ведущие и ведомые узлы должны всегда быть готовы асинхронно принимать сообщение для любого из реализованных объектов и выполнять предписанные в спецификации действия.

2.1 Фазы обмена сообщениями

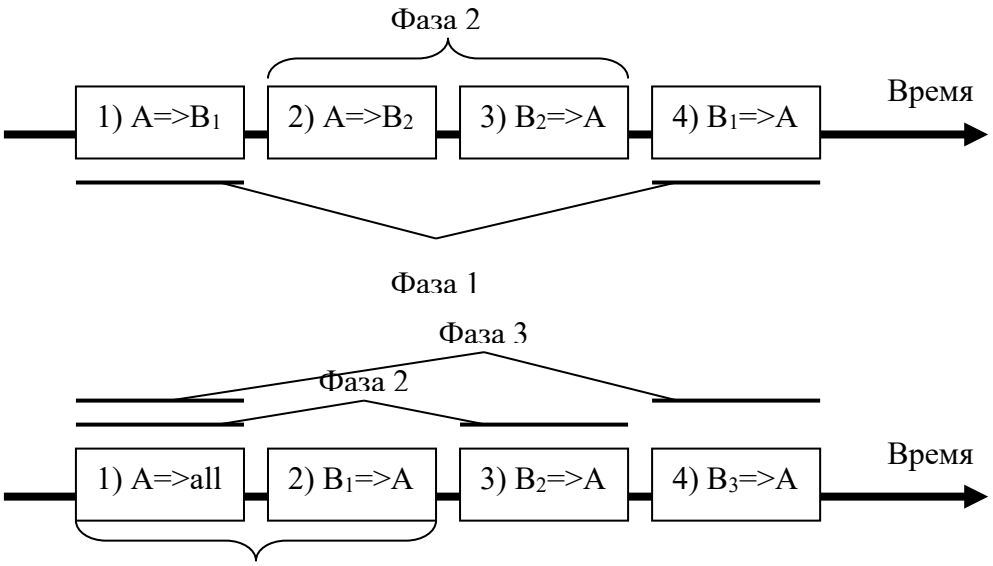
При отправке сообщения узел не обязан ожидать на него ответного сообщения, а может продолжать обрабатывать любые другие сообщения, в том числе и генерировать асинхронные, согласно логике работы сети. Особенно важно такое поведение ведущего узла при сборе данных с узлов ввода/вывода. Нет необходимости ожидать ответа от каждого узла последовательно, можно раздать запросы на чтение данных (в крайнем случае, всего один широковещательный) и потом асинхронно обрабатывать ответы, автоматически определяя отказавшие узлы и не теряя времени на ожидание ответа от таких узлов.
Не все передаваемые в сеть сообщения подразумевают (требуют) передачи ответного сообщения, однако в общем случае обмен сообщениями можно рассматривать именно как запрос/ответ, где сообщение-ответ может являться запросом в следующей паре запрос/ответ. Пара логически связанных сообщений называется фазой обмена и является базовым элементом обмена данными в сети.
На рисунке представлены 3 фазы обмена сообщениями между узлами A и B. Видно, что ответ фазы 1 является запросом фазы 2. Обмен в фазе 3 не предусматривает ответного сообщения.



В некоторых случаях картина соотношения фаз и сообщений может изменяться.

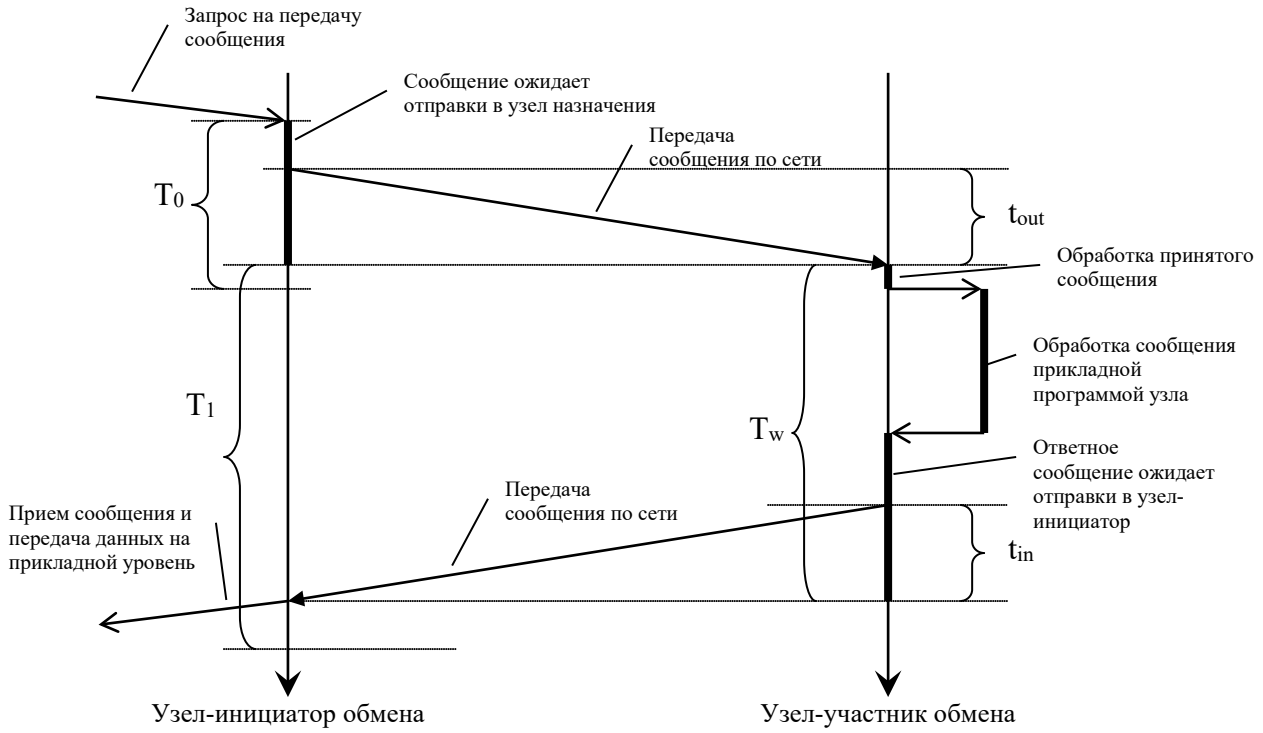


Сообщения, передаваемые узлом A (фазы 1 и 3) не требуют ответных сообщений и соответствующие фазы (фаза 2 и 4) для узла B сократились до одного сообщения. Логика связи фазы 2 и фазы 3 в узле A лежит вне сети и определяется прикладной задачей (например, последовательный опрос ячеек). Фазы могут перекрываться или иметь общие сообщения. Такие ситуации представлены ниже.



Правила передачи сообщений в сети определяются приоритетами, состоянием узла и временной моделью фазы обмена сообщениями. Если в узле на передачу имеется несколько сообщений, передаваться должно самое приоритетное (о приоритетах сообщений см. 2.3 Объекты данных). Важно, что если в момент ожидания передачи сообщения появляется более приоритетное на передачу должно быть отправлено именно новое, более приоритетное сообщение, а старое помещено в очередь.

Состояния узла могут “маскировать” некоторые объекты. Т.е. в некоторых состояниях узлов определенные объекты становятся недоступны, и прием/передача сообщений для таких объектов приостанавливается.



Временные задержки передачи наиболее приоритетного на текущий момент сообщения определяются временной моделью фазы обмена сообщениями.
После того, как регистрируется запрос на передачу сообщения, сетевая подсистема узла-инициатора создает новое сообщение, заполняет необходимые поля и помещает сообщение в очередь на передачу. Сообщению устанавливается временная метка помещения в очередь на передачу. Сообщение участвует в проверке приоритетов среди сообщений, находящихся в узле в очереди на передачу, передающихся по сети в данный момент. Производятся попытки повторной передачи сообщения в случае ошибки передачи. Все эти процессы продолжаются не более T_0. В течение этого времени конкретное сообщение должно быть доставлено в узел назначения. Очевидно, что $T_0 > t_{out}$, где t_{out} время передачи сообщения по сети, которое зависит от скорости шины и размера сообщения. То, на сколько T_0 больше t_{out} зависит от типа объекта, направления и других параметров сообщения. Также на это влияет целевая система, в которой используется данная сеть. В общем случае параметр T_0 определяет задержку до момента захвата сообщением времени среды передачи. Если в течение интервала T_0 после размещения сообщения в очереди на передачу оно не было доставлено в целевой узел, сообщение удаляется и на прикладной уровень возвращается статус операции TIMEOUT0, что обозначает перегруженность сети в данный момент для данного типа сообщений. TIMEOUT0 также следует возвращать если в настоящий момент очередь передачи переполнилась.
С момента доставки сообщения в узел назначения (если для данного сообщения предусмотрен ответ), в узле-инициаторе для этого сообщения начинает отсчитываться интервал времени T_1, отпущенный узлу на ответ. Т.о. фаза обмена сообщениями никогда не будет длиться дольше, чем T_0+T_1. Если в течение времени T_1 ответ получен не будет, то на прикладной уровень возвращается статус операции TIMEOUT1, что, в общем случае, обозначает проблемы с функционированием узла-назначения.
При получении сообщения, на который нужно сформировать ответ узел назначения отсчитывает для него время T_w, отведенное на обработку запроса и передачу ответного сообщения в узел инициатор. Узел назначения обрабатывает запросы в порядке приоритетов и с учетом “маскированных” объектов, так что возможны ситуации, что интервал T_w истечет во время ожидания передачи, во время обработки или в очереди на обработку. Обрабатывать или нет сообщение в последних двух случаях решает узел назначения самостоятельно (это не регламентируется), но ответного сообщения в любом из трех случаев передано не должно быть.
Все три характеристики T_0, T_1 и T_w могут быть выбраны разработчиком сети, в зависимости от типов объектов, сущности узлов, природы данных и стиля работы прикладной программы.

2.2 Структура сообщения

Каждое сообщение в сети характеризуется следующими параметрами:

- Объект. Тип объекта, которому направляется сообщение. Т.к. обмен данными возможен только между одноименными объектами, то этот параметр также определяет объект-источник сообщения.
- Направление. Направление сообщения определяет тип узла отправителя и получателя сообщения. Направление может быть двух типов: от ведущего узла к ведомому или от ведомого к ведущему. Два ведущих или два ведомых узла не могут обмениваться сообщениями непосредственно в рамках сети. Иногда требуется обеспечить обмен данными между ведущими устройствами сети. В этом случае программное обеспечение ведущего устройства помимо основной

функциональности должно реализовать необходимую функциональность ведомого устройства.

- Тип. Определены два типа сообщений: запрос (REQ) и подтверждение (ACK). Тип сообщения не влияет на его приоритет. Ни в коем случае не нужно считать, что REQ - это запрос (первое сообщение фазы обмена), а ACK - это ответ (второе сообщение фазы обмена). Тип сообщения несет совершенно другую смысловую нагрузку и никак не связан с порядком следования сообщений в фазе обмена.
- Адрес. Параметр идентифицирует узел отправителя (для направления от ведомого к ведущему) или узел получателя (для направления от ведущего к ведомому). Имеются специальные адреса для адресации групп узлов или всех узлов сети.
- Метка времени. Метка времени определяет “возраст” сообщения. Для разных типов узлов, направлений, типов сообщений, адресов определяется максимальное время, после которого сообщение становится неактуальным. Собственно в самом сообщении в среде передачи метка времени не передается. Метка времени существует у сообщения на ведущем узле и на ведомом. Времена на узлах идут совершенно независимо и синхронизации не требуют.
- Данные. Данные сообщения предназначаются непосредственно объекту сообщения. Их формат и содержание полностью определяется объектом конкретного узла. Сеть не накладывает никаких ограничений на тип и формат представления данных объекта, кроме размера. Подробнее см. раздел 3 Протокол. Сообщения, у которых поле данных пустое (DLC = 0), называются пустыми.

Целостность передачи сообщения обеспечивается сетью CAN. Сообщение передается в одном пакете CAN, так что системе на верхнем уровне не нужно заботиться о гарантированной доставке всех частей сообщения.

2.3 Объекты данных

Весь обмен по сети выражен в обмене сообщениями между объектами. Всего определяется 16 стандартных объектов, между которыми происходит независимый и параллельный обмен данными. Любое сообщение в сети инициируется одним из объектов и получает его только одноименный объект. Отправить сообщение от одного объекта, чтобы его получил другой объект в рамках невозможно. В случае необходимости такое взаимодействие должно реализовываться на уровне прикладной задачи в узлах сети. С точки зрения прикладной задачи сеть обеспечивает 16 независимых каналов передачи сообщений. Все 16 каналов могут одновременно существовать в сети.

Каждый узел может обеспечивать поддержку до 16-ти стандартных объектов данных. Объекты в узле пронумерованы 0 – 15 и имеют стандартные приоритеты. Приоритет выше того объекта, чей номер меньше. Это правило распространяется на всю сеть и обозначает, что сообщения для объектов с меньшим номером должны приниматься, обрабатываться и передаваться по сети в первую очередь. В узлах эти приоритеты реализуются программным или любым другим способом, в среде передачи – путем выбора кодирования данных в пакете CAN (см. раздел 3 Протокол).

Несмотря на то, что в принципе каждый узел может поддерживать до 16 объектов одновременно, в каждый конкретный момент времени это количество может быть меньше, из-за ограничений вызванных состоянием узла. В различных состояниях узла некоторые из объектов данных могут быть недоступны и для остальных узлов сети они перестают существовать. Со стороны сети такие объекты выглядят как нереализованные, узел не обрабатывает входящих сообщений для таких объектов и не генерирует исходящих из таких объектов сообщений.

В настоящей версии сети определяется всего несколько объектов. Остальные зарезервированы для расширений. Однако, если допустимо пожертвовать совместимостью с будущими версиями сети, в конкретной реализации зарезервированные объекты можно использовать для обмена сообщениями совершенно произвольным образом. Типы и номера объектов, а, следовательно, и приоритеты, реализованных в текущей версии сети, приведены в таблице.

Приоритет	Номер	Название	Назначение
Высший	0	Объект ошибок	Данные об ошибочных состояниях узлов, входов и выходов.
	1	Объект общих данных	Данные ввода/вывода всего узла целиком.
	2	Объект групповых данных	Данные ввода/вывода отдельных каналов (групп каналов) узла.
	3	-	Не определен. Зарезервирован для расширений.
	4	-	Не определен. Зарезервирован для расширений.
	5	-	Не определен. Зарезервирован для расширений.
	6	-	Не определен. Зарезервирован для расширений.
	7	-	Не определен. Зарезервирован для расширений.
	8	-	Не определен. Зарезервирован для расширений.
	9	-	Не определен. Зарезервирован для расширений.
	10	-	Не определен. Зарезервирован для расширений.
	11	-	Не определен. Зарезервирован для расширений.
	12	-	Не определен. Зарезервирован для расширений.
	13	-	Не определен. Зарезервирован для расширений.
	14	Объект конфигурации	Конфигурирование узла, изменение режимов работы.
Низший	15	Объект идентификации	Идентификация узла, установка сетевого адреса и уникального идентификатора, передача специальных запросов и сообщений

3 Протокол

3.1 Упаковка данных

Протокол передачи сообщений базируется на CAN 2.0B и одно сообщение отображается на один пакет сети CAN расширенного формата (extended data frame, поле RTR – “рецессивное”). Отсюда вытекает основное ограничение на структуру сообщения: поле данных любого сообщения в сети не может быть больше 8 байтов. Упаковка данных осуществляется следующим образом:

- Объект, направление, тип и адрес сообщения передаются в расширенном идентификаторе пакета CAN.
- Количество данных передается в поле DLC пакета CAN.
- Данные передаются в поле DATA пакета CAN.

Данные сообщений интерпретируются в зависимости от типа объекта. Формат нерегламентированных данных определяется прикладной задачей узлов, участвующих в обмене сообщением.

Биты поля данных нумеруются с 0 до 63, начиная с младшего бита первого байта поля данных CAN пакета. Если передаваемые данные занимают нецелое количество байтов, биты последнего байта сохраняются в младшей части байта. Остальные биты байта обнуляются.

3.2 Extended CAN ID

Отображение полей сетевого сообщения на идентификатор пакета CAN.

28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
signature				OBJ				D	NA							LA				signature								T
1	1	0	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	1	1	1	1	1	1	1	1	x

- signature – сигнатура сообщения. Сообщения, для которых значение поля сигнатура не совпадает с указанным на рисунке, все узлы сети должны игнорировать.
- OBJ – номер типа объекта. Нумерация соответствует приведенной в таблице в разделе 2.3 Объекты данных.
- D – направление сообщения. 0 – сообщение передается от ведущего узла ведомому, 1 – сообщение передается от ведомого узла ведущему. Узлы обязаны игнорировать сообщения предназначенные не им.
- NA – сетевой (физический) адрес узла. Число в пределах 1-126, являющееся физической частью адреса узла. Значение 0 зарезервировано. Значение 127 определяет широковещательный запрос всем узлам в сети. Сетевой (физический) адрес узла устанавливается при инициализации физического узла и изменяется только при его рестарте.
- LA – логический адрес узла. Число в пределах 1-14, являющееся логической частью адреса узла. Значение 0 зарезервировано для доступа к объектам физического контроллера с адресом NA. Значение 15 определяет широковещательный запрос всем логическим узлам в физическом контроллере с адресом NA.

- T – тип сообщения. 0 – REQ, 1 – ACK.

В общей сложности сеть обеспечивает работоспособность 126*14 = 1764 логических ведомых узлов. Подробнее о разделении ведомых узлов на физический и логические см. в разделе 5 Ведомый узел.

3.3 Объект ошибок

Объект предназначен для передачи сообщений о статусе элементов ввода/вывода ведомых узлов. Обмен сообщениями объекта ошибок может быть инициирован любым узлом.

28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
signature				OBJ				D	NA							LA				Signature								T
1	1	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x	x	x	1	1	1	1	1	1	1	1	x

Поле данных содержит N битов состояний элементов ввода/вывода узла. Каждому элементу соответствует 1 бит. Биты в поле данных упакованы в порядке возрастания элементов ввода/вывода.
0 – элемент ввода/вывода функционирует нормально
1 – обнаружена неполадка или элемент несконфигурирован. Данные элемента неверны.

7	6	5	4	3	2	1	0
-	-	-	e_4	e_3	e_2	e_1	e_0
0	0	0	1	0	1	1	0

DLC = 1

Поле данных сообщения для объекта ошибок узла с пятью элементами ввода/вывода данные о статусе элементов
e0 – в порядке
e1 – ошибка, данные элемента не верны
e2 – ошибка, данные элемента не верны
e3 – в порядке
e4 – ошибка, данные элемента не верны

Для объекта ошибок физического узла в настоящее время определены только два бита статуса.
e0 – зарезервировано, = 0
e1 – ошибка питания

3.4 Объект общих данных

Объект предназначен для передачи сообщений с данными элементов ввода/вывода ведомого узла целиком. Обмен сообщениями объекта общих данных может быть инициирован любым узлом. Передаваемые данные описывают все элементы ввода/вывода узла.

28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
signature				OBJ				D	NA							LA				Signature								T
1	1	0	0	0	0	0	1	x	x	x	x	x	x	x	x	x	x	x	x	1	1	1	1	1	1	1	1	x

Поле данных содержит S битов данных элементов ввода/вывода узла. Каждому элементу соответствует s_i бит. Биты в поле данных упакованы в порядке возрастания элементов ввода/вывода. Порядок следования битов и содержимое данных элемента ввода/вывода определяется прикладной задачей узлов.
S = sum s_i, 1<=i<=N, где N – количество элементов ввода/вывода узла.

23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	D4		D3					D2					D1					D0				
0	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x

DLC = 3

Поле данных сообщения для объекта общих данных узла с пятью элементами ввода/вывода первые 4 имеют данные по 5 битов, пятый элемент – 2 бита.

3.5 Объект групповых данных

Объект предназначен для передачи сообщений с данными группы или индивидуальных элементов ввода/вывода ведомого узла. Обмен сообщениями объекта групповых данных может быть инициирован любым узлом. Передаваемые данные описывают группу элементов ввода/вывода узла. Объект по своим свойствам похож на объект общих данных. Отличие состоит в том, что данный объект позволяет индивидуально адресовать элементы ввода вывода логического узла.

28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
signature				OBJ				D	NA							LA				Signature								T
1	1	0	0	0	0	1	0	x	x	x	x	x	x	x	x	x	x	x	x	1	1	1	1	1	1	1	1	x

Поле данных содержит N+S битов данных элементов ввода/вывода узла и карта активных элементов. Каждому элементу соответствует s_i бит. Биты в поле данных упакованы в порядке возрастания элементов ввода/вывода. Первые N битов данных определяют карту активных элементов ввода/вывода. 1 – элемент активен, его данные присутствуют в блоке данных, 0 – элемент неактивен, данные отсутствуют (пропущены). Порядок следования битов и содержимое данных элемента ввода/вывода определяется прикладной задачей узлов. Указание карты из всех 1 эквивалентно использованию сообщения для объекта общих данных (поле данных из-за карты в случае групповых данных будет больше).
S = sum s_i, 1<=i<=K, где K – количество элементов ввода/вывода в карте активных элементов.

23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	D4		D3					D1					MAP				
0	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x	x	x	1	1	0	1	0

DLC = 3

Поле данных сообщения для объекта групповых данных узла с пятью элементами ввода/вывода первые 4 имеют данные по 5 битов, пятый элемент – 2 бита. В карте активных элементов выбраны элементы 1, 3 и 4.

3.6 Объект конфигурации

Объект предназначен для передачи конфигурационных сообщений ведомым логическим узлам. Обмен сообщениями объекта конфигурации может быть инициирован любым узлом. Передаваемые данные описывают режимы работы и настройки всех элементов ввода/вывода ведомого узла.

28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
signature				OBJ				D	NA							LA				Signature								T
1	1	0	0	1	1	1	0	x	x	x	x	x	x	x	x	x	x	x	x	1	1	1	1	1	1	1	1	x

Поле данных содержит S битов конфигурационных данных элементов ввода/вывода узла. Каждому элементу соответствует s_i бит. Биты в поле данных упакованы в порядке возрастания элементов ввода/вывода. Порядок следования битов и содержимое конфигурационных данных элемента ввода/вывода определяется прикладной задачей узлов.
S = sum s_i, 1<=i<=N, где N – количество элементов ввода/вывода узла.

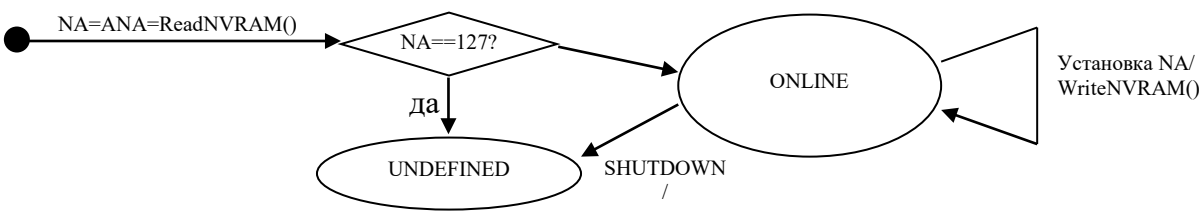
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CFG4				CFG3				CFG2		CFG1			CFG0		
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x

DLC = 2

Поле данных сообщения для объекта конфигурации узла с пятью элементами ввода/вывода первые 4 имеют конфигурационные данные по 3 бита, пятый элемент – 4 бита.

3.7 Объект идентификации

Объект предназначен для работы с сетевым адресом физического узла и уникальным идентификатором. Объект позволяет выполнять над физическим узлом специальные операции. Обмен сообщениями объекта идентификации может быть инициирован только ведущим узлом.
В процессе работы сетевому узлу может понадобиться изменять свой сетевой адрес. Сетевой (физический) адрес узла может быть изменен только его перезагрузкой. Для изменения сетевого адреса без перезагрузки узла вводится понятие прикладного сетевого адреса. При старте узла его физический сетевой адрес устанавливается равным значению прикладного сетевого адреса, сохраненному в энергонезависимой памяти. Прикладной сетевой адрес в процессе работы может быть изменен командами с ведущего узла сети. При изменении прикладного сетевого адреса он сохраняется в энергонезависимой памяти узла и после рестарта узла становится его физическим сетевым адресом.



Общий принцип работы объекта идентификации заключается в том, что нераспознанные форматы сообщений и/или неподдерживаемые команды игнорируются объектом.

Объект идентификации воспринимает сообщения с полем данных размером 8 байтов. На все остальные размеры поля данных входных сообщений объект идентификации отвечает стандартным ответом следующего формата (OTHER-ответ).

28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
signature				OBJ				D	NA							LA				Signature								T
1	1	0	0	1	1	1	1	x	x	x	x	x	X	x	x	x	x	x	x	1	1	1	1	1	1	1	1	x

Поле данных содержит 7 байтов, в которые входят прикладной сетевой адрес, уникальный идентификатор устройства.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ID2								ID1								ID0								ANA							
x	x	x	x	x	x	x	X	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x

55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
ID5								ID4								ID3							
X	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x

DLC = 7

ANA – прикладной сетевой адрес ведомого узла.
ID[0..5] – байты уникального идентификатора физического узла.

Ответное сообщение имеет тип REQ и длину поля данных 7 байтов (REQ7). Оно может быть использовано для получения прикладного сетевого (физического) адреса узла и уникального идентификатора. В ответном сообщении поля заполняются в соответствии со значениями параметров узла. Узел создает сообщение со своими текущими ANA (прикладным сетевым адресом) и ID[0..5] (уникальным идентификатором). Байт ID[5] замещается на код версии программного обеспечения и может не совпадать со значением, указанным на корпусе контроллера.
При получении REQ7 сообщения от узла по полям NA и ANA можно определить его состояние.

NA	ANA	Состояние
1-126	== NA	Узел находится в нормальном состоянии. После рестарта узла его ANA не изменялся.
1-126	1-126 != NA	Узел находится в нормальном состоянии. После рестарта узла его ANA был изменен и вступит в силу (станет NA) после рестарта узла.
1-126	0	Узел находится в нормальном состоянии. После рестарта значение NA станет равным 0, что недопустимо. Необходимо изменить значение ANA до рестарта узла.
127	x	Физический сетевой адрес узла в энергонезависимой памяти некорректен. Узел находится в состоянии UNDEFINED.
1-126	127	Узлу была отдана команда SHUTDOWN

Выдача в сеть широковещательного ACK0 (тип ACK, количество данных 0) сообщения идентификационному объекту позволяет узнать количество, идентификаторы и адреса всех активных в данный момент физических узлов.

Объект идентификации может поддерживать несколько типов входных сообщений с размером поля данных равным 8 байтам (поведение при получении сообщения с размером поля данных 0-7 байтов см. выше). Форматы таких сообщений представлены ниже

28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
signature				OBJ				D	NA							LA				Signature								T
1	1	0	0	1	1	1	1	x	x	x	x	x	X	x	x	x	x	x	x	1	1	1	1	1	1	1	1	x

Поле данных содержит 8 байтов, в которые входят команда и специфичные для команды данные.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
D3								D2								D1								D0							
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
CMD				type		f1	f0	D6								D5								D4							
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x

DLC = 8

D[6..0] – специфичные для команды данные
f[1..0] – флаги, влияющие на исполнение команды
type – тип сообщения объекта идентификатора
CMD – код команды

Определяется 2 типа сообщений объекта идентификатора: *STD-сообщения* (type=0) и *EXT1-сообщения* (type=1).

3.7.1 STD-сообщения объекта идентификатора

28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
signature				OBJ				D	NA							LA				Signature								T
1	1	0	0	1	1	1	1	x	x	x	x	x	X	x	x	x	x	x	x	1	1	1	1	1	1	1	1	x

Поле данных содержит 8 байтов, в которые входят прикладной сетевой адрес, уникальный идентификатор и команда. Тип сообщения – REQ.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ID2								ID1								ID0								ANA							
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
CMD				type		a	i	ID5								ID4								ID3							
x	x	x	x	0	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x

DLC = 8 или 7

ANA – прикладной сетевой адрес ведомого узла.
ID[0..5] – байты уникального идентификатора физического узла.
i – маска идентификатора
a – маска физического адреса
CMD - команда

Если входное STD-сообщение имеет тип ACK, то объект идентификации отвечает OTHER-ответом.
Входное сообщение REQ8 выдает контроллеру команду. При получении сообщения узел анализирует биты 63..56 и выполняет запрошенную команду. Для выполнения команды может потребоваться дополнительное совпадение сетевого (физического) адреса и/или идентификатора. Если поле i = 0 (бит 56), то чтобы выполнить запрошенную команду у узла должен совпадать уникальный идентификатор, с идентификатором, переданным в сообщении (биты 55..8). Если i=0 и идентификаторы не совпадают, сообщение *игнорируется*. Режим сравнения (совпадения) идентификаторов контроллеров зависит от производителя устройств и должен обеспечивать уникальность контроллеров в сети. Если поле a = 0 (бит 57), то чтобы выполнить запрошенную команду у узла должен совпадать сетевой (физический) адрес, с адресом, переданным в сообщении (биты 7..0). Если a=0 и адреса не совпадают, сообщение *игнорируется*. Если необходимые поля совпали, узел выполняет запрошенную команду. Команда кодируется полем CMD. Определены 4 команды:

CMD = 5_d (0101_b) установить идентификатор. Узел устанавливает свой уникальный идентификатор равным переданному значению (биты 55..8). Новое значение вступает в силу после перезагрузки узла. Допускается, чтобы изменения вступали в силу немедленно (без перезагрузки узла).

CMD = 3_d (0011_b) установить прикладной сетевой адрес. Узел устанавливает свой прикладной сетевой адрес равным переданному значению (биты 7..0). Системный сетевой адрес будет изменен только после перезагрузки узла. Допускается, чтобы изменения вступали в силу немедленно (без перезагрузки узла).

CMD = 10_d (1010_b) запрос на перезапуск узла. Физический узел немедленно перезагружается, до завершения процесса перезагрузки ресурсы узла полностью недоступны в сети.

CMD = 12d (1100b) SHUTDOWN, запрос на перевод узла в состояние UNCONFIGURED, прикладной сетевой адрес устанавливается равным 127. Из этого состояния контроллер может быть выведен только запросом на перезапуск узла.

Если код команды (CMD) не распознается узлом (он не поддерживает такую команду или это неизвестный для узла код команды), то узел *игнорирует* сообщение.
После выполнения (не игнорирования) команды узел отправляет ответное сообщение, по формату совпадающее с ответным сообщением на сообщение типа OTHER.
Ответное сообщение на REQ8 запрос имеет тип ACK и длину поля данных 8 байтов (ACK8). Биты 63..56 для нового сообщения повторяются из REQ8 сообщения.

3.7.2 EXT1-сообщения объекта идентификации

Поле данных содержит 8 байтов, произвольного содержания, кроме байта команды. Тип сообщения может быть как REQ, так и ACK.

В настоящей версии сети определяются две стандартных команды для EXT1-сообщений. Данные команды используются для обеспечения доступа к SSI-интерфейсу узла.

Доступ к SSI-интерфейсу узла через объект идентификации

SSI-интерфейс узла, если он реализован, через объект идентификации доступен только в том случае, когда сам узел находится в состоянии UNCONFIGURED. Перед использованием объекта идентификации для доступа к SSI-интерфейсу в случае необходимости выдайте узлу команду SHUTDOWN (см. STD-сообщения объекта идентификации).

Общий формат входного сообщения для доступа к SSI-интерфейсу приведен ниже

28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
signature				OBJ				D	NA							LA				Signature								T
1	1	0	0	1	1	1	1	x	x	x	x	x	X	x	x	x	x	x	x	1	1	1	1	1	1	1	1	x

Поле данных содержит 8 байтов, в которые входят идентификатор сессии, статус, данные и код операции. Тип сообщения – REQ.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA[0]								STATUS																SES_ID							
x	x	x	x	x	X	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
CMD				type		f	e	DATA[3..1]																							
x	x	x	x	0	1	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x

DLC = 8

SES_ID – идентификатор сессии работы с SSI-интерфейсом
STATUS – статус выполнения операции
DATA[3..0] – данные выполнения операции
e – расширения кода операции
f – флаг выполнения операции
CMD – код операции

Команда кодируется полем CMD. Определены 2 команды:
CMD = 0_d (0000_b) Управляющая команда открытия сессии.
CMD = 1_d (0001_b) Команда обмена данными.
В случае получения любой другой команды, или сообщения с типом ACK, такие сообщения игнорируются.

Общий принцип работы с SSI-интерфейсом по сети CANPro+

Основным назначением SSI-интерфейса узлов является предоставление возможности обновления системного ПО. Базовое API SSI-интерфейса состоит из 5 функций:

SsiUnlock(unlockKey) – Открытие доступа к хранилищу для последующего размещения обновлений системного ПО. Параметр unlockKey используется в алгоритмах собственно для открытия доступа к хранилищу.

SsiAllocate(size) – Резервирование области данных в хранилище. Параметр size – размер области в элементах хранения (32-битные слова).

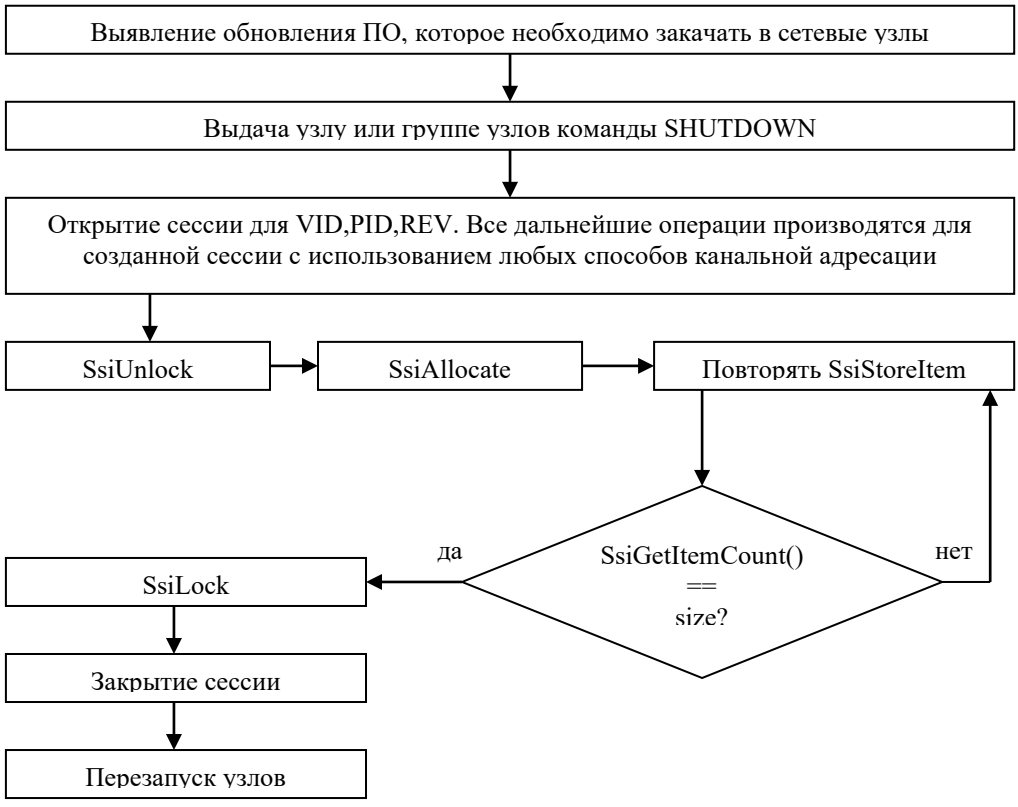
SsiStoreItem(data) – Сохранение элемента данных в хранилище.

offset SsiGetItemCount() – Количество элементов, сохраненных в зарезервированной области хранилища.

SsiLock(status) – Закрытие хранилища. status – команда на закрытие хранилища (0-сбросить все сохраненные данные,1-интерпретировать сохраненные данные как обновления системного ПО)

Подробнее об SSI-интерфейсе можно узнать в документе ssi_spec.pdf.

Сеть CANPro+ позволяет в широковещательном режиме обновлять системное ПО однотипных узлов с автоматическим контролем версий. Одновременно (параллельно и независимо) в сети может существовать несколько процессов загрузки обновлений. Для обеспечения независимости этих процессов используется понятие *сессии*, которая открывается вначале загрузки обновлений и закрывается по завершению загрузки. Узел, в котором открыта (активна) сессия, не отвечает на запросы других сессий и не открывает новые сессии до закрытия текущей активной. Сессия идентифицирует обновление конкретной версии системного ПО для конкретного типа сетевых узлов. Этот механизм позволяет сделать процессы загрузки обновлений ПО независимыми для разных типов узлов и версий ПО, даже при использовании широковещательных запросов.
Для загрузки обновлений системного ПО предпочтительней использовать именно широковещательные запросы, т.к. за время обновления одного узла будут обновлены все узлы сети такого типа[1]. При открытии сессии имеется возможность указать VID,PID сетевого узла и REV обновления ПО. Каждый из узлов имеет возможность проверить на совпадение VID,PID и сравнить текущую ревизию ПО с ревизией предлагаемых обновлений, указанную при открытии сессии. Если все параметры узел удовлетворяют, он делает сессию активной и начинает работу с SSI-интерфейсом.
Обобщенный алгоритм доступа к SSI-интерфейсу узла для сети CANPro+ следующий:



[1] Если вы не хотите обновлять системное ПО конкретного узла, но при этом желаете использовать широковещательные запросы к узлам такого типа, не выдавайте узлу команду SHUTDOWN и он будет игнорировать сообщения доступа к SSI-интерфейсу.

Форматы команд доступа к SSI-интерфейсу

Как было сказано выше определены 2 кода операции (0 и 1). Для CMD=1 в случае поля f=1 определяется вторичный код команды, который передается в сообщении в поле DATA[3].

Код операции
CMD = 0_d (0000_b) Управляющая команда открытия сессии.
Действия
Команда открывает сессию.
Параметры

SES_ID	идентификатор сессии работы с SSI-интерфейсом. В случае успешного сравнения параметров сообщения сессия с таким идентификатором должна быть открыта.
STATUS	e=0 : unlockKey для вызова SsiUnlock e=1 : серийный номер (SN) узла
DATA[0]	VID типа сетевого узла, которому предназначается обновление
DATA[2..1]	PID типа сетевого узла, которому предназначается обновление DATA[1] – $PID_{7..0}$, DATA[2] – $PID_{15..8}$
DATA[3]	ревизия предлагаемого обновления
e	флаг контроля серийного номера
f	флаг контроля ревизии

Обработка
При получении сообщения узел должен находиться в состоянии UNCONFIGURED. Если узел имеет открытую сессию, не совпадающую с передаваемой в команде, команда игнорируется.
Команды с e=1 предназначены для добавления индивидуальных контроллеров в сессию. Команды с e=0 предназначены для отмыкания хранилища и доступа к интерфейсу SSI.
При получении команды с e=1 узлы, у которых уже есть открытая сессия, команду игнорируют. Если открытой сессии нет, узел сравнивает тройку VID,PID,SN со значениями своего идентификатора. В случае f=1 также убеждается, что предлагаемая ревизия ПО строго *больше* той, что имеет он сейчас. В случае успеха узел открывает новую сессию SES_ID.
При получении команды с e=0 узел, у которого нет открытой сессии, сравнивает пару VID,PID со значениями своего идентификатора. В случае f=1 также убеждается, что предлагаемая ревизия ПО строго *больше* той, что имеет он сейчас. При неудаче команда игнорируется. В противном случае открывается сессия SES_ID. Узел, уже имевший открытую сессию или только что ее открывший, вызывает локальную функцию SsiUnlock, параметром которой становится значение поля STATUS.
Ответ
Нет

Код операции
CMD = 1_d (0001_b) Команда обмена данными.
Вторичный код команды = нет (т.е. поле f=0)
Действия
Передача элемента данных

Параметры

SES_ID	идентификатор сессии работы с SSI-интерфейсом. Сессия должна была быть открыта управляющей командой открытия сессии (CMD=0)
STATUS	требуемое смещение элемента данных в хранилище
DATA[3..0]	32-х битный элемент данных для сохранения i DATA[3]=$i_{31..24}$, DATA[2]=$i_{23..16}$, DATA[1]=$i_{15..8}$, DATA[0] =$i_{7..0}$
e	не используется, должно быть равно 0
f	0

Обработка

При получении сообщения узел должен находиться в состоянии UNCONFIGURED и его активная сессия должна быть равна SES_ID. Если текущее смещение в хранилище полученное функцией SsiGetItemCount() совпадает с требуемым (параметр STATUS), то узел сохраняет очередной элемент в хранилище с помощью вызова SsiStoreItem(DATA). В противном случае узел не предпринимает никаких действий.

Ответ

Нет

Код операции

CMD = 1_d (0001_b) Команда обмена данными.

Вторичный код команды = 0 (т.е. поле f=1, DATA[3]=0)

Действия

Резервирование места в хранилище

Параметры

SES_ID	идентификатор сессии работы с SSI-интерфейсом. Сессия должна была быть открыта управляющей командой открытия сессии (CMD=0)
STATUS	требуемый размер хранилища
DATA[2..0]	не используется, должно быть равно 0
DATA[3]	вторичный код команды, = 0
e	не используется, должно быть равно 0
f	1

Обработка

При получении сообщения узел должен находиться в состоянии UNCONFIGURED и его активная сессия должна быть равна SES_ID. Узел вызывает функцию SsiAllocate(STATUS).

Ответ

Нет

Код операции

CMD = 1_d (0001_b) Команда обмена данными.

Вторичный код команды = 1 (т.е. поле f=1, DATA[3]=1)

Действия

Получение количества элементов данных в хранилище

Параметры

SES_ID	идентификатор сессии работы с SSI-интерфейсом. Сессия должна была быть открыта управляющей командой открытия сессии (CMD=0)
STATUS	ожидаемое количество сохраненных элементов
DATA[1..0]	не используется, должно быть равно 0
DATA[2]	пользовательский маркер запроса
DATA[3]	вторичный код команды, = 1
e	не используется, должно быть равно 0
f	1

Обработка
При получении сообщения узел должен находиться в состоянии UNCONFIGURED и его активная сессия должна быть равна SES_ID. Узел вызывает функцию SsiGetItemCount(). Если полученное значение оказалось меньше ожидаемого (STATUS) или ожидаемое значение было равно 0, то узел отправляет ответное сообщение. В противном случае узел не предпринимает никаких действий.
Ответ
Ответное сообщение совпадает по формату с входным, имеет тип ACK и в поле STATUS содержит результат вызова функции SsiGetItemCount(), в поле DATA[1..0] – серийный номер узла.

SES_ID	идентификатор сессии работы с SSI-интерфейсом
STATUS	реальное количество сохраненных узлом элементов
DATA[1..0]	серийный номер узла
DATA[2]	пользовательский маркер запроса, значение поля совпадает ссо значением поля во входном сообщении
DATA[3]	вторичный код команды, = 1
e	не используется, должно быть равно 0
f	1

Код операции
CMD = 1_d (0001_b) Команда обмена данными.
Вторичный код команды = 2 или 3 (т.е. поле f=1, DATA[3]=2 или 3)
Действия
Закрытие сессии, закрытие сессий с рестартом.
Параметры

SES_ID	идентификатор сессии работы с SSI-интерфейсом. Сессия должна была быть открыта управляющей командой открытия сессии (CMD=0)
STATUS	статус завершения сессии
DATA[2..0]	не используется, должно быть равно 0
DATA[3]	вторичный код команды, = 2 или 3
e	не используется, должно быть равно 0
f	1

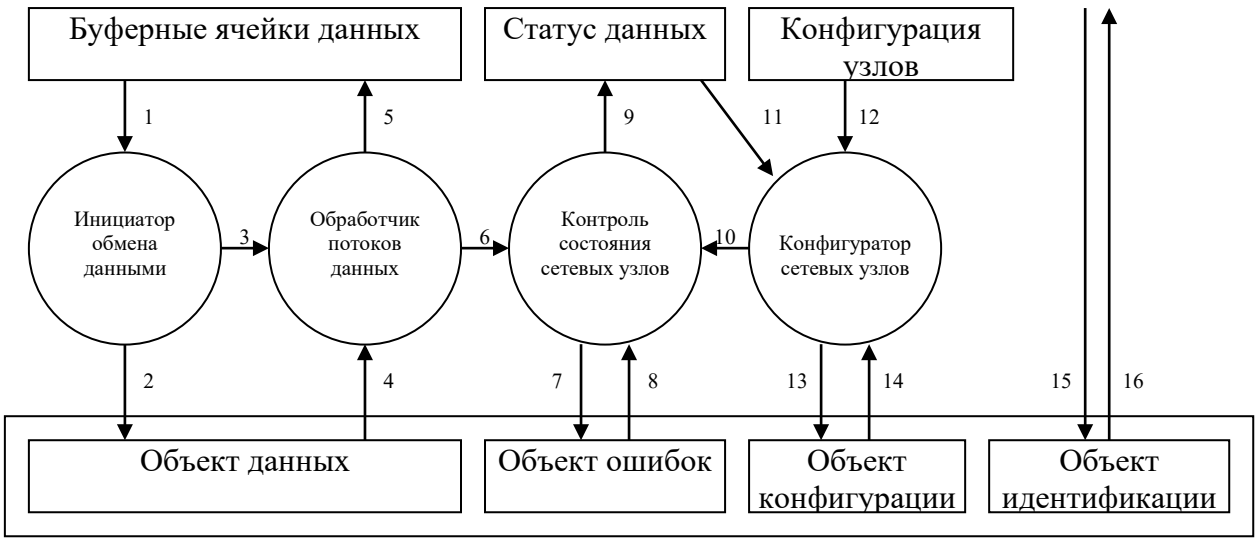
Обработка
При получении сообщения узел должен находиться в состоянии UNCONFIGURED и его активная сессия должна быть равна SES_ID. Узел вызывает функцию SsiLock(STATUS). После этого закрывает активную сессию. Если DATA[3] = 2, то узел остается в состоянии UNCONFIGURED. Если DATA[3] = 3, то узел выполняет стандартную процедуру рестарта.
Ответ
Нет

4 Ведущий узел

В составе сети ведущий узел выполняет функции синхронизации сетевых процессов, инициирует ввод/вывод данных, контролирует целостность сети, определяет режимы работы ведомых узлов, выполняет прикладной алгоритм целевой задачи. В общем случае система поддерживает несколько ведущих узлов, работающих независимо.

4.1 Модель ведущего узла

Структура сети не накладывает особых ограничений на организацию прикладной задачи ведущего узла. Для того чтобы решать возложенные на ведущий узел задачи сетевая подсистема должна поддержать работу объектов ошибок, данных и конфигурации. Объект идентификации может быть не реализован.
Основным процессом обработки данных является *обработчик потоков данных*. Его задачей является отслеживание асинхронных сообщений объекту данных (4), приходящих из сети. Обрабатывая полученные сообщения, обработчик поддерживает синхронность буферных ячеек данных ведущего узла системы (5). Использование только асинхронной связи с ведомыми узлами достаточно для отслеживания состояния всей системы.
В некоторых случаях ведущий узел может запрашивать данные узла или передавать собственные. Эту задачу решает *инициатор обмена данными*. По сигналу прикладной системы инициатор создает сообщение, заполняет поле данных необходимой информацией (1) и отправляет сообщение в сеть (2). О факте отправки сообщения сообщается (3) обработчику потоков данных, который запускает процесс отслеживания таймаутов фазы обмена сообщениями. О фактах нарушения тех или иных таймаутов сообщается (6) процессу контроля состояния сетевых узлов, который изменит состояния ячеек данных или узлов и инициирует процесс согласования ошибок.



Процесс *контроля состояния сетевых узлов* предоставляет ведущему узлу информацию о состоянии всех узлов системы и их элементов ввода/вывода (9). Процесс реализует стандартный асинхронный обмен сообщениями для объекта ошибок (7,8). Самостоятельный опрос статуса узлов процесс может начать по сигналу конфигуратора (10), что происходит при конфигурации очередного узла.

Конфигуратор сетевых узлов отслеживает соответствие текущих конфигураций узлов и конфигураций, заданных в специальной базе (12). Эта задача решается с помощью стандартного обмена сообщениями для объекта конфигурации (13,14). Еще одной задачей для конфигуратора является поиск в сети новых устройств (узлов) и попытка восстановить связь с узлами, которая была потеряна (11).
Объект идентификации может не поддерживаться сетевой системой и обмен сообщениями осуществляется из прикладной задачи (15,16).

4.2 Обработка сообщений ведущим узлом

Сообщения для каждого типа объектов всегда обрабатываются ведущим узлом одинаковым образом, независимо от порядка следования и результатов обработки предыдущих сообщений. Порядок обработки сообщения для определенного типа объекта в большинстве случаев определяется типом сообщения (REQ или ACK). Общие правила обработки сообщений представлены в таблице. Это необходимые действия ведущего узла. Помимо этих действий узел вправе производить любые другие операции, если они не нарушат описанную логику работы сети.

Тип сообщения	Реакция узла в зависимости от типа объекта			
	Ошибки	Данные	Конфиг. (данные совпали)	Конфиг. (данные не совпали)
REQ	Отправить ACK-сообщение с полученными статусами элементов ввода/вывода.	Получение (изменение) входных данных узла. Изменить соответствующие буферные ячейки.	Отослать ACK-сообщение объекту конфигурации.	Пометить соответствующий узел как несконфигурированный. Отослать REQ-сообщение объекту конфигурации.
ACK	Пометить в карте соответствующего узла состояние его элементов ввода/вывода. Состояние получено в сообщении	Получение (изменение) выходных данных узла. Изменить соответствующие буферные ячейки.	Пометить соответствующий узел как сконфигурированный. Отослать ACK-сообщение объекту ошибок.	

Сравнение данных конфигурационных сообщений происходит с данными, расположенными в базе данных конфигураций узлов ведущего узла. Данные сравниваются побитно, с учетом размера поля данных сообщения.

4.2.1 Сообщения объекту ошибок

Порядок обмена сообщениями
Сообщениями объекту ошибок имеет смысл обмениваться с узлами в сконфигурированном состоянии. После конфигурации узла, ведущий узел отправляет ACK-сообщение объекту ошибок, что запускает процесс согласования ошибок.

REQ-сообщение
Получение REQ-сообщения говорит о том, что состояние ведомого узла изменилось и его необходимо подтвердить. Подтверждается состоянием отправкой ACK-сообщения объекту ошибок соответствующего узла.

ACK-сообщение
Получение ACK-сообщения говорит о том, что ведомый узел согласовал состояние элементов ввода/вывода. Ведущий узел помечает в специальной карте новые состояния элементов ввода/вывода узла.

4.2.2 Сообщения объекту данных

Порядок обмена сообщениями
Ведущий узел должен обрабатывать сообщения объекту данных только от узлов, которые в его картах помечены как сконфигурированные.
Данные буферных ячеек ввода/вывода обновляются, только если соответствующий узел помечен как сконфигурированный и статус соответствующего элемента ввода/вывода рабочий.

REQ-сообщение
Получение REQ-сообщения говорит о том, что у ведомого узла изменились входные данные соответствующего элемента ввода/вывода. Ведущий узел должен обновить соответствующие буферные ячейки ввода/вывода.

ACK-сообщение
Получение REQ-сообщения говорит о том, что у ведомого узла изменились выходные данные соответствующего элемента ввода/вывода. Ведущий узел должен обновить соответствующие буферные ячейки ввода/вывода.

4.2.3 Сообщения объекту конфигурации

Порядок обмена сообщениями
Сообщение может быть получено в любой момент от любого узла. После конфигурации узла (получение ACK-сообщения с корректными данными), ведущий узел отправляет ACK-сообщение объекту ошибок, что запускает процесс согласования ошибок.
В случае необходимости, сетевая система может экспортировать специальный интерфейс конфигурирования индивидуальных узлов в процессе функционирования сети. Прикладная задача обновляет запись в базе данных и вызывает сетевой сервис, который просто посылает пустое REQ-сообщение объекту конфигурации выбранного узла. Это сообщение запускает процесс конфигурирования узла в новый режим работы.

Сравнение поля данных сообщения
Сравнение поля данных сообщения необходимо для того, чтобы убедиться в корректности воспринятой узлом конфигурации. Сравнение производится побитно с учетом длины поля данных. Образцовая конфигурация определяется прикладной задачей и располагается в базе данных ведущего узла.

REQ-сообщение
Получение REQ-сообщения говорит о том, что конфигурация ведомого узла не совпадает (изменилась) с предполагаемой. Ведущему узлу необходимо немедленно пометить узел как несконфигурированный. При совпадении полученной конфигурации с указанной в базе данных отправляется ACK-сообщение с подтверждением. При несовпадении отправляется REQ-сообщение с конфигурационными данными из базы данных ведущего узла.

ACK-сообщение
Получение ACK-сообщения говорит о том, что конфигурация ведомого узла была применена и узел работает в новом режиме. При совпадении полученной конфигурации с указанной в базе данных узел помечается как сконфигурированный и отправляется ACK-сообщение объекту ошибок. При несовпадении узел помечается как несконфигурированный и отправляется REQ-сообщение с конфигурационными данными из базы данных ведущего узла.

4.2.4 Сообщения объекту идентификации

Никаких особых требований к реализации обмена сообщениями объекта идентификации для ведущего узла не выдвигается. Ведущий узел может не поддерживать объект идентификации вообще.
Может быть реализован ведущий узел, который поддерживает только объект идентификации. Такой узел может быть использован параллельно и независимо с основным, для удаленного контроля конфигурации сети (определение идентификаторов, выяснение работающих узлов, перезагрузка контроллеров и т.д.).
Описание сообщений объекту идентификации см. 3.7 Объект идентификации.

4.3 Инициализация ведущего узла

При старте ведущего узла необходимо выяснить текущее состояние всех узлов в сети, не нарушая ее работоспособность. Для решения указанной задачи и поддержания работоспособности сети ведущий узел должен следовать трем правилам:

1. При старте узла пометить все узлы сети как не ответившие и статус всех их элементов ввода/вывода установить в ошибку;
2. После инициализации всех внутренних служб отослать в сеть широковещательное пустое ACK-сообщение объекту конфигурации;
3. При получении ACK-сообщения от объекта конфигурации любого узла, послать пустое ACK-сообщение объекту ошибок этого узла.

Выполнение этих правил запускает соответствующие сетевые механизмы, которые обеспечивают целостность сетевого обмена и достоверность передачи информации.

Ниже описана процедура старта системы при включении ведущего узла. Правила обработки сообщений см. 4.2 Обработка сообщений ведущим узлом и 5.3 Обработка сообщений ведомым узлом. К моменту старта в сети функционировало 2 логических узла A и B. Конфигурация узла A CFGa совпадает с требуемой, узла B CFGb` – нет. Реальный порядок следования сообщений может отличаться, но это ни на что не повлияет, т.к. после отправки ведущим узлом широковещательного ACK-сообщения объекту конфигурации процессы в сети для узлов A и B протекают совершенно независимо.

1. На ведущий узел подано питание, узел инициализирует все свои подсистемы.
2. (Правило 1) Все узлы помечаются как несконфигурированные, все элементы ввода/вывода в состоянии ошибки. Никакие сообщения, кроме сообщений конфигурационных объектов, от узлов не обрабатываются.
3. (Правило 2) Ведущий узел отсылает в сеть широковещательное пустое ACK-сообщение объекту конфигурации.
4. Узлы A и B принимают сообщение. Так как сообщение пустое, сравнение конфигурации не проходит, и оба узла отвечают REQ-сообщениями с текущими конфигурациями. Узел A отвечает

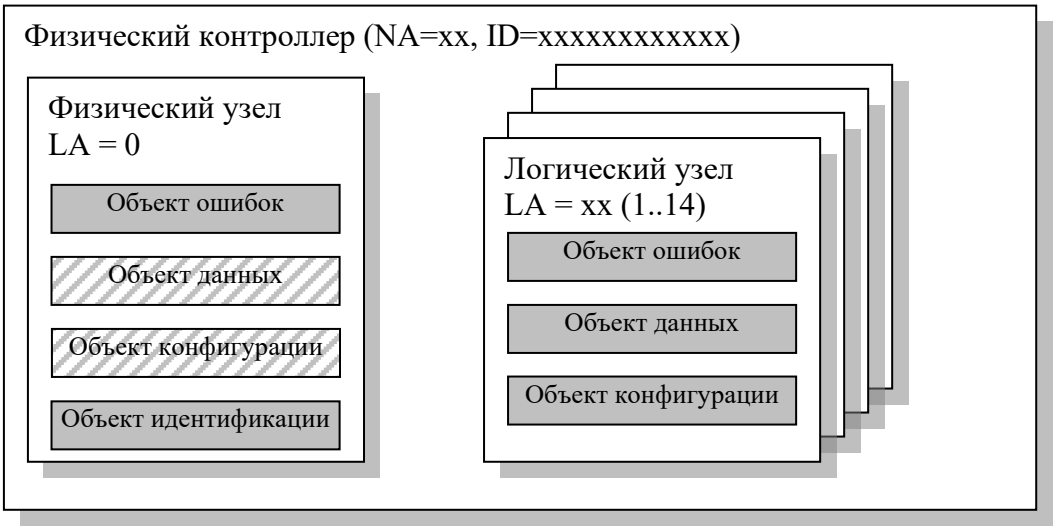
REQ(CFGa), узел B отвечает REQ(CFGb`). Конфигурация CFGb` не совпадает с конфигурацией CFGb, хранящейся в базе данных для узла B.

5. Ведущий узел получает сообщение REQ(CFGa). Так как получено REQ-сообщение объекту конфигурации, узел A помечается как несконфигурированный (собственно он таким и был) и конфигурация подтверждается. Ведущий узел отправляет сообщение ACK(CFGa).
6. Ведущий узел получает сообщение REQ(CFGb`). Так как получено REQ-сообщение объекту конфигурации, узел B помечается как несконфигурированный (собственно он таким и был). Кроме того, полученная конфигурация узла не совпадает с указанной в базе данных для этого узла, поэтому ведущий узел отправляет обратно REQ(CFGb).
7. Узел A получает сообщение ACK(CFGa). Конфигурация подтверждена, узел остается сконфигурированным и отвечает сообщением ACK(CFGa).
8. Узел B получает сообщение REQ(CFGb). Полученная конфигурация не совпадает с текущей. Узел переходит в состояние UNCONFIGURED и отвечает сообщением REQ(CFGb).
9. (Правило 3) Ведущий узел получает сообщение ACK(CFGa). Узел помечается как сконфигурированный и отсылается пустое ACK-сообщение объекту ошибок узла A.
10. Ведущий узел получает сообщение REQ(CFGb). Так как получено REQ-сообщение объекту конфигурации, узел B помечается как несконфигурированный (собственно он таким и был) и конфигурация подтверждается. Ведущий узел отправляет сообщение ACK(CFGb).
11. Узел A получает пустое ACK-сообщение для объекта ошибок. Т.к. сообщение пустое, то статусы не совпадают, и узел отвечает сообщением REQ(ERRa).
12. Узел B получает сообщение ACK(CFGb). Конфигурация подтверждена, узел переходит в состояние ONLINE или STALL и отвечает сообщением ACK(CFGb).
13. Ведущий узел получает сообщение REQ(ERRa). Обратно отсылается сообщение ACK(ERRa).
14. (Правило 3) Ведущий узел получает сообщение ACK(CFGb). Узел помечается как сконфигурированный и отсылается пустое ACK-сообщение объекту ошибок узла B.
15. Узел A получает ACK(ERRa). Статусы элементов ввода/вывода совпадают, поэтому узел переходит (остается) в состояние ONLINE. Ведущему узлу отсылается сообщение ACK(ERRa).
16. Узел B получает пустое ACK-сообщение для объекта ошибок. Т.к. сообщение пустое, то статусы не совпадают, и узел отвечает сообщением REQ(ERRb).
17. Ведущий узел получает сообщение ACK(ERRa). В карте помечаются неработающие элементы ввода/вывода узла A. С этого момента с узлом A возможен нормальный обмен данными. Узел A зарегистрирован в сети, без нарушения его работы.
18. Ведущий узел получает сообщение REQ(ERRb). Обратно отсылается сообщение ACK(ERRb).
19. Узел B получает ACK(ERRb). Статусы элементов ввода/вывода совпадают, поэтому узел переходит (остается) в состояние ONLINE. Ведущему узлу отсылается сообщение ACK(ERRb).
20. Ведущий узел получает сообщение ACK(ERRb). В карте помечаются неработающие элементы ввода/вывода узла B. С этого момента с узлом B возможен нормальный обмен данными. Узел B зарегистрирован в сети, он был сконфигурирован в требуемый режим работы и синхронизированы состояний элементов ввода/вывода.

5 Ведомый узел

В составе сети ведомый узел выполняет функции удаленного ввода/вывода данных. В зависимости от реализованных типов объектов функциональность отдельных узлов сети может отличаться друг от друга.

5.1 Модель ведомого узла



В общем случае узел ввода/вывода может обеспечивать большое количество конфигураций оконечных устройств и сочетаний каналов ввода/вывода данных. Физически узел удаленного ввода вывода представляет собой контроллер с интерфейсом CAN для подключения к сети и произвольным набором других интерфейсов, через которые осуществляется ввод/вывод данных.

Физический контроллер, который представляет собой физический узел, обладает сетевым (физическим) адресом и уникальным идентификатором. Его задачи обрабатывать сбойные ситуации, координировать работу всех каналов ввода/вывода, в случае необходимости обеспечивать доступ к физическим ресурсам контроллера (обновление программного обеспечение контроллера, программирование энергонезависимой памяти, ведение различных журналов, контроль температуры, питания и корректности функционирования оборудования, калибровка каналов ввода/вывода и т.д.). Физический узел имеет логический адрес LA=0. Обычно физический узел реализует объект ошибок (см. раздел 3.3 Объект ошибок) и объект идентификации (см. раздел 3.7 Объект идентификации). Объекты конфигурации и данных могут быть реализованы специфическим для контроллера образом и в настоящее время не регламентированы. В каждом физическом контроллере должен быть физический узел (LA=0). Каждый физический контроллер кроме физического узла может содержать до 14 логических узлов.

Физический контроллер с адресом A виден в сети как объекты физического узла (NA=A,LA=0) и объекты нескольких логических узлов (NA=A,LA=x, где x=1..14 или 15 для группового доступа ко всем логическим узлам физического контроллера). Реализация физического узла тесно связана с особенностями физического контроллера. Логические узлы обеспечивают унифицированный ввод/вывод данных, не акцентируя внимания на особенностях реализации тех или иных каналов ввода/вывода данных. Каждый канал ввода/вывода данных в логическом контроллере характеризуется шириной слова данных и

размером конфигурационной информации. Примером может служить аналоговый вход (10 бит слово данных, 3 бита конфигурации [количество усреднений]) или дискретный выход (1 бит слово данных, 2 бита конфигурации [режим и полярность]). Такой канал ввода/вывода называется элементом ввода/вывода логического контроллера и именно с этими элементами идет работа при обмене сообщениями с логическими узлами сети. Каждый логический контроллер может поддерживать до 64 элементов ввода вывода (ограничение связано с размером поля данных сообщения в сети).
Разработчик прикладного программного обеспечения устройства имеет полную свободу в распределении ресурсов физического контроллера между логическими узлами в его составе. Единственным ограничением служит то, что для реализуемых объектов всех узлов поле данных сообщения не должно превышать 64 бита.

5.2 Состояния ведомого узла

Каждый узел (логический или физический) в процессе функционирования изменяет свое состояние с точки зрения сети. Состоянием узла в текущий момент определяется его поведение при обработке сообщений, количество и состояние активных логических узлов, перечень доступных объектов в физическом и логических узлах и некоторые другие параметры.

Для узла определены следующие состояния:

Название	Описание
UNDEFINED	Неопределенное состояние узла. Индивидуальный узел недоступен и для сети не существует. Физический узел ожидает установки сетевого адреса. Обрабатывает ТОЛЬКО широковещательные сообщения для физического узла (NA=0x7F,LA=0).
UNCONFIGURED	Конфигурация узла не определена. Логический узел виден в сети, но для дальнейшей работы его нужно сконфигурировать. Физический узел доступен полностью.
ONLINE	Узел полностью работоспособен. Доступны все его ресурсы.
STALL	Функционирование узла приостановлено по причине сбоя. Необходимо синхронизировать (подтвердить) состояние. В логических узлах недоступны объекты данных.
POWER	Узел обнаружил ошибку питания и временно для сети недоступен.

Состояния логических узлов зависят от состояния физического узла. Весь старт контроллера происходит сначала через инициализацию физического узла, после чего инициализируются логические узлы. Допустимые комбинации состояний приведены в таблице (если строка таблицы заполнена минусами, это значит, физический узел не может находиться в таком состоянии):

Состояние физического узла		Возможное состояние логического узла				
		1	**2**	**3**	**4**	**5**
UNDEFINED	**1**	X				
UNCONFIGURED	**2**	-	-	-	-	-
ONLINE	**3**		X	X	X	
STALL	**4**	-	-	-	-	-
POWER	**5**	X				

В определенных состояниях узла становятся недоступны некоторые типы объектов этого узла. Этот процесс называется “маскированием” объектов. Сообщения предназначенные “замаскированным” объектам “замораживаются” на время “маскирования” объекта. Они не обрабатываются и не передаются в сеть, даже если все остальные сообщения обработаны и переданы. В “замороженном” состоянии все необходимые временные интервалы для сообщений отслеживаются (см. раздел 2.1 Фазы обмена сообщениями) и при устаревании сообщения оно уничтожается.
Соответствие состояний физического узла и доступных типов объекта приведено в таблице:

Состояние физического узла	Тип объекта			
	Ошибки	Данные	Конфиг.	Идентиф.
UNDEFINED	*	*	*	*
ONLINE	+	+	+	+
POWER	-	-	-	-

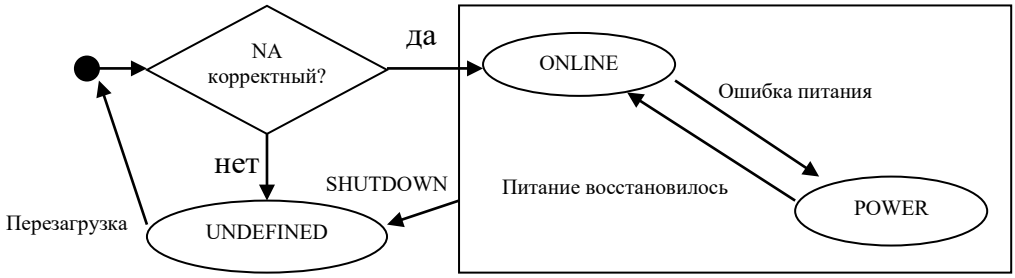
* - объект доступен только с помощью широковещательного сообщения

Соответствие состояний логического узла и доступных типов объекта приведено в таблице:

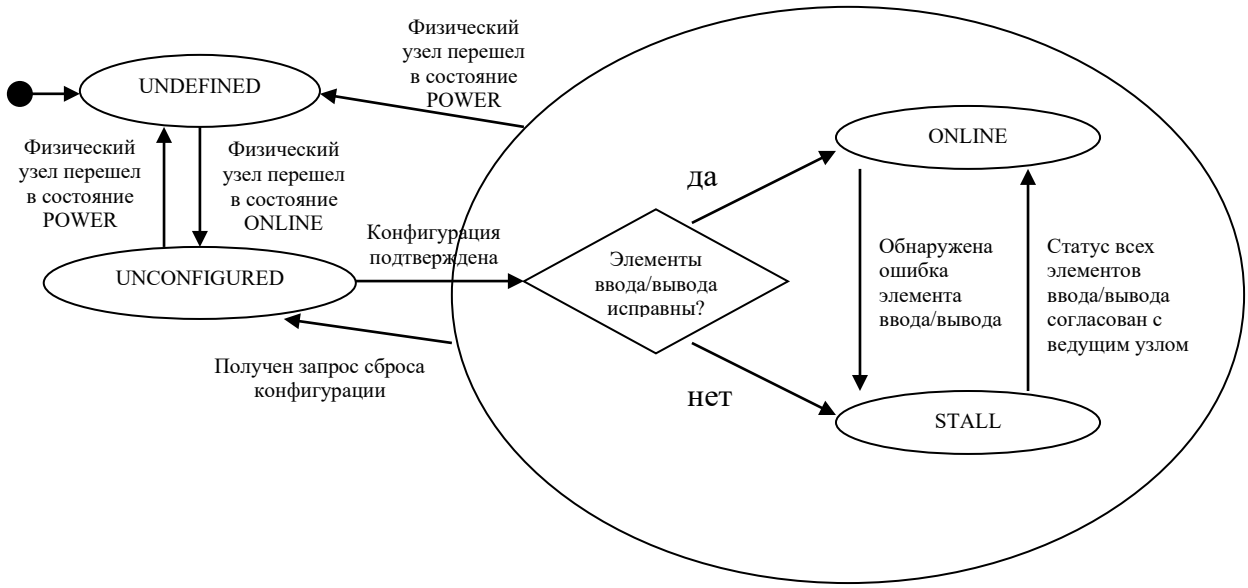
Состояние логического узла	Тип объекта		
	Ошибки	Данные	Конфиг.
UNDEFINED	-	-	-
UNCONFIGURED	-	-	+
ONLINE	+	+	+
STALL	+	-	+

Смена состояний узла определяется конечным автоматом. Смена состояний физическим узлом ведет к смене состояний логическими узлами, но не наоборот.

Смена состояний физического узла



Смена состояний логического узла



5.3 Обработка сообщений ведомым узлом

Тип сообщения	Результат сравнения аргументов	Реакция узла в зависимости от типа объекта			
		Ошибки	Данные	Конфиг.	Идентиф.
REQ	совпадают	отправить ACK сообщение ведущему узлу	**(запись данных)** Вывод переданных данных на устройства, возвращение состояния выходов (ACK)	отправить ACK сообщение ведущему узлу	См. раздел 3.7 Объект идентификации
	не совпадают	отправить REQ сообщение с реальным статусом ведущему узлу, перейти в состояние STALL		сохранить переданную конфигурацию, отправить REQ сообщение с конфигурацией ведущему узлу, перейти в состояние UNCONFIGURED	
ACK	совпадают	отправить ACK сообщение ведущему узлу, перейти в состояние ONLINE	**(чтение данных)** Считывание запрошенных данных, возвращение состояния входов (REQ)	отправить ACK сообщение ведущему узлу, перейти в состояние ONLINE или STALL в зависимости от статуса элементов ввода/вывода	
	не совпадают	отправить REQ сообщение с реальным статусом ведущему узлу		отправить REQ сообщение с текущей конфигурацией ведущему узлу	

Сообщения для каждого типа объектов всегда обрабатываются ведомым узлом одинаковым образом, независимо от порядка следования и результатов обработки предыдущих сообщений. Порядок обработки сообщения для определенного типа объекта в большинстве случаев определяется типом сообщения (REQ или ACK). Общие правила обработки сообщений представлены в таблице. Это необходимые действия ведомого узла. Помимо этих действий узел вправе производить любые другие операции, если они не нарушат описанную логику работы сети.
Столбик “Результат сравнения аргументов” показывает результат сравнения данных, полученных в сообщении и текущего состояния узла. Способ сравнения и сами локальные данные зависят от типа объекта и подробнее описаны ниже.

5.3.1 Сообщения объекту ошибок

Сообщения могут быть посланы как физическому, так и логическому узлу.
Задача ведомого узла с помощью сообщений данному объекту обеспечить синхронность статуса элементов ввода/вывода логических узлов. В случае возникновения ошибки или несовпадении статуса элементов узла и представления ведущего узла о статусе этих элементов узел переходит в состояние STALL и обмен данными “замораживается”. Этим обеспечивается то, что ведущий узел не получит неверных данных, а когда он начнет получать данные, он будет знать о том, что эти данные идут от неисправного элемента ввода/вывода.

REQ-сообщение
Получение REQ-сообщения говорит о том, что ведущий узел считает (уверен), что статусы элементов ввода/вывода соответствуют переданным в сообщении. Узел сравнивает текущие статусы элементов с полученными в сообщении. В случае совпадения узел отвечает ACK-сообщением с текущим статусом в поле данных, т.е. он подтверждает (согласен) мнение ведущего узла. При несовпадении текущего и полученного статусов элементов ввода/вывода узел переходит в состояние STALL (обмен данными “замораживается”) и отвечает REQ-сообщением с текущим (реальным) статусом в поле данных, т.е. он не согласен с ведущим узлом и указывает реальное состояние элементов ввода/вывода.

ACK-сообщение
Получение ACK-сообщения говорит о том, что ведущий узел уточняет (не уверен) состояние элементов ввода/вывода узла. Предполагаемое состояние элементов ввода/вывода передается в сообщении. Узел сравнивает текущие статусы элементов с полученными в сообщении. В случае совпадения узел отвечает ACK-сообщением с текущим статусом в поле данных, т.е. он подтверждает мнение ведущего узла. При несовпадении текущего и полученного статусов элементов ввода/вывода узел отвечает REQ-сообщением с текущим (реальным) статусом в поле данных, т.е. он не согласен с ведущим узлом и указывает реальное состояние элементов ввода/вывода. При этом в отличие от обработки REQ-сообщения переход в состояние STALL не производится.

Сравнение поля данных сообщения
При сравнении текущего и переданного в сообщении статусов необходимо проверять значение поля DLC в сообщении. Т.е. поле данных должно содержать достаточное количество битов, чтобы передать состояние всех элементов ввода/вывода. Если поле DLC содержит неверное значение (полученных данных недостаточно, чтобы полностью сравнить состояние), узел считает, что статусы не совпадают.

Специальные возможности
Ведущий узел при необходимости может использовать сообщение для объекта ошибок с данными нулевой длины (поле DLC=0). Т.к. все узлы имеют хотя бы один элемент ввода/вывода, то для всех узлов в сообщении объекту ошибок DLC>0. Для всех узлов такое сообщение не пройдет сравнение, и они ответят REQ-сообщением с их реальным статусом элементов ввода/вывода. При этом если сообщение, присланное ведущим узлом, было REQ-сообщением, то все узлы перейдут в состояние STALL.

Асинхронные сообщения
Если ведомый узел обнаружил изменение статуса элемента ввода/вывода, он должен сообщить об этом ведущему узлу. Для этого есть 2 способа:

- переходит в состояние STALL и отправляет ведущему узлу REQ-сообщение (этот способ предпочтительней);
- отправляет ведущему узлу ACK-сообщение.

5.3.2 Сообщения объекту данных

Сообщения могут быть посланы как физическому, так и логическому узлу.
Ведомый узел с помощью сообщений данному объекту обеспечивает ввод/вывод данных. Процессы ввода/вывода данных могут полностью контролироваться ведущим узлом, а могут проводиться с использованием локальных алгоритмов ведомого узла с уведомлением ведущего узла об изменениях. Хотя основным режимом работы данного объекта является обработка сообщений ведущего узла, имеется возможность асинхронного сообщения об изменении данных. Обработка ошибок в процессе обмена данными производится независимо и обеспечивается более высоким приоритетом объекта ошибок и наличием состояния STALL.

REQ-сообщение
Получение REQ-сообщения говорит о том, что ведущий узел производит запись данных в элементы ввода/вывода узла. Узел декодирует данные, упакованные в поле данных сообщения, производит необходимые действия и возвращает ACK-сообщение с выведенными данными. В большинстве случаев эти данные совпадают с переданными ведущим узлом, но могут быть ситуации (выполнение локальных алгоритмов, специальные режимы работы элементов ввода/вывод узла), когда эти данные не совпадают. Контроль совпадения данных осуществляется на прикладном уровне и выходит за рамки сети.

ACK-сообщение
Получение ACK-сообщения говорит о том, что ведущий узел производит чтение данных узла. Узел производит необходимые действия для получения запрошенных данных, формирует ответное сообщение и отправляет REQ-сообщение ведущему узлу. В поле данных ответного сообщения упакованы данные запрошенных элементов ввода/вывода узла.

Сравнение поля данных сообщения
Поле данных ACK-сообщения игнорируется (за исключением объекта групповых данных, в котором передается карта элементов ввода/вывода).
Для REQ-сообщения поле данных должно содержать достаточное количество битов, чтобы передать данные для всех элементов ввода/вывода узла.
При несовпадении длины поля данных, сообщение игнорируется.

Специальные возможности
Ведущий узел имеет возможность отправить широковещательное ACK-сообщение объекту общих данных с DLC = 0. В ответ все узлы передадут REQ-сообщение с данными элементов ввода/вывода.
Элементы ввода/вывода типа OUT должны обрабатывать сообщения REQ и могут обрабатывать сообщения ACK.
Элементы ввода/вывода типа IN должны обрабатывать сообщения ACK и могут обрабатывать сообщения REQ.

Асинхронные сообщения
Если узел обладает возможностью самостоятельно осуществлять ввод/вывод данных, т.е. данные элементов ввода/вывода изменяются без прямого управления ведущего узла, бывает необходимым сообщать ведущему узлу о таких изменениях. Ведомый узел имеет возможность сообщать о таких изменениях, отправляя ведущему узлу асинхронное сообщение объекту данных. В случае изменения входных данных узел посылает REQ сообщение (второе сообщение фазы чтения данных). В случае изменения выходных данных узел посылает ACK сообщение (второе сообщение фазы записи данных). Не все узлы и не все элементы в узле могут поддерживать такой режим работы. Необходимость такого режима работы определяется прикладной задачей.

5.3.3 Сообщения объекту конфигурации

Сообщения могут быть логическому узлу.
Ведомый узел с помощью сообщений данному объекту обеспечивает конфигурирование элементов ввода/вывод узла для работы в заданных режимах.
Для каждого элемента ввода/вывода конфигурационные данные имеют фиксированный размер, и формат этих данных полностью определяется прикладной задачей. Изменение режимов работы элементов ввода/вывода осуществляется одновременно для всего узла.
Работа объекта конфигурации происходит не с непосредственной конфигурацией элементов ввода/вывода, а с ее копией, хранящейся в ведомом узле. Только поле полной уверенности ведущего узла в правильности конфигурации выдается команда на применение конфигурации и конфигурационные данные копии применяются к элементам ввода/вывода.
Каждый элемент ввода/вывода должен иметь хотя бы один конфигурационный бит (пусть даже и неиспользуемый). Использование элементов ввода/вывода с нулевой длиной конфигурационных данных запрещено.

REQ-сообщение
Получение REQ-сообщения говорит о том, что ведущий узел задает (изменяет) режим работы элементов ввода/вывода ведомого узла. Узел сравнивает текущую конфигурацию (копию) с заданной в сообщении. В случае совпадения узел отвечает ACK-сообщением с текущей конфигурацией (копией). При несовпадении конфигурационных данных узел переходит в состояние UNCONFIGURED, обновляет копию конфигурации и отвечает REQ-сообщением с новой (копией) конфигурацией в поле данных. Обмен данными останавливается, до тех пор, пока узел не будет сконфигурирован. Конфигурация, которая сохранилась в копии) будет применена только по специальной команде.

ACK-сообщение

Получение ACK-сообщения говорит о том, что ведущий узел подтверждает конфигурационные данные, хранящиеся в копии в ведомом узле. Узел сравнивает текущую конфигурацию (копию) с данными, полученными в сообщении. В случае совпадения узел, сохраняет “применяет” копию конфигурационных данных (данные применяются для изменения режимов работы элементов ввода/вывода. Если в процессе смены режимов возникли ошибки, выставляются статусы соответствующих элементов ввода/вывода. В зависимости от статусов элементов ввода/вывода узел переходит в состояние ONLINE или STALL. Поле этого узел отвечает ACK-сообщением с текущими конфигурационными данными. При несовпадении текущей (копии) и полученной конфигурации узел отвечает REQ-сообщением с конфигурационными данными (копии) в поле данных. При этом в отличие от обработки REQ-сообщения переход в состояние UNCONFIGURED не производится.

В некоторых случаях, особенно для элементов ввода/вывода настроенных на выход, применить новую конфигурацию в момент получения ACK-сообщения невозможно. Это связано с тем, что при переводе элемента ввода/вывода в другой режим функционирования неизвестно, какое значение нужно выставить (а какое-то значение все равно будет на выходе присутствовать). Данную проблему можно решить двумя способами:

- В конфигурационных данных вместе с режимом функционирования и другими настройками передавать данные, которые необходимо выставить при переводе элемента ввода/вывода в этот режим. Прикладная система сама определяет каким способом применить эти данные и это не связано с обменом данными с объектом данных.
- Применять конфигурацию в момент получения первого сообщения объекту данных заново сконфигурированного элемента ввода/вывода. При обработке ACK-сообщения объекту конфигурации проверяется возможность такой конфигурации, готовность аппаратных блоков, непротиворечивость настроек и т.д. После чего выставляется внутренний флаг готовности, но элемент ввода/вывода не конфигурируется, хотя подтверждающее сообщение отправляется ведущему модулю. В момент получения сообщения объекту данных конфигурация применяется, и элемент ввода/вывода меняет режим функционирования. Если по каким-либо причинам конфигурацию применить оказывается невозможно всегда можно выставить ошибочный статус элемента ввода/вывода и перейти в состояние STALL, что заблокирует обмен данными с конкретным элементов ввода/вывода.

Второй способ применения конфигурации предпочтительней, т.к. обеспечивает большую гибкость и потенциально уменьшает размер конфигурационных пакетов. Но для такой реализации несколько сложнее выглядит прикладная задача ведомого узла. Выбор того или иного способа в конечном итоге зависит от решаемой задачи.

Сравнение поля данных сообщения

При несовпадении длины ACK-сообщения считается, что конфигурации не совпали.

При несовпадении длины REQ-сообщения выполняется специфическая функция “сброс конфигурации”. Узел, у которого таким образом конфигурация “сброшена” (уничтожена копия конфигурационных данных) также переходит в состояние UNCONFIGURED, но копии конфигурационных данных не существует, поэтому все сравнения будут неуспешными, кроме сравнений с конфигурационными данными нулевой длины. Новую конфигурацию необходимо загрузить новым REQ-сообщением. Узел со сброшенной конфигурацией отвечает пустым сообщением объекту конфигурации.

Специальные возможности
Ведущий узел имеет возможность отправить широковещательное пустое ACK-сообщение объекту конфигурации. В ответ все узлы передадут REQ-сообщение с их текущими конфигурационными данными (текущие конфигурации). При этом узлы не изменят режима своего функционирования.
Сбросить конфигурацию любого узла (или всех узлов сети) можно послав пустое REQ-сообщение объекту конфигурации.

Асинхронные сообщения
Ведомый узел может по той или иной причине самостоятельно изменить конфигурацию элементов ввода/вывода. Для уведомления об этом ведущего узла используется асинхронное сообщение объекту конфигурации.
Самостоятельная смена конфигурации равносильна получению REQ-сообщения с новой конфигурацией. Поведение узла в этом случае идентично реакции на получение такого сообщения: переход в состояние UNCONFIGURED и отправка REQ-сообщения или отправка ACK-сообщения.
Самостоятельная смена конфигурации может быть вызвана прикладной задачей ведомого узла при переходе в аварийный режим работы, рестарте контроллера, действий пользователя и т.д. В любом случае, если ведомый узел обладает способностью самостоятельной смены конфигурации, о таких событиях ведомый узел должен сообщать ведущему узлу.

5.3.4 Сообщения объекту идентификации

Сообщения могут быть посланы физическому узлу.
Описание сообщений объекту идентификации см. 3.7 Объект идентификации.

5.4 Инициализация ведомого узла

Порядок смены состояний узла см. в разделе 5.2 Состояния ведомого узла.
После инициализации физического узла и старте логических узлов контроллера, каждый из них должен послать ведущему узлу уведомление о самостоятельной смене конфигурации. Подробнее см. 5.3.3 Сообщения объекту конфигурации.

Ниже описана процедура старта ведомого узла X при его “горячем” подключении к системе. Правила обработки сообщений см. 4.2 Обработка сообщений ведущим узлом и 5.3 Обработка сообщений ведомым узлом. К моменту старта сеть полностью функционально и в ней присутствует ведущий узел. Предполагается, что конфигурация по умолчанию узла X не совпадает с требуемой. Возможно, ведущий узел уже определил, что узел X отсутствует в сети, т.к. от него давно не было ответов, но это не принципиально.

1. На узел X подано питание, узел инициализирует все свои подсистемы.
2. Узел X переходит в состояние UNCONFIGURED.
3. Узлу X только что включили питание и предыдущая конфигурация недоступна, считается, что конфигурации не совпали и ведущему узлу отправляется сообщение REQ(CFGx`).
4. Ведущий узел получает сообщение REQ(CFGx`). Так как получено REQ-сообщение объекту конфигурации, узел X помечается как несконфигурированный (поэтому было сказано, что неважно каким он был с самого начала). Кроме того, полученная конфигурация узла не совпадает с указанной в базе данных для этого узла, поэтому ведущий узел отправляет обратно REQ(CFGx).
5. Узел X получает сообщение REQ(CFGx). Полученная конфигурация не совпадает с текущей. Узел остается в состоянии UNCONFIGURED и отвечает сообщением REQ(CFGx).

6. Ведущий узел получает сообщение REQ(CFGx). Так как получено REQ-сообщение объекту конфигурации, узел X помечается как несконфигурированный (собственно он таким и был) и конфигурация подтверждается. Ведущий узел отправляет сообщение ACK(CFGx).
7. Узел X получает сообщение ACK(CFGx). Конфигурация подтверждена, узел переходит в состояние ONLINE или STALL и отвечает сообщением ACK(CFGx).
8. (Правило 3) Ведущий узел получает сообщение ACK(CFGx). Узел помечается как сконфигурированный и отсылается пустое ACK-сообщение объекту ошибок узла X.
9. Узел X получает пустое ACK-сообщение для объекта ошибок. Т.к. сообщение пустое, то статусы не совпадают, и узел отвечает сообщением REQ(ERRx).
10. Ведущий узел получает сообщение REQ(ERRx). Обратно отсылается сообщение ACK(ERRx).
11. Узел X получает ACK(ERRx). Статусы элементов ввода/вывода совпадают, поэтому узел переходит (остается) в состояние ONLINE. Ведущему узлу отсылается сообщение ACK(ERRx).
12. Ведущий узел получает сообщение ACK(ERRx). В карте помечаются неработающие элементы ввода/вывода узла X. С этого момента с узлом X возможен нормальный обмен данными. Узел X зарегистрирован в сети, он был сконфигурирован в требуемый режим работы и синхронизированы состояний элементов ввода/вывода.

6 Виды сетевых узлов проекта КТЖ-2

В составе сети проекта КТЖ-2 используются 3 вида контроллеров: панель управления MCN-3, модуль аналогового ввода/вывода MA444, модуль дискретного ввода/вывода MD846. На базе контроллера MCN-3 реализуется ведущий узел. Контроллеры MA444 и MD846 обеспечивают распределенный ввод/вывод данных и на них реализованы ведомые узлы.
CAN сеть в системе КТЖ-2 работает на скорости 50 кбит/с.

SYNC_SEG (1)	PROP_SEG PHASE_SEG1 (6)	PHASE_SEG2 (3)

6.1 Идентификация периферийных контроллеров

В системе КТЖ-2 приняты определенные правила идентификации контроллеров. Для решения задачи идентификации используется стандартный объект идентификации физического узла. Тип устройства идентифицируется специальными полями, расположенными в серийном номере.

Поле	Название	Назначение
ID[1..0]	serial number (SN) серийный номер	Уникальный номер конкретного типа устройства. Отображается на корпусе и может быть изменен в энергонезависимой памяти контроллера
ID[2]	vendor ID (VID) идентификатор производителя	Зарегистрированный номер производителя контроллера. Поле сравнивается ВСЕГДА, независимо от значения бита i данных объекта идентификации. Поле отображается на корпусе устройства. Данные для сравнения определяются программным обеспечением контроллера и не могут быть изменены. LMT = 0
ID[4..3]	product ID (PID) идентификатор продукта	Зарегистрированный номер типа устройства. Поле сравнивается ВСЕГДА, независимо от значения бита i данных объекта идентификации. Поле отображается на корпусе устройства. Данные для сравнения определяются программным обеспечением контроллера и не могут быть изменены. MD846 = 0x0112 MD846R2,MD846R3 = 0x0212 MA444 = 0x0113 MA444R2,MA444R3 = 0x0213 MA004,MA444R3 = 0x012E MD808R1 = 0x169 MD808R2 = 0x269 MD808R3 = 0x369 MA844R1 = 0x173
ID[5]	revision (REV) версия	Ревизия программного обеспечения периферийного контроллера. Поле не сравнивается НИКОГДА,

	программного обеспечения	независимо от значения бита i данных объекта идентификации. Поле, указанное на корпусе контроллера не используется и не должно влиять на функциональность системы в целом. Поле не может быть изменено и определяется программным обеспечением контроллера.

6.2 Контроллер MCN-3

Ведущий узел на MCN-3 существует в двух конфигурациях. В одной из конфигураций узел реализует только объект идентификации, в другой объекты ошибок, общих данных и конфигурации.
Ведущий узел контроллера MCN-3 не полностью соответствует описанной спецификации сети, а именно он **не поддерживает**:

- объект групповых данных;
- обработку асинхронных сообщений объекту общих данных;
- работу с объектом данных, если в состав его данных входят элементы с разными правами доступа.

Конфигурация сети, типы контроллеров, декомпозиция на физические и логические узлы описываются в специальном конфигурационном файле.
Общий формат описания контроллера (физический узел и логические узлы):

```
CONTROLLER_NAME : CONTROLLER
     IO_NAME1 : [CFG_SIZE,CFG_VALUE] AT %[IQM][XBW]LA.INDEX
     IO_NAME2 : [SIZE,CFG_SIZE,CFG_VALUE] AT %[IQM]LA.INDEX
END_CONTROLLER
```

CONTROLLER_NAME	Уникальное в рамках всего проекта имя контроллера. Выбирается разработчиком.
IO_NAME1, IO_NAME2	Уникальное в рамках контроллера название канала ввода/вывода. Выбирается разработчиком.
SIZE	Размер данных канала в битах (не больше 16 бит)
CFG_SIZE	Размер конфигурационных данных в битах
CFG_VALUE	Значение конфигурации по умолчанию
I,Q,M	I-ввод, Q-вывод, M-ввод/вывод
X,B,W	X-бит,B-байт(8бит),W-слово(16бит)
LA	Адрес логического контроллера, которому принадлежит описываемый канал ввода/вывода
INDEX	Порядковый номер описываемого канала ввода/вывода в логическом узле. Начинается с 1.

6.3 Контроллер MA444 / MA444R2 / MA444R3 / MA004 / MA004R3

Контроллер MA444 обладает следующими возможностями ввода/вывода:

- 4 аналоговых входа, 16 бит
- 4 входа температурных датчиков DS18B20, 16 бит
- 4 аналоговых выхода, 16 бит

Контроллер MA444R2/MA444R3 обладает следующими возможностями ввода/вывода:

- 8 аналоговых входов, 16 бит
- 4 входа температурных датчиков DS18B20, 16 бит
- 4 аналоговых выхода, 16 бит

Контроллер MA004/MA004R3 обладает следующими возможностями ввода/вывода:

- 4 входа температурных датчиков DS18B20, 16 бит

Контроллер MA844R1 обладает следующими возможностями ввода/вывода:

- 8 аналоговых входов, 12 бит
- 4 входа температурных датчиков DS18B20, 16 бит
- 4 аналоговых выхода, 12 бит
- 2 релейных выхода, 1 бит

Параметры аналоговых входов и выходов сохраняются в энергонезависимой памяти контроллера в базе данных калибровочных параметров. Каждая запись калибровочной базы данных содержит следующую информацию:

- Уникальный пользовательский идентификатор режима;
- Номер канала контроллера MA444 / MA444R2 / MA444R3/MA844;
- Аппаратные настройки канала в данном режиме;
- Калибровочные константы.

Идентификатор режима позволяет определять независимые уникальные номера режимов для различных каналов ввода вывода.

Номер канала контроллера MA444 определяет канал ввода вывода контроллера

Номер	Название
1	Аналоговый вход 1
2	Аналоговый вход 2
3	Аналоговый вход 3
4	Аналоговый вход 4
5	Аналоговый выход 1
6	Аналоговый выход 2
7	Аналоговый выход 3
8	Аналоговый выход 4

Номер канала[2] контроллера MA444R2/MA444R3/MA844R1 определяет канал ввода вывода контроллера

Номер	Номер в двоичном виде	Название
9	**001** 001	Аналоговый вход 1
10	**001** 010	Аналоговый вход 2
11	**001** 011	Аналоговый вход 3
12	**001** 100	Аналоговый вход 4
41	**101** 001	Аналоговый вход 5
42	**101** 010	Аналоговый вход 6
43	**101** 011	Аналоговый вход 7
44	**101** 100	Аналоговый вход 8
25	**011** 001	Аналоговый выход 1
26	**011** 010	Аналоговый выход 2

[2] Номер канала ввода-вывода формируется из двух трехбитовых значений: номера логического узла и номера подканала внутри узла: в старшей части кодируется номер LA, в младшей номер подканала.

33	**100** 001	Аналоговый выход 3
34	**100** 010	Аналоговый выход 4

Аппаратные настройки канала определяют режим работы АЦП и ЦАП. В настройки входит такая информация, как режим тока или напряжения, коэффициент усиления, параметры преобразования данных и т.д.
Калибровочные константы определяют поправочные коэффициенты, учитывающие разброс физических параметров разных каналов ввода/вывода.

Контроллер MA444 реализует физический узел и 4 логи ческих узла. Контроллер MA444R2/MA444R3 реализует физический узел и 5 логических узлов. Контроллер MA004/MA004R3 реализует физический узел и 1 логический узел.

6.3.1 Физический узел контроллера MA444/MA444R2/MA444R3/MA004/MA004R3

Физический узел контроллера MA444/MA444R2/MA444R3 реализует объект ошибок, объект данных и объект идентификации. Физический узел контроллера MA004/MA004R3 реализует только объекты ошибок и идентификации. Объект ошибок и объект идентификации поддерживаются в полном объеме, согласно 5.3.1 Сообщения объекту ошибок и 5.3.4 Сообщения объекту идентификации. Объект данных используется для доступа к базе данных калибровочной информации. Протокол обмена сообщениями с объектом данных физического узла контроллера MA444/MA444R2/MA444R3 описан ниже.

Формат сообщения объекту данных для доступа к базе данных калибровочной информации:

28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
signature				OBJ				D	NA							LA				Signature								T
1	1	0	0	0	0	0	1	x	x	x	x	x	x	x	x	x	x	x	x	1	1	1	1	1	1	1	1	x

Допустимы 2 вида сообщений, отличающихся размером поля данных. Возможные размеры полей данных 8 и 3 байта. В обоих случаях первые 3 байта поля данных сообщения обозначают одно и то же. В ответном сообщении ведущему узлу также допустимо DLC=1, что обозначает ошибку формата входного сообщения.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SETUP								CHANNEL								MODE_ID								SELECTOR							
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	1	1	0	0

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
DATA3								DATA2								DATA1								DATA0							
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	X	x	x	x

DLC = 8 или 3

SELECTOR — Константа равная 0x0C определяет назначение пакета для доступа к базе данных калибровочной информации

MODE_ID — Уникальный пользовательский идентификатор режима

CHANNEL	Номер канала контроллера MA444/MA444R2/MA444R3
SETUP	Аппаратные настройки канала в данном режиме
DATA[0..3]	Калибровочные константы

Во всех сообщениях конкретный элемент определяется парой {MODE_ID,CHANNEL}.

Значение битов для поля SETUP:

- Бит 7: - поменять hi и Hi местами (если при калибровке мы вышли за диапазон)
- Бит 6: - поменять lo и LO местами
- Бит 5: - не используется
- Бит 4: - тип входа/выхода (если поддерживается): 0 – напряжение, 1 – ток
- Биты 0..3: - используются в каналах вывода

Внутри MA844 все отсчеты АЦП и ЦАП приводятся к 16-ти разрядному диапазону 0..0xFFFF.

***HI** – максимальное значение для отсчетов АЦП (0xFFFF)*
***LO** - минимальное значение для отсчетов АЦП (0x0000)*
***hi** – максимальное значение АЦП, полученное при калибровке*
***lo** – минимальное значение АЦП, полученное при калибровке*

Вычисление приведенного значения АЦП или ЦАП производится по следующей формуле:

$$Nпр = ((Nизм - lo) * HI)/(hi - lo)$$

Где, Nпр – приведенное значение АЦП
Nизм – измеренное значение АЦП

Возможные входные сообщения:

REQ8	записать/переписать элемент базы данных
REQ3	стереть элемент, очистить базу данных *
ACK3	читать элемент

Возможные выходные сообщения:

ACK1	ошибка входного сообщения (неверная длина)
ACK3	элемент {MODE_ID,CHANNEL} не найден, нет свободного места или произошла ошибка работы с базой данных
ACK8	элемент {MODE_ID,CHANNEL} прочитан, данные возвращены в сообщении
REQ8	элемент {MODE_ID,CHANNEL} сохранен, данные возвращены в сообщении
REQ3	элемент {MODE_ID,CHANNEL} стерт в случае MODE_ID=CHANNEL=0xFF форматирование БД *

Возможные ответные сообщения на входные сообщения:

Входное сообщение	Выходное сообщение	Причина
REQ8	ACK1	Ошибка сообщения
	ACK3	Нет свободного места или ошибка работы с базой данных
	REQ8	Нет ошибок

REQ3	ACK1	Ошибка сообщения
	ACK3	Элемент не найден
	REQ3	Нет ошибок, команда выполнена
ACK3	ACK1	Ошибка сообщения
	ACK3	Элемент не найден
	ACK8	Нет ошибок

Команды, помеченные * реализованы только в версии контроллера MA444R2/MA444R3.

6.3.2 Логические узлы контроллера MA444/MA444R2/MA004

Логические узлы контроллера MA444/MA444R2/MA444R3/MA004/MA004R3 реализуют объект ошибок, объект данных и объект конфигурации. Все объекты поддерживаются в полном объеме, согласно 5.3.1 Сообщения объекту ошибок, 5.3.2 Сообщения объекту данных и 5.3.3 Сообщения объекту конфигурации.

[MA444] Логический узел “Аналоговые входы” (LA=1)
[MA444R2/MA444R3/MA844]Логические узлы “Аналоговые входы A” (LA=1) и “Аналоговые входы B” (LA=5)
[MA004/MA004R3] Не поддерживается
Логический узел объединяет 4 элемента ввода/вывода. Каждый элемент ввода/вывода является каналом аналогового ввода контроллера MA444/MA444R2/MA444R3. Все 4 элемента ввода/вывода логического узла имеют одинаковые параметры:
Размер данных – 16 битов, данные аналогового входа
Размер конфигурации – 4 бита, уникальный пользовательский идентификатор режима

[MA444R3/MA844]Логический узел “Температурные входы” (LA=2)
Логический узел объединяет 4 элемента ввода/вывода. Каждый элемент ввода/вывода является каналом подключения температурного датчика контроллера MA444. Все 4 элемента ввода/вывода имеют одинаковые параметры:
Размер данных – 16 битов, температурные данные
Размер конфигурации – 2 бита:

- 0 - темп. датчики выключены;
- 1 - темп. датчики включены;
- 2 - не исп.;
- 3 - входы используются как дискретные порты (см. LA=6, DSDIN).

[MA444/MA444R2/MA444R3/MA844]Логические узлы “Аналоговые выходы 1” (LA=3) и “Аналоговые выходы 2” (LA=4)
[MA004/MA004R3] Не поддерживается
Каждый логический узел объединяет 2 элемента ввода/вывода. Каждый элемент ввода/вывода является каналом аналогового вывода контроллера MA444. Оба элемента ввода/вывода имеют одинаковые параметры:
Размер данных – 16 битов, данные для выдачи в ЦАП
Размер конфигурации – 24 бита

Биты	Название	Назначение
0..15	AL_ST	данные, выдаваемые на выход в аварийном режиме
16..19	MODE_ID	уникальный пользовательский идентификатор

		режима
20	AL	1 – контроль аварийного режима включен 0 – контроль аварийного режима выключен
21-23	-	Не используются, выравнивание на границу 3 байтов

[MA444/MA844]Логические узлы "Дискретные входы (DSDIN)" (LA=6)
Логический узел объединяет 4 элемента ввода/вывода без гальванической изоляции. Каждый элемент ввода/вывода является каналом дискретного ввода контроллера MA844. Конфигурация хранится в младшей тетраде. Биту 0 соответствует DIN0, биту 3 - DIN3. Логическая «1» в бите конфигурации означает инверсию входа, а «0» отсутствие инверсии.
[MA844]Логические узлы "Дискретные выходы 1" (LA=7)

Логический узел объединяет 2 элемента ввода/вывода. Каждый элемент ввода/вывода является каналом дискретного или релейного вывода контроллера MA844. Оба элемента ввода/вывода имеют одинаковые параметры:
Размер данных – 1 битов, данные дискретного/релейного выхода
Размер конфигурации – 5 бит

Биты	Название	Назначение
0	PL	Полярность выхода
1..2	MODE	Режим 0 – статический 1 – импульсный 2 – зарезервировано 3 – выкл.
3..4	AL	Состояние в аварийном режиме 0 – выкл. 1 – не изменять 2 – установить 0 3 – установить 1

[MA844]Логические узлы "Дискретные входы 1" (LA=8)

Логический узел объединяет 2 элемента ввода/вывода. Каждый элемент ввода/вывода является каналом дискретного ввода контроллера MA844. Оба элемента ввода/вывода имеют одинаковые параметры:
Размер данных – 1 бит, данные дискретного входа
Размер конфигурации – 1 бит, полярность дискретного входа

Формальное описание контроллера MA444

```
MA444 : CONTROLLER[0,0x113]
     ADC1 : [4,0]  AT %IW1.1       (* аналоговый вход 1    *)
     ADC2 : [4,0]  AT %IW1.2       (* аналоговый вход 2    *)
     ADC3 : [4,0]  AT %IW1.3       (* аналоговый вход 3    *)
     ADC4 : [4,0]  AT %IW1.4       (* аналоговый вход 4    *)
     T1   : [2,0]  AT %IW2.4       (* температурный вход 1 *)
```

```
    T2   : [2,0]  AT %IW2.3       (* температурный вход 2 *)
    T3   : [2,0]  AT %IW2.2       (* температурный вход 3 *)
    T4   : [2,0]  AT %IW2.1       (* температурный вход 4 *)
    DAC1 : [24,0] AT %MW3.1       (* аналоговый выход 1   *)
    DAC2 : [24,0] AT %MW3.2       (* аналоговый выход 2   *)
    DAC3 : [24,0] AT %MW4.1       (* аналоговый выход 3   *)
    DAC4 : [24,0] AT %MW4.2       (* аналоговый выход 4   *)
END_CONTROLLER
```

6.3.3 Формальное описание контроллера MA444R2/MA444R3

```
MA444R2: CONTROLLER[0,0x213]
    AIN1  : [4,0]  AT %IW1.2      (* АЦП1 вход A          *)
    AIN2  : [4,0]  AT %IW1.1      (* АЦП2 вход A          *)
    AIN3  : [4,0]  AT %IW1.4      (* АЦП3 вход A          *)
    AIN4  : [4,0]  AT %IW1.3      (* АЦП4 вход A          *)
    AIN5  : [4,0]  AT %IW5.2      (* АЦП1 вход B          *)
    AIN6  : [4,0]  AT %IW5.1      (* АЦП2 вход B          *)
    AIN7  : [4,0]  AT %IW5.4      (* АЦП3 вход B          *)
    AIN8  : [4,0]  AT %IW5.3      (* АЦП4 вход B          *)
    T1    : [2,0]  AT %IW2.4      (* температурный вход 1 *)
    T2    : [2,0]  AT %IW2.3      (* температурный вход 2 *)
    T3    : [2,0]  AT %IW2.2      (* температурный вход 3 *)
    T4    : [2,0]  AT %IW2.1      (* температурный вход 4 *)
    AOUT1 : [24,0] AT %MW3.1      (* аналоговый выход 1   *)
    AOUT2 : [24,0] AT %MW3.2      (* аналоговый выход 2   *)
    AOUT3 : [24,0] AT %MW4.1      (* аналоговый выход 3   *)
    AOUT4 : [24,0] AT %MW4.2      (* аналоговый выход 4   *)
END_CONTROLLER
```

6.3.4 Формальное описание контроллера MA004/MA004R3

```
MA004R2: CONTROLLER[0,0x12E]
    T1    : [2,0]  AT %IW2.4      (* температурный вход 1 *)
    T2    : [2,0]  AT %IW2.3      (* температурный вход 2 *)
    T3    : [2,0]  AT %IW2.2      (* температурный вход 3 *)
    T4    : [2,0]  AT %IW2.1      (* температурный вход 4 *)
END_CONTROLLER
```

6.4 Контроллер MD846 / MD846R2 / MD846R3

Контроллер MD846/MD846R2/MD846R3 обладает следующими возможностями ввода/вывода:

- 8 дискретных входов, 1 бит
- 4 дискретных выхода, 1 бит
- 6 релейных выхода, 1 бит

Контроллер MD846/MD846R2/MD846R3 реализует физический узел и 2 логических узла. Параметры дискретных и релейных выходов совпадают и логически эти выходы объединены в одну группу.

6.4.1 Физический узел контроллера MD846/MD846R2/MD846R3

Физический узел контроллера MD846/MD846R2/MD846R3 реализует объект ошибок и объект идентификации. Объект ошибок и объект идентификации поддерживаются в полном объеме, согласно 5.3.1 Сообщения объекту ошибок и 5.3.4 Сообщения объекту идентификации.

6.4.2 Логические узлы контроллера MD846/MD846R2/MD846R3

Логические узлы контроллера MD846/MD846R2/MD846R3 реализуют объект ошибок, объект данных и объект конфигурации. Все объекты поддерживаются в полном объеме, согласно 5.3.1 Сообщения объекту ошибок, 5.3.2 Сообщения объекту данных и 5.3.3 Сообщения объекту конфигурации.

Логический узел “Дискретные входы” (LA=1)
Логический узел объединяет 8 элементов ввода/вывода. Каждый элемент ввода/вывода является каналом дискретного ввода контроллера MD846/MD846R2/MD846R3. Все 8 элементов ввода/вывода имеют одинаковые параметры:
Размер данных – 1 битов, данные дискретного входа
Размер конфигурации – 1 бит, полярность дискретного входа

Логический узел “Дискретные выходы” (LA=2)
Логический узел объединяет 10 элементов ввода/вывода. Каждый элемент ввода/вывода является каналом дискретного или релейного вывода контроллера MD846/MD846R2/MD846R3. Все 10 элементов ввода/вывода имеют одинаковые параметры:
Размер данных – 1 битов, данные дискретного/релейного выхода
Размер конфигурации – 5 бит

Биты	Название	Назначение
0	PL	Полярность выхода
1..2	MODE	Режим 0 – статический 1 – импульсный 2 – зарезервировано 3 – выкл.
3..4	AL	Состояние в аварийном режиме 0 – выкл. 1 – не изменять 2 – установить 0 3 – установить 1

6.4.3 Формальное описание контроллера MD846

```
MD846 : CONTROLLER[0,0x112]
     DIN1   : [1,0] AT %IX1.1        (* дискретный вход 1      *)
     DIN2   : [1,0] AT %IX1.2        (* дискретный вход 2      *)
     DIN3   : [1,0] AT %IX1.3        (* дискретный вход 3      *)
     DIN4   : [1,0] AT %IX1.4        (* дискретный вход 4      *)
     DIN5   : [1,0] AT %IX1.5        (* дискретный вход 5      *)
     DIN6   : [1,0] AT %IX1.6        (* дискретный вход 6      *)
     DIN7   : [1,0] AT %IX1.7        (* дискретный вход 7      *)
     DIN8   : [1,0] AT %IX1.8        (* дискретный вход 8      *)
     RELE1 : [5,0] AT %MX2.1         (* релейный выход 1       *)
     RELE2 : [5,0] AT %MX2.2         (* релейный выход 2       *)
     RELE3 : [5,0] AT %MX2.3         (* релейный выход 3       *)
     RELE4 : [5,0] AT %MX2.4         (* релейный выход 4       *)
     RELE5 : [5,0] AT %MX2.5         (* релейный выход 5       *)
     RELE6 : [5,0] AT %MX2.6         (* релейный выход 6       *)
     DOUT1 : [5,0] AT %MX2.10        (* дискретный выход 1     *)
     DOUT2 : [5,0] AT %MX2.9         (* дискретный выход 2     *)
     DOUT3 : [5,0] AT %MX2.8         (* дискретный выход 3     *)
     DOUT4 : [5,0] AT %MX2.7         (* дискретный выход 4     *)
END_CONTROLLER
```

6.4.4 Формальное описание контроллера MD846R2/MD846R3

```
MD846R2: CONTROLLER[0,0x212]
     DIN1   : [1,0] AT %IX1.5        (* дискретный вход 1      *)
     DIN2   : [1,0] AT %IX1.6        (* дискретный вход 2      *)
     DIN3   : [1,0] AT %IX1.7        (* дискретный вход 3      *)
     DIN4   : [1,0] AT %IX1.8        (* дискретный вход 4      *)
     DIN5   : [1,0] AT %IX1.1        (* дискретный вход 5      *)
     DIN6   : [1,0] AT %IX1.2        (* дискретный вход 6      *)
     DIN7   : [1,0] AT %IX1.3        (* дискретный вход 7      *)
     DIN8   : [1,0] AT %IX1.4        (* дискретный вход 8      *)
     RELE1 : [5,0] AT %MX2.1         (* релейный выход 1       *)
     RELE2 : [5,0] AT %MX2.2         (* релейный выход 2       *)
     RELE3 : [5,0] AT %MX2.3         (* релейный выход 3       *)
     RELE4 : [5,0] AT %MX2.4         (* релейный выход 4       *)
     RELE5 : [5,0] AT %MX2.5         (* релейный выход 5       *)
     RELE6 : [5,0] AT %MX2.6         (* релейный выход 6       *)
     DOUT1 : [5,0] AT %MX2.7         (* дискретный выход 1     *)
     DOUT2 : [5,0] AT %MX2.8         (* дискретный выход 2     *)
     DOUT3 : [5,0] AT %MX2.9         (* дискретный выход 3     *)
     DOUT4 : [5,0] AT %MX2.10        (* дискретный выход 4     *)
END_CONTROLLER
```

6.5 Контроллер CSC4

Контроллер CSC4 обладает следующими возможностями ввода/вывода:

- 4 аналоговых входа, 8/10 бит или 4 аналоговых входа 16 бит
- 4 входа температурных датчиков DS18B20, 16 бит
- 4 дискретных входа, 1 бит
- 4 дискретных выхода, 1 бит
- 2 канала подключения энкодера, 16 бит

Параметры аналоговых входов сохраняются в энергонезависимой памяти контроллера в базе данных калибровочных параметров. Каждая запись калибровочной базы данных содержит следующую информацию:

- Уникальный пользовательский идентификатор режима;
- Номер канала контроллера CSC4;
- Аппаратные настройки канала в данном режиме;
- Калибровочные константы.

Идентификатор режима позволяет определять независимые уникальные номера режимов для различных каналов аналогового ввода.

Номер канала контроллера CSC4 определяет канал ввода вывода контроллера

Номер	Название
1	Аналоговый вход 1 (8/10)
2	Аналоговый вход 2 (8/10)
3	Аналоговый вход 3 (8/10)
4	Аналоговый вход 4 (8/10)
5	Аналоговый вход 1 (16)
6	Аналоговый вход 2 (16)
7	Аналоговый вход 3 (16)
8	Аналоговый вход 4 (16)

Аппаратные настройки канала определяют режим работы АЦП. В настройки входит такая информация, как коэффициент усиления, точность и скорость преобразования, параметры преобразования данных и т.д.
Калибровочные константы определяют поправочные коэффициенты, учитывающие разброс физических параметров разных каналов ввода/вывода.

Контроллер CSC4 реализует физический узел и 6 логических узлов.

6.5.1 Физический узел контроллера CSC4

Физический узел контроллера CSC4 реализует объект ошибок, объект данных и объект идентификации. Объект ошибок и объект идентификации поддерживаются в полном объеме, согласно 5.3.1 Сообщения объекту ошибок и 5.3.4 Сообщения объекту идентификации. Объект данных используется для доступа к базе данных калибровочной информации. Протокол обмена сообщениями с объектом данных физического узла контроллера CSC4 описан в разделе 6.3.1 Физический узел контроллера

MA444/MA444R2/MA444R3/MA004/MA004R3. Дополнительные команды, помеченные * также реализованы в контроллере CSC4.

6.5.2 Логические узлы контроллера CSC4

Логические узлы контроллера CSC4 реализуют объект ошибок, объект данных и объект конфигурации. Все объекты поддерживаются в полном объеме, согласно 5.3.1 Сообщения объекту ошибок, 5.3.2 Сообщения объекту данных и 5.3.3 Сообщения объекту конфигурации.

Логический узел “Аналоговые входы” (LA=1)
Логический узел объединяет 4 элемента ввода/вывода. Каждый элемент ввода/вывода является каналом аналогового ввода контроллера MA444/MA444R2/MA444R3. Все 4 элемента ввода/вывода логического узла имеют одинаковые параметры:
Размер данных – 16 битов, данные аналогового входа
Размер конфигурации – 4 бита, уникальный пользовательский идентификатор режима

Логический узел “Температурные входы” (LA=2)
Логический узел объединяет 4 элемента ввода/вывода. Каждый элемент ввода/вывода является каналом подключения температурного датчика контроллера MA444. Все 4 элемента ввода/вывода имеют одинаковые параметры:
Размер данных – 16 битов, температурные данные
Размер конфигурации – 2 бита, (0 – выкл., 1 – вкл., 2 и 3 – зарезервированы).

[MA444/MA444R2/MA444R3]Логические узлы “Аналоговые выходы 1” (LA=3) и “Аналоговые выходы 2” (LA=4)
[MA004/MA004R3] Не поддерживается
Каждый логический узел объединяет 2 элемента ввода/вывода. Каждый элемент ввода/вывода является каналом аналогового вывода контроллера MA444. Оба элемента ввода/вывода имеют одинаковые параметры:
Размер данных – 16 битов, данные для выдачи в ЦАП
Размер конфигурации – 24 бита

Биты	Название	Назначение
0..15	AL_ST	данные, выдаваемые на выход в аварийном режиме
16..19	MODE_ID	уникальный пользовательский идентификатор режима
20	AL	1 – контроль аварийного режима включен 0 – контроль аварийного режима выключен
21-23	-	Не используются, выравнивание на границу 3 байтов

6.5.3 Формальное описание контроллера CSC4

```
CSC4: CONTROLLER[0,0x13E]
    T1    : [2,0]  AT %IW1.1 (* температурный вход 1 *)
    T2    : [2,0]  AT %IW1.2 (* температурный вход 2 *)
    T3    : [2,0]  AT %IW1.3 (* температурный вход 3 *)
    T4    : [2,0]  AT %IW1.4 (* температурный вход 4 *)
    DOUT1 : [5,0]  AT %MX3.1 (* дискретный выход 1   *)
    DOUT2 : [5,0]  AT %MX3.2 (* дискретный выход 2   *)
```

```
    DOUT3 : [5,0]  AT %MX3.3 (* дискретный выход 3   *)
    DOUT4 : [5,0]  AT %MX3.4 (* дискретный выход 4   *)
    AIN1  : [4,0]  AT %IW4.1
    AIN2  : [4,0]  AT %IW4.2
    AIN3  : [4,0]  AT %IW4.3
    AIN4  : [4,0]  AT %IW4.4
    AIN5  : [4,0]  AT %IW5.1
    AIN6  : [4,0]  AT %IW5.2
    AIN7  : [4,0]  AT %IW5.3
    AIN8  : [4,0]  AT %IW5.4
    DIN1  : [16,0] AT %IX2.1
    DIN2  : [16,0] AT %IX2.2
    DIN3  : [16,0] AT %IX2.3
    DIN4  : [16,0] AT %IX2.4
    STEP0 : [4,0]  AT %MW6.1
    TAG0  : [0,0]  AT %MX6.2
    RST0  : [0,0]  AT %MX6.3
    STEP1 : [4,0]  AT %MW6.4
    TAG1  : [0,0]  AT %MX6.5
    RST1  : [0,0]  AT %MX6.6
END_CONTROLLER
```