ФЕДЕРАЛЬНОЕ АГЕНТСТВО

ПО ТЕХНИЧЕСКОМУ РЕГУЛИРОВАНИЮ И МЕТРОЛОГИИ

НАЦИОНАЛЬНЫЙ СТАНДАРТ РОССИЙСКОЙ ФЕДЕРАЦИИ

ГОСТ Р 56947— 2016/ ISO/IEC/IEEE 21450:2010

Информационные технологии

ИНТЕРФЕЙС ИНТЕЛЛЕКТУАЛЬНОГО ПРЕОБРАЗОВАТЕЛЯ ДЛЯ ДАТЧИКОВ И ИСПОЛНИТЕЛЬНЫХ УСТРОЙСТВ

Общие функции, протоколы взаимодействия и форматы электронной таблицы данных преобразователя (ЭТДП)

(ISO/IEC/IEEE 21450:2010, IDT)

Издание официальное

Москва
Стандартинформ
2016

Предисловие

1 ПОДГОТОВЛЕН Научно-исследовательским и испытательным центром биометрической техники Московского государственного технического университета имени Н.Э. Баумана (НИИЦ БТ МГТУ им. Н.Э. Баумана) на основе собственного перевода на русский язык англоязычной версии стандарта, указанного в пункте 4, при консультативной поддержке Ассоциации автоматической идентификации «ЮНИСКАН/ГС1 РУС»

2 ВНЕСЕН Техническим комитетом по стандартизации ТК 355 «Технологии автоматической идентификации и сбора данных»

3 УТВЕРЖДЕН И ВВЕДЕН В ДЕЙСТВИЕ Приказом Федерального агентства по техническому регулированию и метрологии от 7 июня 2016 г. № 533-ст

4 Настоящий стандарт идентичен международному стандарту ISO/IEC/IEEE 21450:2010 «Информационные технологии. Интерфейс интеллектуального преобразователя для датчиков и исполнительных устройств. Общие функции, протоколы взаимодействия и форматы электронной таблицы данных преобразователя (ЭТДП)» (ISO/IEC/IEEE 21450:2010 «Information technology — Smart transducer interface for sensors and actuators — Common functions, communication protocols, and Transducer Electronic Data Sheet (TEDS) formats», IDT).

При применении настоящего стандарта рекомендуется использовать вместо ссылочных международных стандартов соответствующие им национальные стандарты, сведения о которых приведены в дополнительном приложении ДА

5 ВВЕДЕН ВПЕРВЫЕ

6 Некоторые элементы настоящего стандарта могут быть объектами патентных прав. Международная организация по стандартизации (ИСО), Международная электротехническая комиссия (МЭК) и Институт инженеров по электротехнике и радиоэлектронике (ИИЭР) не несут ответственности за установление подлинности каких-либо или всех таких патентных прав

Правила применения настоящего стандарта установлены в ГОСТ Р 1.0—2012 (раздел 8). Информация об изменениях к настоящему стандарту публикуется в годовом (по состоянию на 1 января текущего года) информационном указателе «Национальные стандарты», а официальный текст изменений и поправок — в ежемесячно издаваемом информационном указателе «Национальные стандарты». В случае пересмотра (замены) или отмены настоящего стандарта соответствующее уведомление будет опубликовано в ближайшем выпуске ежемесячного информационного указателя «Национальные стандарты». Соответствующие информация, уведомления и тексты размещаются также в информационной системе общего пользования — на официальном сайте Федерального агентства по техническому регулированию и метрологии в сети Интернет (www.gost.ru)

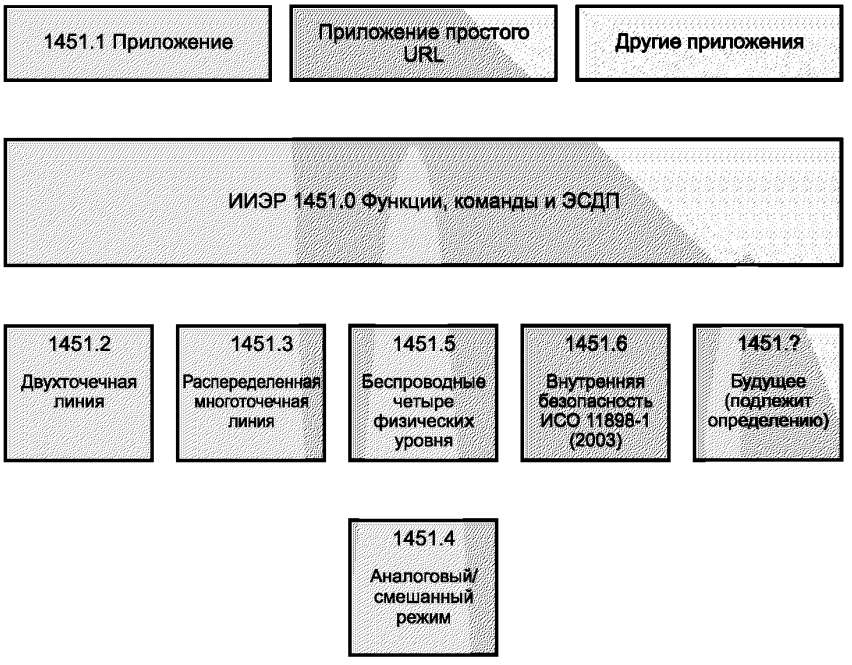
Содержание

Введение

Настоящее введение не является частью стандарта ИИЭР 1451.0—2007 «Стандарт для интерфейса интеллектуального преобразователя для датчиков и исполнительных устройств. Общие функции, протоколы взаимодействия и форматы электронной таблицы данных преобразователя (ЭТДП)».

Настоящий стандарт является основой для будущих стандартов комплекса ИИЭР 1451, использующих цифровые интерфейсы. Он также должен быть принят во внимание во время пересмотра существующих стандартов комплекса ИИЭР 1451 для обеспечения самой высокой степени совместимости между стандартами комплекса. Настоящий стандарт не распространяется на стандарт ИИЭР 1451.4—2004, в котором описаны только ограниченные по размеру ЭТДП и аналоговый интерфейс.

Взаимосвязи между настоящим стандартом и другими стандартами комплекса ИИЭР 1451 показаны на диаграмме ниже. Стандарты ИИЭР 1451.1—1999, ИИЭР 1451.2—1997 и ИИЭР 1451.3—2003 были завершены до начала разработки настоящего стандарта и не соответствуют настоящему стандарту, но будут соответствовать после пересмотра. Стандарт ИИЭР 1451.1 — приложение, которое в будущем будет согласовывать сеть пользователя с настоящим стандартом. Стандарты ИИЭР 1451.2 и ИИЭР 1451.3 также будут изменены для согласования с настоящим стандартом. Когда эти изменения будут внесены, функции преобразователя ИИЭР 1451 будут соответствовать описанным в данном стандарте, так же как и команды и ЭТДП. Стандарт ИИЭР 1451.5—2007, в котором используется любая беспроводная среда передачи данных из определенного набора, и стандарт ИИЭР Р1451.6 были написаны в рамках функций, команд и ЭТДП, описанных в настоящем стандарте. Стандарт ИИЭР 1451.4 использует интерфейс аналогового сигнала и ЭТДП, которая отличается от ЭТДП, используемых другими стандартами комплекса. Данная ЭТДП может использоваться в качестве исходных данных для любых других стандартов комплекса, но она не подчиняется функциям, командам и ЭТДП, определенным в настоящем стандарте. Элементы в таблице на сером фоне не описаны ни в одном из стандартов комплекса ИИЭР 1451, но при этом они могут быть использованы.



Основная цель данного комплекса стандартов — позволить изготовителям разработать элементы системы, имеющие возможность взаимодействовать. Для достижения данной цели в комплексе стандартов ИИЭР 1451 части системы делятся на две основные категории устройств. Одна категория — это сетевой процессор приложений (СПП) (network capable application processor, NCAP), который функционирует в качестве шлюза между сетью пользователей и интерфейсными модулями преобразователей (ИМП) (transducer interface modules, TIM). СПП является устройством на базе процессора, имеющим два интерфейса. Физический интерфейс для сети пользователей не определен ни в одном стандарте комплекса стандартов. В стандарте ИИЭР 1451.1 приведена логическая объектная модель для данного интерфейса. Другие приложения также могут быть использованы по усмотрению изготовителя. Коммуникационный интерфейс между СПП и ИМП определен в остальных стандартах комплекса стандартов. Если СПП и ИМП разработаны различными изготовителями, и при этом оба соответствуют требованиям настоящего стандарта, то они должны взаимодействовать.

В настоящем стандарте описаны функции, которые должны выполняться посредством интерфейсного модуля преобразователя или ИМП. Для обеспечения высокого уровня адресации, не зависящего от протоколов среднего и низкого физического уровня, которые используются для осуществления связи, принимаются определенные меры резервирования. В связи с этим стандарт определяет общие характеристики для всех устройств, которые приводят в действие модули преобразователей. В нем описано время получения или обработки выборок данных, определены методы группировки выходных данных от нескольких преобразователей в пределах одного ИМП. Также определены общепринятые слова состояния (статусы).

Определен стандартный набор команд для облегчения настройки и контроля модулей преобразователей, а также для считывания и записи данных, используемых системой. Также предусмотрены команды для считывания и записи ЭТДП, в которых приводятся необходимые для использования модулей преобразователей операционные характеристики системы. В стандарт также включен метод добавления уникальных команд изготовителя.

Кроме того, в настоящем стандарте представлены форматы для ЭТДП и определены некоторые ЭТДП, четыре из которых являются обязательными, остальные — дополнительными. Некоторые ЭТДП предназначены для того, чтобы позволить пользователю задавать информацию и сохранять ее в ЭТДП.

Настоящий стандарт также определяет области, которые «открыты для изготовителей». Следует отметить, что любое использование этих областей может привести к нарушениям работы функции самонастройки plug&play («включай и работай»), имеющейся у контроллеров и ИМП.

Замечания для пользователей

Опечатки

Опечатки, если таковые имеются, для настоящего и всех других стандартов доступны по следующему адресу: http://standards.ieee.org/reading/ieee/updates/errata/index.html. Пользователям рекомендуется периодически проверять данный ресурс на предмет наличия исправлений.

Пояснения

Текущие пояснения доступны по следующему адресу: http://standards.ieee.org/reading/ieee/interp/index.html.

Патенты

Следует обратить внимание на то, что некоторые элементы настоящего стандарта могут быть объектами патентных прав. При публикации настоящего стандарта не принимаются во внимание никакие положения относительно наличия или действия любых патентных прав, связанных со стандартом. ИИЭР не несет ответственности за установление подлинности каких-либо или всех таких патентных прав. Дополнительная информация может быть получена в Ассоциации стандартов ИИЭР.

НАЦИОНАЛЬНЫЙ СТАНДАРТ РОССИЙСКОЙ ФЕДЕРАЦИИ

Информационные технологии

ИНТЕРФЕЙС ИНТЕЛЛЕКТУАЛЬНОГО ПРЕОБРАЗОВАТЕЛЯ ДЛЯ ДАТЧИКОВ И ИСПОЛНИТЕЛЬНЫХ УСТРОЙСТВ

Общие функции, протоколы взаимодействия и форматы электронной таблицы данных преобразователя (ЭТДП)

Information technologies. Smart transducer interface for sensors and actuators.
Common functions, communication protocols, and transducer electronic data Sheet (TEDS) formats

Дата введения — 2017—07—01

1 Общие положения

В настоящем стандарте введены понятия интерфейсного модуля преобразователя (ИМП) (transducer interface module, TIM) и сетевого процессора приложений (СПП) (network capable application processor, NCAP), связанных с помощью передающей среды, которая определена в другом стандарте комплекса ИИЭР 1451. ИМП — это модуль, который содержит интерфейс, обработку сигнала, аналого-цифровое и цифро-аналоговое преобразования и во многих случаях преобразователь. ИМП может представлять собой как один датчик или исполнительное устройство, так и блоки, состоящие из нескольких преобразователей (датчиков и исполнительных устройств). СПП — это совокупность аппаратного и программного обеспечения, которое выполняет функцию шлюза между модулями ИМП и пользовательской сетью или главным процессором. В другом стандарте комплекса определен интерфейс связи между СПП или главным процессором и одним или несколькими ИМП. В настоящем стандарте описаны три типа преобразователей: датчики, датчики событий и исполнительные устройства.

Преобразователь называется интеллектуальным, если ему присущи следующие признаки:

- он описан в машиночитаемой электронной таблице данных преобразователя (ЭТДП) (transducer electronic data sheet, TEDS);
- управление и данные, связанные с преобразователем, являются цифровыми;
- запуск, состояние и управление предоставляются для поддержания нормального функционирования преобразователя.

СПП или главный процессор управляет ИМП с помощью цифровой интерфейсной среды. СПП является посредником между ИМП и цифровой сетью более высокого уровня и может обеспечить локальный обмен информацией.

В настоящем стандарте определен прикладной программный интерфейс (API) для приложений, которые обеспечивают связь между сетью пользователей и уровнем ИИЭР 1451.0. Также с помощью API обеспечивается связь между уровнем ИИЭР 1451.0 и нижележащими физическими уровнями связи, которые обозначены в настоящем стандарте как ИИЭР 1451.X. Такое определение API относится к системам, имеющим видимые интерфейсы. Для СПП и ИМП с единым набором аппаратного и программного обеспечения без учета отличительных особенностей отдельных интерфейсов ИИЭР 1451.0 и ИИЭР 1451.X API не является обязательным до тех пор, пока сообщения на видимых интерфейсах подчиняются правилам остальной части настоящего стандарта. Задача таких API заключается в облегчении модульного принципа проектирования до такой степени, чтобы различные поставщики могли закладывать различные выполняемые функции и при этом иметь возможность простого интегрирования различных частей по «бесшовной» технологии.

Издание официальное

В настоящем стандарте определены ИМП, которые могут быть подключены к системе и могут быть использованы без добавления специальных драйверов, профилей или других изменений в системе. Данный процесс аналогичен технологии plug&play («включай и работай»). Основными компонентами, обеспечивающими данную технологию, являются ЭТДП и набор команд. ИМП могут быть добавлены или удалены из физического уровня ИИЭР 1451 путем не более чем мгновенного воздействия на данные, передаваемые по шине. Для обозначения этой функции используется термин «горячая замена».

Настоящий стандарт построен следующим образом.

В разделе 1 «Общие положения» описана область применения настоящего стандарта.

В разделе 2 «Нормативные ссылки» перечислены стандарты и документы, на которые имеются ссылки в настоящем стандарте.

В разделе 3 «Термины и определения, обозначения и сокращения» содержатся определения, которые либо не встречаются в других стандартах, либо были изменены для использования в целях настоящего стандарта.

В разделе 4 «Типы данных» определены типы данных, используемые в настоящем стандарте.

В разделе 5 «Функциональная спецификация интеллектуальных преобразователей» определены функции ИМП и каждого канала преобразователя, который он включает.

В разделе 6 «Структура сообщений» определены структуры сообщений, которые используются для инкапсуляции данных, передаваемых между СПП и модулями ИМП.

В разделе 7 «Команды» представлены синтаксис команд и ожидаемые ответы.

В разделе 8 «Спецификация ЭТДП» определены структура и содержание электронной таблицы данных преобразователя.

В разделе 9 «Введение в прикладной программный интерфейс (API) уровня ИИЭР 1451.0» приведены общие черты двух API.

В разделе 10 «API сервисов преобразователя» представлен API, который должен быть использован приложением для работы по настоящему стандарту.

В разделе 11 «API модульных связей» описан API, который должен использоваться настоящим стандартом для связи с модулями ИМП, использующими функции связи физического интерфейса, определенные в другом стандарте комплекса ИИЭР 1451.

В разделе 12 «Протокол HTTP» описан протокол, используемый для передачи информации в Интернет. Он является более простым по сравнению с протоколом, применяемым в настоящее время согласно ИИЭР 1451.1—1999.

В приложении А «Библиография» содержатся ссылки на дополнительную информацию о темах, упомянутых в настоящем стандарте.

В приложении В «Руководящие указания для интерфейса сервисов преобразователя» приведены примеры использования данного интерфейса для измерительных и управляющих приложений, которые взаимодействуют с уровнем ИИЭР 1451.0 с помощью интерфейса сервисов преобразователя.

В приложении С «Руководящие указания для интерфейса модульных связей» приведены дополнительные указания по логической связи между СПП и модулями ИМП или между модулями ИМП, использующими API модульных связей.

В приложении D «XML-схема для текстовых ЭТДП» приведены основные схемы для текстовых ЭТДП, определенных в настоящем стандарте.

В приложении Е «Пример ЭТДП мета-идентификации» приведен пример возможной ЭТДП мета-идентификации.

В приложении F «Пример ЭТДП идентификации канала преобразователя» приведен пример возможной ЭТДП идентификации канала преобразователя.

В приложении G «Пример ЭТДП идентификации калибровки» приведен пример возможной ЭТДП идентификации калибровки.

В приложении Н «Пример командной ЭТДП» приведен пример возможной командной ЭТДП.

В приложении I «Пример ЭТДП места нахождения и заголовка» приведен пример возможной ЭТДП места нахождения и заголовка.

В приложении J «Пример ЭТДП с расширенным набором единиц измерения» приведен пример возможной ЭТДП с расширенным набором единиц измерения.

В приложении К «Примеры физических единиц» приведена серия примеров представлений физических единиц, используя представление, определенное в настоящем стандарте.

В приложении L «Протоколы считывания и записи ЭТДП» описаны процессы, которые могут быть использованы для записи или считывания ЭТДП.

В приложении М «Конфигурации логических блоков триггера» показаны некоторые возможные логические конфигурации триггера, разрешенные настоящим стандартом.

В приложении N «Система обозначений для IDL» приведены рекомендации по использованию системы обозначений IDL в настоящем стандарте.

В приложении О «Реализация ЭТДП простого датчика» приведен пример простого датчика, реализующего использование структур, определенных в настоящем стандарте.

1.1 Область применения

В настоящем стандарте разработан набор общих функций для комплекса ИИЭР 1451 стандартов интерфейса интеллектуальных преобразователей. Данный набор функций не зависит от физических сред передачи данных. Он включает в себя основные функции, необходимые для контроля и управления интеллектуальными преобразователями, общие протоколы передачи данных и форматы ЭТДП, независимые от среды связи. Это, в свою очередь, определяет набор независимых от конечной реализации API. В настоящем стандарте не рассматриваются вопросы обработки и преобразования сигнала, физическая среда и вопросы использования ЭТДП в приложениях.

1.2 Цели

В настоящее время в комплексе стандартов ИИЭР 1451 существуют три одобренных и три предлагаемых стандарта интерфейса интеллектуальных преобразователей. Все они имеют некоторые общие характеристики, однако отсутствует общий набор функций, протоколов связи и форматов ЭТДП, который обеспечивал бы взаимодействие между этими стандартами. Настоящий стандарт обеспечивает необходимую функциональную совместимость и облегчает разработку последующих стандартов для различных физических уровней, имеющих возможность взаимодействовать в рамках комплекса стандартов.

1.3 Соответствие

Основным принципом, лежащим в основе требований соответствия в настоящем подразделе, является обеспечение структуры, необходимой для повышения уровня совместимости преобразователей и систем, построенных в соответствии с настоящим стандартом, при сохранении возможности постоянного технического совершенствования и дифференциации. Реализация ИМП считается соответствующей настоящему стандарту, если она удовлетворяет следующим требованиям:

a) ИМП поддерживает необходимые технические характеристики, указанные в разделе 5;

b) ИМП поддерживает структуру сообщений, указанную в разделе 6;

c) ИМП поддерживает необходимые команды, указанные в разделе 7;

d) ИМП поддерживает необходимые ЭТДП, формат и содержание которых указаны в разделе 8;

e) ИМП поддерживает один из протоколов связи и физические носители, определенные в другом стандарте семейства комплекса ИИЭР 1451.

Примечание — Существуют некоторые рекомендации, которые желательно учитывать и которые определены в настоящем стандарте, но которые нецелесообразно переводить в разряд жестких требований. Желательно, чтобы чувствительный элемент датчика был неотъемлемой частью ИМП, но для чувствительных элементов, таких как структурные тензодатчики и термопары, это нецелесообразно, поэтому данная рекомендация не превратилась в жесткое требование. Кроме того, очень желательно, чтобы ЭТДП была расположена в пределах ИМП, но существуют системы, где окружающая среда и/или физические размеры делают это требование нецелесообразным, так что в настоящем стандарте не запрещено удаленное расположение ЭТДП от ИМП[1)].

Реализация СПП считается соответствующей настоящему стандарту, если она удовлетворяет следующим требованиям:

- СПП поддерживает необходимые технические характеристики, указанные в разделе 5;
- СПП поддерживает структуру сообщений, указанную в разделе 6;
- СПП поддерживает необходимые команды, указанные в разделе 7;
- СПП поддерживает один из протоколов связи и физические носители, определенные в другом стандарте комплекса ИИЭР 1451.

[1)] Примечания в тексте, таблицах и на рисунках приведены для справки и не содержат требований, необходимых для соответствия настоящему стандарту.

1.3.1 Соответствие ключевых слов

В настоящем стандарте для различных уровней требований и рекомендаций используются следующие ключевые слова.

1.3.1.1 Ключевое слово «должен»

Ключевое слово «должен» указывает на обязательное требование. Разработчики обязаны выполнять все такие требования для обеспечения совместимости с другими продуктами, которые соответствуют настоящему стандарту.

1.3.1.2 Ключевое слово «не должен»

Ключевое слово «не должен» указывает на обязательное недопущение. Разработчики обязаны выполнять все такие требования для обеспечения совместимости с другими продуктами, которые соответствуют настоящему стандарту.

1.3.1.3 Ключевое слово «рекомендуется»

Ключевое слово «рекомендуется» указывает на гибкость выбора с предпочтением предлагаемой альтернативы. Ключевое слово «следует» имеет тот же смысл.

1.3.1.4 Ключевое слово «следует»

Ключевое слово «следует» указывает на гибкость выбора с предпочтением предлагаемой альтернативы. Ключевое слово «рекомендуется» имеет тот же смысл.

1.3.1.5 Ключевое слово «не следует»

Ключевое слово «не следует» указывает на гибкость выбора с предпочтением того, что данная альтернатива не будет реализована.

1.3.1.6 Ключевое слово «может»

Ключевое слово «может» указывает на гибкость выбора без какого-либо предпочтения.

2 Нормативные ссылки

В настоящем стандарте использованы нормативные ссылки на следующие стандарты и другие нормативные документы, которые необходимо учитывать при использовании настоящего стандарта. В случае ссылок на документы, в которых указана дата утверждения, необходимо пользоваться только указанной редакцией. В случае, когда дата утверждения не приведена, следует пользоваться последней редакцией ссылочных документов, включая любые поправки и изменения к ним.

ANSI X3.4—1986 (Reaff 1992) Coded Character Sets — 7-bit American National Standard Code For Information Interchange[1)] (Кодированные наборы знаков. 7-битовый американский стандартный код для обмена информацией)

Extensible Markup Language (XML) 1.0 (Second Edition), W3C Recommendation, 6 October 2000[2)] [Расширяемый язык разметки (XML) 1.0 (второе издание), рекомендация W3C, 6 октября 2000 года]

HTTP URL syntax (RFC 2616) HyperText Transfer Protocol (W3C)[3)] [Синтаксис URL HTTP (RFC 2616), протокол передачи гипертекста (W3C)]

IEEE Std 754™—1985 (Reaff 1990) IEEE Standard for Binary Floating-Point Arithmetic[4), 5)] (Стандарт ИИЭР для двоичных исчислений с плавающей запятой)

IEEE Std 802.3™—2002 IEEE Standard for Information technology — Telecommunications and information exchange between systems — Local and metropolitan area networks — Specific requirements — Part 3: Carrier Sense Multiple Access with Collision Detection (CSMA/CD) access method and physical layer specifications [Стандарт ИИЭР по информационным технологиям. Телекоммуникации и обмен информацией между системами. Местные и городские сети. Особые требования. Часть 3. Метод доступа

---

1) Публикации АНСИ доступны в отделе продаж Американского национального института стандартов, 25 West 43rd Street, 4-й этаж, Нью-Йорк, 10036, США (http://www.ansi.org/).

2) Документы на языке ExtensibleMarkup можно загрузить с http://www.w3.org/TR/2000/REC-xml-20001006 или заказать в WorldWideWebConsortium, з / MIT, 32 Baccap-стрит, номер 32-G515, Кембридж, Массачусетс, 02139, США.

3) Документы, описывающие Протокол HTTP 1.1, можно загрузить с http://www.w3.org/Protocols/ или заказать в WorldWideWebConsortium, з / MIT, 32 Baccap-стрит, номер 32-G515, Кембридж, Массачусетс, 02139, США.

4) Публикации ИИЭР доступны в Институте инженеров по электротехнике и радиоэлектронике, Inc, 445 Hoes Lane, Piscataway, NJ 08854, США (http://standards.ieee.org/).

5) Стандарты ИИЭР или продукты, упомянутые в настоящем пункте, являются товарными знаками Института инженеров по электротехнике и радиоэлектронике, Inc.

CSMA/CD (коллективный (параллельный) доступ с контролем несущей частоты с обнаружением конфликтов сети) и требования к физическому уровню]

IEEE Std 1451.1™—1999 IEEE Standard for a Smart Transducer Interface for Sensors and Actuators — Network Capable Application Processor (NCAP) Information Model [Стандарт ИИЭР для интерфейса интеллектуальных датчиков и исполнительных устройств. Информационная модель сетевого процессора приложений (СПП, NCAP)]

IEEE Std 1451.2™—1997 IEEE Standard for a Smart Transducer Interface for Sensors and Actuators — Transducer to Microprocessor Communication Protocols and Transducer Electronic Data Sheet (TEDS) Formats [Стандарт ИИЭР для интерфейса интеллектуальных датчиков и исполнительных устройств. Протоколы связи между преобразователем и микропроцессором, форматы электронной таблицы данных преобразователя (ЭТДП, TEDS)]

IEEE Std 1451.3™—2003 IEEE Standard for a Smart Transducer Interface for Sensors and Actuators — Digital Communication and Transducer Electronic Data Sheet (TEDS) Formats for Distributed Multidrop Systems (Стандарт ИИЭР для интерфейса интеллектуальных датчиков и исполнительных устройств. Цифровая связь и форматы ЭТДП для распределенных моноканальных систем)

IEEE Std 1451.4™—2004 IEEE Standard for a Smart Transducer Interface for Sensors and Actuators — Mixed-Mode Communication Protocols and Transducer Electronic Data Sheet (TEDS) Formats [Стандарт ИИЭР для интерфейса интеллектуальных датчиков и исполнительных устройств. Протоколы связи в смешанном режиме и форматы электронной таблицы данных преобразователя (ЭТДП, TEDS)]

IEEE Std 1588™—2002 IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems (Стандарт ИИЭР для протокола точной синхронизации времени для сетевых измерений и систем управления)

ISO 639:1988[1)] Codes for the Representation of Names of Languages[2)] (Коды для представления названий языков)

ISO 19136 Geographic information — Geography Markup Language (GML) [Географическая информация. Язык географической разметки (GML)]

ISO/IEC 14750:1999 Information technology — Open Distributed Processing — Interface Definition Language[3)] (Информационные технологии. Открытые процессы. Язык интерфейса)

3 Термины и определения, обозначения и сокращения

3.1 Термины и определения

В настоящем стандарте применены термины и определения в соответствии со словарем терминов стандартов Института инженеров по электротехнике и радиоэлектронике (ИИЭР, IEEE) [B2][4)], а также следующие термины с соответствующими определениями:

3.1.1 **исполнительное устройство** (actuator): Преобразователь, на вход которого поступает выборка (выборки) данных, которые затем преобразуются в действие. Действие может быть осуществлено в пределах или за пределами ИМП.

3.1.2 **адрес** (address): Символ или группа символов, которая определяет регистр, определенную часть запоминающего устройства или какой-либо другой источник данных или место назначения.

3.1.3 **адресная группа** (Address Group): Совокупность каналов преобразователя, которые соответствуют одному адресу.

---

1) Заменен на серию стандартов ИСО 639, состоящую из пяти частей. Однако для однозначного соблюдения требования настоящего стандарта, выраженного в датированной ссылке, рекомендуется использовать только указанное в этой ссылке издание.

2) Публикации ИСО доступны в центральном секретариате ИСО по адресу: Case Postale 56, 1 rue de Varembé, CH-1211, Genève 20, Switzerland/ Suisse (http://www.iso.ch/). Публикации ИСО также доступны в США в отделе продаж Американского института национальных стандартов: American National Standards Institute, 25 West 43rd Street, 4th Floor, New York, NY 10036, USA (http://www.ansi.org/).

3) Публикации ИСО доступны в центральном секретариате ИСО по адресу: Case Postale 56, 1 rue de Varembé, CH-1211, Genève 20, Switzerland/ Suisse (http://www.iso.ch/). Публикации ИСО также доступны в США в Global Engineering Documents, 15 Inverness Way East, Englewood, CO 80112, USA (http://global.ihs.com/). Электронные копии доступны в США в Американском институте национальных стандартов: American National Standards Institute, 25 West 43rd Street, 4th Floor, New York, NY 10036, USA (http:// www.ansi.org/).

4) Ссылка на литературу, указанную в приложении А.

3.1.4 **буфер** (buffer): Промежуточное место хранения данных, которое используется для компенсации разницы в скорости потока данных или во времени возникновения событий при передаче информации от одного устройства к другому.

3.1.5 **калибровка** (calibration): Процесс определения информации, которая постоянно хранится в ЭТДП калибровки, для обеспечения коррекции.

3.1.6 **контрольная группа** (Control Group): Технические характеристики изготовителя, которые определяют внутренние связи между каналами многоканального ИМП. Информация контрольной группы, как правило, не используется в самом ИМП. Эта информация применяется для идентификации каналов преобразователя, которые используются для управления какой-либо характеристикой другого канала преобразователя. Например, контрольная группа может быть использована для идентификации исполнительного устройства, которое применяется для установления порога аналогового датчика событий.

3.1.7 **коррекция** (correction): Вычисление полиномиальной функции с использованием информации от ЭТДП калибровки совместно с данными от одного или нескольких преобразователей.

3.1.8 **модель данных** (data model): Числовой формат, в котором ИМП должен получать или передавать данные.

3.1.9 **набор данных** (data set): Совокупность выборок, полученных от датчика (или применяемых исполнительным устройством) в ответ на команду запуска.

3.1.10 **таблица данных** (data sheet): Перечень информации об устройстве, которая определяет параметры работы и условия применения (как правило, разрабатывается изготовителем).

3.1.11 **цифровой интерфейс** (digital interface): Средства коммуникации и протокол для передачи информации только в виде двоичного кода.

3.1.12 **электронная таблица данных** (electronic data sheet): Таблица данных, которая хранится в какой-либо форме в электронно-считываемом запоминающем устройстве (в отличие от бумаги).

3.1.13 **встроенный преобразователь** (embedded transducer): Устройство, действующее как преобразователь с точки зрения СПП, даже если никакие параметры за пределами ИМП не считываются и не изменяются. Встроенные преобразователи могут быть использованы для установки или считывания рабочих параметров других преобразователей.

3.1.14 **нумерация** (enumeration): Присвоение числового значения определенным значениям в контексте определенного поля данных. Для удобства пользователя двоичные числа, как правило, выражаются в десятичном виде. Не все возможные числовые значения обязательно должны иметь значения из поля данных. Числовые значения, которым соответствуют пустые значения определенного поля данных, не используются или резервируются для использования в будущем. Нумерация является процессом кодировки интерпретируемой человеком информации в вид, удобный для хранения и обмена в цифровой электронной машине. Любой подраздел, который определяет поле данных ЭТДП для нумерации, должен содержать таблицу, определяющую значение определенного поля данных для каждого возможного числового значения. Значения, закодированные в каждом поле данных, должны быть конкретными и уникальными для этой области данных и только в этой области данных. Числовое значение становится бессмысленным, если оно не связано с полем данных и не определено в таблице.

3.1.15 **датчик событий** (event sensor): Датчик, который обнаруживает изменение состояния в материальном мире. При этом для такого датчика «измерением» считается факт изменения состояния в материальном мире и/или момент времени изменения состояния, а не величина данного изменения или значение состояния.

3.1.16 **горячая замена** (hot swap): Акт подключения или отключения ИМП от интерфейсной среды преобразователя без предварительного отключения питания, которое подводится к ИМП через среду.

3.1.17 **младший значащий бит**; МЗБ (least significant bit; LSB): Бит в двоичной записи числа, который является показателем наименьшей возможной степени.

3.1.18 **сообщение** (message): Информация, которая должна быть передана между устройствами как единый логический объект. Сообщение может занимать один или несколько пакетов.

3.1.19 **мета-** (meta-): Приставка, заимствованная из греческого языка, которая означает что-то, относящееся к целому или общему логическому объекту или являющееся общим или совместным со всеми логическими элементами, составляющими целое.

3.1.20 **мета-ЭТДП** (meta-TEDS): Совокупность тех полей данных ЭТДП, которые относятся к целому или общему объекту, или тех, которые являются общими или совместными со всеми логическими элементами каналов преобразователя, составляющими весь продукт.

3.1.21 **полином** (multinomial): Линейная сумма слагаемых, возведенных в степень для более чем одной переменной (степенной ряд).

$$\sum_{i_1=0}^{N_1}\sum_{i_2=0}^{N_2}\ldots\sum_{i_m=0}^{N_m} A(i_1,i_2,\ldots i_m)x_1^{i_1}x_2^{i_2}\ldots x_m^{i_m}$$

3.1.22 **сетевой процессор приложений**; СПП (network-capable application processor; NCAP): Устройство между модулями преобразователей и сетью. СПП осуществляет сетевую связь, связь модулей ИМП и преобразование данных или другие функции обработки.

3.1.23 **нечисло**; НЧ (not-a-number; NaN): Как определено в ИИЭР 754—1985[1)] — битовая комбинация действительного числа одинарной точности или двойной точности, которая является результатом неправильной операции с плавающей точкой. Для действительных чисел одинарной точности битовая комбинация должна иметь степень 255 (десятичное число) и ненулевую мантиссу. Знак не учитывается при определении того, является ли значение нечислом.

3.1.24 **байт**[2)] (octet): Группа из 8 битов (в Соединенных Штатах Америки октет обычно называют байтом).

3.1.25 **пакет** (packet): Блок информации, который должен быть передан физическим уровнем между устройствами за одну передачу.

3.1.26 **отсчет зафиксирован** (sample latched): Данный термин используется в датчике, чтобы сигнализировать, что отсчет был получен. Например, когда выборка и блокировочная схема переключаются в режим «hold» (режим «удержать») или при другой подобной операции. Для исполнительного устройства данный термин сигнализирует, что отдельный входной сигнал исполнительного устройства был обработан согласно логике устройства.

3.1.27 **датчик** (sensor): Преобразователь физического, биологического или химического параметра в электрический сигнал.

3.12.28 **время установки** (setup time): Время между первоначальным запросом выполнения функции и непосредственным запуском задачи.

3.1.29 **формирование сигнала** (signal conditioning): Обработка сигнала преобразователя, которая включает такие операции, как усиление, компенсация, фильтрация и нормализация.

3.1.30 **интеллектуальный преобразователь** (smart transducer): Преобразователь, который выполняет функции помимо тех, которые необходимы для формирования правильного представления получаемой или регулируемой величины. Эти дополнительные функции обычно упрощают интеграцию преобразователя в приложения в сетевой среде.

3.1.31 **сигнал синхронизации** (synchronization signal): В настоящем стандарте сигнал, передаваемый через СПП, чтобы обеспечить сигналы тактовой или временной синхронизации для ИМП или группы ИМП. Сигналы синхронизации имеются в распоряжении не для всех стандартов физического уровня. ИМП с низкой стоимостью и производительностью могут не содержать ресивер для синхронизации сигнала.

3.1.32 **преобразователь** (transducer): Устройство, которое преобразует энергию из одного вида в другой. Устройство может быть либо датчиком, либо исполнительным устройством.

3.1.33 **интерфейсный модуль преобразователя**; ИМП (transducer interface module; TIM): Модуль, содержащий ЭТДП, логику реализации интерфейса преобразователя, преобразователь (преобразователи) или подключение к преобразователю (преобразователям), а также любое преобразование или формирование сигнала.

3.1.34 **канал преобразователя** (Transducer Channel): Преобразователь и все связанные с ним компоненты формирования и преобразования сигнала.

3.1.35 **номер канала преобразователя** (Transducer Channel number): 16-битное число, присвоенное изготовителем индивидуальному каналу преобразователя в ИМП.

3.1.36 **прокси-сервер канала преобразователя** (Transducer Channel proxy): Устройство, которое обеспечивает обработку совокупности каналов преобразователя как единого целого. Прокси-сервер канала преобразователя аналогичен каналу преобразователя за исключением того, что он не требует наличия ЭТДП канала преобразователя, не может иметь ЭТДП калибровки, ЭТДП передаточной функ-

---

[1)] Информация о ссылках приведена в разделе 2.

[2)] В оригинале ISO/IEC/IEEE 21450:2010 используется термин «октет».

ции или ЭТДП частотной характеристики. Он может поддерживать другие ЭТДП. Прокси-сервер канала преобразователя может реагировать на команды.

3.1.37 **электронная таблица данных преобразователя**; ЭТДП (Transducer Electronic Data Sheet; TEDS): Электронная таблица данных, описывающая ИМП или канал преобразователя. Структуры сложных ЭТДП описаны в настоящем стандарте.

3.1.38 **передача** (transfer): Акт или процесс перемещения информации из одного цифрового устройства в другое.

3.1.39 **передаточная функция** (transfer function): Функция, которая определяет отклик канала преобразователя на сигналы различной амплитуды и частоты.

3.1.40 **триггер** (trigger): Сигнал или сообщение, которое используется для запуска действия.

3.1.41 **векторная группа** (Vector Group): Технические характеристики изготовителя, которые определяют внутренние связи между каналами многоканального ИМП. Информация векторной группы, как правило, не используется в самом ИМП. Данная информация обычно применяется в приложениях СПП, чтобы правильно составить читаемые человеком дисплеи или при формулировке других вычислений. Например, векторные группы могут быть использованы для указания, какие каналы преобразователя представляют каждую из трех векторных осей в трехмерном пространстве измерений.

3.1.42 **виртуальная ЭТДП** (virtual TEDS): ЭТДП, которая постоянно хранится в отличном от ИМП месте.

3.2 Сокращения

В настоящем стандарте применены следующие сокращения:

н. э. — нашей эры (Anno Domini, AD);

ОСК — опорная система координат (Coordinate Reference System, CRS);

DTD — объявление типа документа со ссылкой на язык разметки XML (Extensible Markup Language, XML);

HTTP — протокол передачи гипертекста (Hypertext Transfer Protocol);

MJD — измененный юлианский календарь (modified Julian date);

НЧ — нечисло (not a number, NaN);

СПП — сетевой процессор приложений (network-capable application processor, NCAP);

OMG — группа управления объектами (Object Management Group, OMG) — консорциум, который стремится к разработке технически совершенных, коммерчески жизнеспособных и независимых от поставщиков спецификаций для индустрии программного обеспечения;

ФУ — физический уровень (physical layer, PHY);

СИ — Международная система единиц (International System of Units, SI) [B9];

МАВ — Международное атомное время (International Atomic Time, TAI);

ИМП — интерфейсный модуль преобразователя (transducer interface module, TIM);

ЭТДП — электронная таблица данных преобразователя (transducer electronic data sheet, TEDS);

СИП — сервисный интерфейс преобразователя (Transducer Service Interface, TSI);

АСКОИ — американский стандартный код для обмена информацией (АНСИ X3.4-1986) (U.S. American standard code for information interchange, USASCII);

ВСВ — Всемирное скоординированное время (universal coordinated time, UTC);

УУИД — универсальный уникальный идентификатор (universal unique identifier, UUID);

Xdcr — преобразователь (transducer);

XML — расширяемый язык разметки (eXtensible Markup Language).

4 Типы данных

Все типы данных, используемые в настоящем стандарте, определены в соответствующих подразделах.

4.1 8-разрядное целое число без знака

Символ: Uint8.

Размер: 1 байт.

IDL: `typedefoctet UInt8.`

Данный тип данных используется для представления положительных целых чисел от 0 до 255.

4.2 16-разрядное целое число без знака

Символ: UInt16.
Размер: 2 байта.
IDL: `typedef unsigned short UInt16.`
Данный тип данных может принимать любое значение от 0 до 65 535.

4.3 32-разрядное целое число со знаком

Символ: Int32.
Размер: 4 байта.
IDL: `typedef long Int32.`
Данный тип данных используется для представления целого числа от –2 147 483 648 до 2 147 483 647.

4.4 32-разрядное целое число без знака

Символ: UInt32.
Размер: 4 байта.
IDL: `typedef unsigned long UInt32.`
Данный тип данных используется для представления положительных целых чисел от 0 до 4 294 967 295.

4.5 Действительное число одинарной точности

Символ: Float32.
Размер: 4 байта.
IDL: `typedef float Float32.`
Действительное число одинарной точности является 32-разрядной двоичной последовательностью, которая кодирует действительные числа, как определено в стандарте ИИЭР 754—1985.

4.5.1 НЧ с плавающей точкой в ЭТДП

В соответствии с ИИЭР 754—1985 число одинарной точности с показателем (порядком, степенью) 255 и дробной частью (мантиссой), отличной от нуля, является НЧ независимо от знакового бита (знакового разряда). Рекомендуемое значение для НЧ для использования в поле ЭТДП — 0x7FFFFFFF (шестнадцатеричный).

4.6 Действительное число двойной точности

Символ: Float64.
Размер: 8 байтов.
IDL: `typedef double Float64.`
Действительное число двойной точности является 64-разрядной двоичной последовательностью, которая кодирует действительные числа, как определено в стандарте ИИЭР 754—1985.

4.7 Тип данных String (тип данных «строка»)

Символ: _String.
Размер: переменный.
IDL: `typedef string _String; // Leading '_' to escape reserved IDL keyword.`
Использование текстовых строк как типовой строки в качестве основного типа, который определен в ИИЭР 1451.2—1997, было заменено с использованием стандартов консорциума Object Management Group (OMG) для языка описания интерфейса (IDL).

XML, используемый в 8.9, — это текстовые ЭТДП. Регулирующим документом для XML являются Рекомендации W3C по расширяемому языку разметки (XML) 1.0 (второе издание) [W3C Recommendation *Extensible Markup Language (XML) 1.0 (Second Edition)*]. Все текстовые строки в ЭТДП должны соответствовать этим рекомендациям или будущим обновлениям к ним.

4.8 Тип Boolean (логический тип данных)

Символ : _Boolean.
Размер: 1 байт.

IDL: `typedef boolean _Boolean; // Leading '_' to escape reserved IDL keyword.`

Логический тип является основным типом, который определен в стандартах консорциума Object Management Group (OMG) для языка описания интерфейса (IDL).

В настоящем стандарте бит или байт с ненулевым значением имеет статус «True» («истина»). Переменная с нулевым значением имеет статус «False» («ложь»).

4.9 Класс «IEEE1451Dot0::Args::TimeRepresentation» (класс «ИИЭР1451.0::Аргументы::Представление времени»)

Данный абстрактный класс определяет представление времени. Он состоит из двух подклассов: TimeInstance и TimeDuration. Определение данных двух аргументов приведено в таблице 1.

IDL:
```
struct TimeRepresentation {
    UInt32 secs;
    UInt32 nsecs;
    }.
```

Примечание — Для данного формата исправлены недостатки неправильного временного представления, обнаруженные в ИИЭР 1588—2002.

Таблица 1 — Структура представления временных данных

Параметр	Тип	Описание
secs (с)	UInt32	Секунды — беззнаковое 32-разрядное число, представляющее собой число секунд с начала периода отсчета времени (обычно 00 часов 1 января 1970 г.)
nsecs (нс)	UInt32	Знак, наносекунды — беззнаковое 32-разрядное целое число, состоящее из двух меньших по размеру полей. Старший бит будет интерпретироваться как знаковый бит значения времени. Младшие 31 бит представляют число наносекунд, которые будут добавлены к значению, определенному в поле «секунды» до применения знака. Значение, указанное в поле «наносекунды», ограничивается областью от 0 до 999 999 999 включительно

В настоящем стандарте используется временное представление, которое определено в стандарте ИИЭР 1588—2002. Одним из последствий этого выбора является то, что время измеряется в секундах в соответствии с Международным атомным временем (International Atomic Time, TAI), а не Всемирным координированным временем (Universal Time Coordinated, UTC). TAI является международным стандартом для времени на основе секунды международной системы СИ, основанной на вращающемся геоиде. TAI обеспечивается набором атомных часов и формирует основу отсчета времени для других общепринятых временных шкал. Из этих шкал UTC является временной шкалой, представляющей наибольший инженерный и коммерческий интерес. Представление UTC определено в ИСО 8601 [B5] в формате **ГГГГ-ММ-ДД** для даты и **чч:мм:сс** для времени суток.

Длительность секунды во времени по UTC идентична длительности секунды по TAI. Время UTC отличается от времени TAI на постоянное смещение. Данное смещение преобразуется время от времени путем прибавления или вычитания секунд координации (високосных секунд).

Начиная с 00 часов 1 января 1972 г. [согласно измененному юлианскому календарю (MJD) 41 317.0] стандартные системы времени в мире начали вводить секунды координации для обеспечения только интегральной секундной коррекции между секундами по UTC и по TAI. И в UTC, и в TAI время выражается в днях, часах, минутах и секундах. Внесение секунд координации, которое применяется в системе UTC, но не применяется в TAI, предпочтительно осуществляется после 23:59:59 в последний день июня или декабря. Первая такая коррекция, одиночная положительная коррекция за счет секунд координации, была сделана после 23:59:59 30 июня 1972 г., и на тот момент время по UTC отставало на 11 секунд от времени по TAI.

4.9.1 Подкласс «IEEE1451Dot0::Args::TimeDuration» (подкласс «ИИЭР1451.0::Аргументы::Продолжительность по времени»)

Данный подкласс представления времени используется для указания временного интервала, а не временного значения. Определение данных двух аргументов приведено в 4.9.

IDL:
```
struct TimeDuration {
    UInt32 secs;
    UInt32 nsecs;
    }.
```

4.9.2 Подкласс «EEE1451Dot0::Args::TimeInstance»
(подкласс ИИЭР1451.0::Аргументы::Момент времени»)

Значения времени задаются данной структурой на уровне ИИЭР 1451.0. Данная структура используется, когда необходимо представить значение времени, а не продолжительность времени. Значение времени до начала отсчетного периода времени задается отрицательным полем «нс» (nsecs).

IDL:
```
struct TimeInstance {
    UInt32 secs;
    UInt32 nsecs;
    }.
```

Подкласс **TimeInstance** основан на периоде, который начался в 00 часов 1 января 1970 г. Определение данных двух аргументов приведено в 4.9.

4.10 Типы данных для соответствующих приложений

Типы данных, описанные в подразделе, не используются в настоящем стандарте, но приводятся для обеспечения передачи данных в этих форматах приложениям, использующим настоящий стандарт.

4.10.1 Восьмиразрядное целое число со знаком

Символ: Int8.

Размер: 1 байт.

IDL: `typedef char Int8.`

Данный тип данных используется для представления целых чисел от минус 128 до 127.

4.10.2 Шестнадцатиразрядное целое число со знаком

Символ: Int16.

Размер: 2 байта.

IDL: `typedef short Int16.`

Данный тип данных используется для представления целых чисел от минус 32 768 до 32 767.

4.11 Физические единицы измерения

Символ: UNITS.

Размер: 11 байтов.

IDL:
```
struct Units {
    UInt8 interpretation;
    UInt8 radians;
    UInt8 steradians;
    UInt8 meters;
    UInt8 kilograms;
    UInt8 seconds;
    UInt8 amperes;
    UInt8 kelvins;
    UInt8 moles;
    UInt8 candelas
    UInt8 Units Extension TEDS Access Code
    }.
```

Физические единицы измерения — это двоичная последовательность из 10 байтов, которая кодирует физические единицы в соответствии с таблицами 2 и 3. Каждое поле должно толковаться как целое число без знака. Каждая единица должна быть представлена в виде произведения основных единиц СИ, а также радиан и стерадиан, каждая из которых возведена в рациональную степень. Структура, представленная в таблице 2, кодирует только показатели степени; само произведение является неявным. Примеры физических единиц приведены в приложении J. Для получения дополнительной информации см. Hamilton [B1].

Таблица 2 — Структура типов данных физических единиц

Поле	Описание	Тип данных	Число байтов
1	Интерпретация физических единиц (см. таблицу 3)	UInt8	1

Окончание таблицы 2

Поле	Описание	Тип данных	Число байтов
2	(2 × <показатель степени для единицы «радиан»>) + 128	UInt8	1
3	(2 × <показатель степени для единицы «стерадиан»>) + 128	UInt8	1
4	(2 × <показатель степени для единицы «метр»>) + 128	UInt8	1
5	(2 × <показатель степени для единицы «килограмм»>) + 128	UInt8	1
6	(2 × <показатель степени для единицы «секунда»>) + 128	UInt8	1
7	(2 × <показатель степени для единицы «ампер»>) + 128	UInt8	1
8	(2 × <показатель степени для единицы «кельвин»>) + 128	UInt8	1
9	(2 × <показатель степени для единицы «моль»>) + 128	UInt8	1
10	(2 × <показатель степени для единицы «кандела»>) + 128	UInt8	1

Формы U/U (значения 1 и 3 графы «Нумерация» в таблице 3) используются для выражения «безразмерных» единиц, таких как деформация (в метрах на метр) и концентрация (в молях на моль). Числитель и знаменатель единиц идентичны, каждый из них определяется подполем от 2 до 10.

Логические данные (значения {0, 1} или {False, True}) должны быть представлены в виде цифровых данных (значение 4 графы «Нумерация» в таблице 3).

Таблица 3 — Интерпретация физических единиц

Нумерация	Наименование постоянной	Определение
0	PUI_SI_UNITS	Единица описывается произведением основных единиц СИ, а также радиан и стерадиан, возведенных в степени, записанные в полях со 2 по 10 в таблице 2. Единицы измерения для некоторых величин, таких как число людей через турникет, не могут быть представлены с использованием этих единиц. Для таких случаев, когда величина находится в процессе определения, следует пользоваться нумерацией 0 с установленными значениями 128 для полей 2—10
1	PUI_RATIO_SI_UNITS	Единицей является U/U, где U описывается произведением основных единиц СИ, а также радиан и стерадиан, возведенных в степени, записанные в поля со 2 по 10
2	PUI_LOG10_SI_UNITS	Единицей является log10(U), где U описывается произведением основных единиц СИ, а также радиан и стерадиан, возведенных в степени, записанные в поля со 2 по 10
3	PUI_LOG10_RATIO_SI_UNITS	Единицей является log10(U/U), где U описывается произведением основных единиц СИ, а также радиан и стерадиан, возведенных в степени, записанные в поля со 2 по 10
4	PUI_DIGITAL_DATA	Соответствующая величина — это цифровые данные (например, бит-вектор), и эта величина не имеет размерности. В полях 2—10 должно быть установлено значение 128. Тип «цифровые данные» относится к данным, которые не представляют количество, например, текущие положения переключателей коммутационного блока
5	PUI_ARBITRARY	Соответствующая физическая величина представляется значениями на произвольной шкале (например, прочность). Поля 2—10 резервируются, в них должны быть установлены значения 128
6—255	—	Зарезервировано

4.12 Универсальная уникальная идентификация (УУИД)

Символ: UUID.
Размер: 10 байтов.
IDL: `typedef UUID Short [5].`
Поле «УУИД» является полем идентификации, связанным с ИМП, значение которого является уникальным на планете. Поле «УУИД» должно иметь размер 10 байтов и состоять из четырех расположенных по порядку от старшего (наиболее значимого) к младшему подполей: подполя «Место нахождения», подполя «Изготовитель», подполя «Год» и подполя «Время», определенных в таблице 4.
При этом не требуется, чтобы интерпретация УУИД отражала фактическое место или время изготовления ИМП. Время и место должны использоваться в алгоритме только для обеспечения уникальности. Таким образом, изготовитель должен иметь возможность использовать поля, определенные с помощью алгоритма, по своему усмотрению при условии, что при интерпретации согласно таблице 4 не должно быть никакого другого изготовителя, претендующего на использование той же идентификации УУИД для ИМП.

4.13 Произвольный байтовый массив

Символ: OctetArray.
Размер: меняется.
Данный тип данных включает в себя произвольное число байтов, обработанных как совокупный логический объект, которые могут быть или не могут быть истолкованы как число. Массив OctetArray может являться структурой, содержащей один или более простейших типов данных, массивы простейших типов данных или меньшие массивы OctetArrays.

4.14 Массив строк

Символ: StringArray.
Размер: меняется.
Данный тип данных включает в себя произвольное число данных типа **String** (см. 4.7), обработанных как совокупный логический объект.

4.15 Массив логических данных

Символ: BooleanArray.
Размер: меняется.
Данный тип данных включает в себя произвольное число данных типа **Boolean** (см. 4.8), обработанных как совокупный логический объект.

4.16 Массив 8-разрядных целых чисел со знаком

Символ: Int8Array.
Размер: меняется.
Данный тип данных включает в себя произвольное число байтов, обработанных как совокупный логический объект, состоящий из 8-разрядных целых чисел со знаком (Int8).

4.17 Массив 16-разрядных целых чисел со знаком

Символ: Int16Array.
Размер: меняется.
Данный тип данных включает в себя произвольное число 16-разрядных целых чисел со знаком (Int16), обработанных как совокупный логический объект.

4.18 Массив 32-разрядных целых чисел со знаком

Символ: Int32Array.
Размер: меняется.
Данный тип данных включает в себя произвольное число 32-разрядных целых чисел со знаком (Int32)[1)], обработанных как совокупный логический объект.

[1)] В оригинале ISO/IEC/IEEE 21450:2010 допущена ошибка. Ошибочно приведено значение Int16.

Таблица 4 — Структура типов данных универсальной уникальной идентификации УУИД

Поле	Описание	Число битов
1	Поле «Место нахождения» (Location): значение данного поля должно быть выбрано изготовителем ИМП для определения конкретного места на Земле, места нахождения, над которым изготовитель имеет физический контроль. Данное значение может представлять фактическое место нахождения изготовителя ИМП. Изготовитель может использовать в своей работе различные значения данного поля, но только если они удовлетворяют требованиям настоящего подраздела. Поле «Место нахождения» должно быть представлено 42 битами. Старший значащий бит указывает на северную (бит установлен) или южную (бит не установлен) широту. Следующие 20 старших значащих битов данного поля представляют собой значение широты места нахождения как целое число угловых секунд. Следующий старший значащий бит должен указывать на восточную (бит установлен) или западную (бит не установлен) долготу. Остальные 20 битов представляют значение долготы места нахождения как целое число угловых секунд. Значения широты более 90° зарезервированы. Значения долготы более 180° зарезервированы. Примечание — Одна угловая секунда на экваторе составляет около 30 м. Таким образом, диапазон, представляемый каждым 20-битовым полем, составляет от 0 до 1 048 575 угловых секунд или от 0° до 291°, что является достаточным для представления широты и долготы на поверхности Земли	42
2	Поле «Изготовитель» (Manufacturer): значение данного поля может быть выбрано изготовителем ИМП для любых целей при условии, что не возникает конфликтных ситуаций, связанных с совпадениями при использовании поля «Место нахождения». Такая конфликтная ситуация в поле «Место нахождения» происходит в том случае, если на физический контроль над местом нахождения, заданным в поле «Место нахождения», могут претендовать более одного изготовителя. Если такой конфликт существует, то все пострадавшие изготовители должны согласовать использование значений поля «Изготовитель» для исключения каких-либо совпадений. Таким образом, сочетание поля «Место нахождения» и поля «Изготовитель» должно однозначно определить конкретного изготовителя ИМП. Такое согласование должно возобновляться каждый раз, когда происходит совпадение, вызывающее конфликтную ситуацию	4
3	Поле «Год» (Year): значение данного поля должно отображать текущий год. Поле «Год» должно быть представлено 12-разрядным целым значением. Диапазон данного поля должен составлять от 0 г. до 4095 г. н. э. Началом года принято считать 1 января, 00:00:00 по TAI	12
4	Поле «Время» (Time): данное значение должно быть выбрано изготовителем ИМП таким образом, чтобы в сочетании с полями «Место нахождения», «Изготовитель» и «Год» результирующий УУИД являлся уникальным для всех ИМП, сделанных под контролем данного изготовителя. Выбор значений для поля «Время» должен быть, кроме того, ограничен таким образом, чтобы значения, которые интерпретируются как время с начала года, не представляли ни время, предшествующее получению изготовителем физического контроля над местом нахождения, ни значений времени в будущем. Поле «Время» должно быть представлено 22-разрядным целым числом. Диапазон должен составлять от 0 до 4 194 303. Если необходимо интерпретировать данное поле как время с начала года, то оно должно быть представлено целым числом 10-секундных интервалов. В этом случае значения времени более одного года зарезервированы. Началом года принято считать 1 января, 00:00:00 по TAI. Примечание — В году примерно 31 536 000 с	22

4.19 Массив 8-битовых целых чисел без знака

Символ: UInt8Array.

Размер: меняется.

Данный тип данных включает в себя произвольное число байтов, обработанных как совокупный логический объект, состоящий из 8-разрядных целых чисел без знака (Int8).

4.20 Массив 16-битовых целых чисел без знака

Символ: UInt16Array.
Размер: меняется.
Данный тип данных включает в себя произвольное число 16-разрядных целых чисел без знака (Int16), обработанных как совокупный логический объект.

4.21 Массив 32-битовых целых чисел без знака

Символ: UInt32Array.
Размер: меняется.
Данный тип данных включает в себя произвольное число 32-разрядных целых чисел без знака (Int32)[1)], обработанных как совокупный логический объект.

4.22 Массив вещественных чисел одинарной точности

Символ: Float32Array.
Размер: меняется.
Данный тип данных включает в себя произвольное число вещественных чисел одинарной точности, как определено в ИИЭР 754—1985, обработанных как совокупный логический объект.

4.23 Массив вещественных чисел двойной точности

Символ: Float64Array.
Размер: меняется.
Данный тип данных включает в себя произвольное число вещественных чисел двойной точности, как определено в ИИЭР 754—1985, обработанных как совокупный логический объект.

4.24 Массив типов данных TimeDuration

Символ: TimeDurationArray.
Размер: меняется.
Данный тип данных включает в себя произвольное число типов данных TimeDuration, как определено в 4.9.1, обработанных как совокупный логический объект.

4.25 Массив типов данных TimeInstance

Символ: TimeInstanceArray.
Размер: меняется.
Данный тип данных включает в себя произвольное число типов данных TimeInstance, как определено в 4.9.2, обработанных как совокупный логический объект.

5 Функциональная спецификация интеллектуальных преобразователей

В настоящем разделе стандарта приведено описание функций интеллектуальных преобразователей ИИЭР 1451. Более детальное описание этих функций приведено в разделах 6—11.

5.1 Базовая модель комплекса стандартов ИИЭР 1451

В изображении базовой модели, приведенном на рисунках 1 и 2, показаны взаимосвязи между различными представителями комплекса стандартов ИИЭР 1451. Данная информация может быть использована для определения принадлежности к определенному стандарту комплекса. Данная информация не имеет своей целью продемонстрировать необходимое исполнение устройств, соответствующих данному стандарту. Дальнейшее описание элементов, представленных на рисунках 1 и 2, приведено в 5.1.1—5.1.15.

5.1.1 Пользовательская сеть

Пользовательская сеть не является предметом рассмотрения комплекса стандартов ИИЭР 1451. Характеристики данной сети определяются самим пользователем. Единственное требование сети

[1)] В оригинале ISO/IEC/IEEE 21450:2010 допущена ошибка. Ошибочно приведено значение Int16.

предъявляется к СПП, который должен иметь соответствующее программное и аппаратное обеспечение для работы с определенным сетевым интерфейсом. Для сетей, использующих различные средства связи и протоколы, требуются различные СПП.

5.1.2 Модуль сетевого доступа

Модуль сетевого доступа является частью СПП, предоставляющего интерфейс для пользовательской коммуникационной сети. Такая коммуникационная сеть не подпадает под действие какого-либо раздела стандарта комплекса ИИЭР 1451, и следовательно, функции данного модуля также не могут быть полностью определены каким-либо стандартом ИИЭР 1451. В стандарте ИИЭР 1451.1—1999 приведена логическая модель данного модуля, но при этом использование стандарта ИИЭР 1451.1—1999 для СПП не является обязательным для согласования с данным стандартом или любым другим стандартом комплекса ИИЭР 1451, основанным на данном стандарте.

5.1.3 Интерфейс сервисов интеллектуального преобразователя

В настоящем стандарте данный программный интерфейс определен как API. Детальное описание интерфейса приведено в разделе 10.

5.1.4 Сервисы СПП по ИИЭР 1451.0

В данном блоке определены функции и сервисы, предоставляемые модулю связи и приложениям СПП. Функции, выполняемые данным модулем, определены в настоящем стандарте и включают в себя набор команд и ЭТДП. В настоящий стандарт включены некоторые из характеристик, необходимых для поддержания таких функций датчиков, как срабатывание триггера и синхронная выборка (опрос). Детальное описание функций содержится в других стандартах данного комплекса.

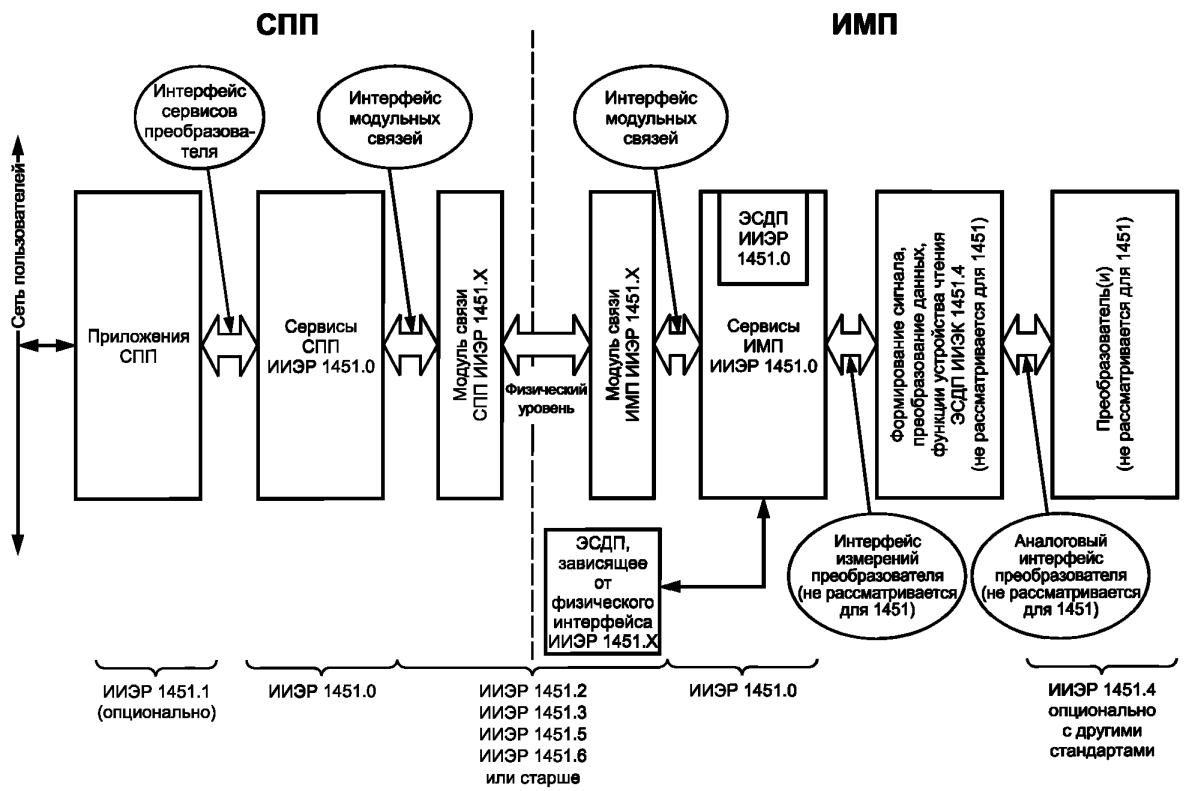


Рисунок 1 — Базовая модель

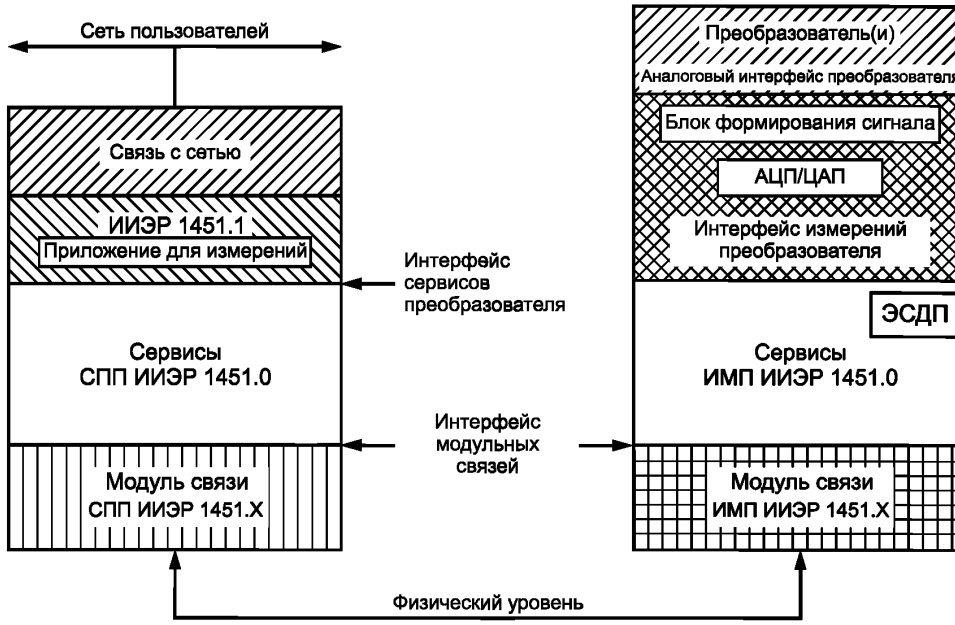


Рисунок 2 — Альтернативное описание базовой модели

5.1.5 Интерфейс модульных связей

В настоящем стандарте интерфейс модульных связей определяется в рамках API, использующегося для передачи информации между сервисами СПП ИИЭР 1451.0 и модулем связи СПП по ИИЭР 1451.X. Физические параметры данного интерфейса в настоящем стандарте не рассматриваются, они определяются изготовителями. Детальное описание данного интерфейса приведено в разделе 11.

5.1.6 Модуль связи СПП по ИИЭР 1451.X

Модуль связи СПП описывается в других стандартах комплекса ИИЭР 1451: например, ИИЭР 1451.2—1997, ИИЭР 1451.3—2003 и ИИЭР 1451.5—2007 [B4]. Данный модуль передает низкоуровневые пакеты протокола связи и прочие сервисы, необходимые для функционирования СПП в системе.

5.1.7 Интерфейс физического уровня

Интерфейс физического уровня между СПП и ИМП не описывается в настоящем стандарте, его описанию посвящены другие стандарты данного комплекса. В настоящем стандарте приведены некоторые функции данного интерфейса, как, например, синхронизация. Физическая реализация данных функций не представлена.

5.1.8 Модуль связи ИМП по ИИЭР 1451.X

Модуль связи ИМП по ИИЭР 1451.0/X является логическим дополнением модуля связи СПП по ИИЭР 1451.0/X, но не является идентичным данному модулю, так как существуют некоторые сервисы ИМП, отличные от тех, которые требуются для СПП. Как и в случае модуля связи СПП по ИИЭР 1451.0/X, данный модуль не определяется в настоящем стандарте, но определяется в других стандартах комплекса.

5.1.9 Сервисы ИМП по ИИЭР 1451.0

В настоящем стандарте описываются сервисы, исполняемые данной функцией. Большинство сервисов, предоставляемых СПП, имеют аналоги в данном блоке.

5.1.10 API измерений преобразователя

Данный интерфейс создается изготовителем и не является предметом рассмотрения данного стандарта.

5.1.11 Блок формирования сигналов. Функции преобразования данных

Параметры формирования сигналов и функций преобразования данных не являются предметом рассмотрения данного стандарта.

5.1.12 Аналоговый интерфейс преобразователя

Данный интерфейс является физическим интерфейсом между функцией формирования сигналов и функцией преобразования данных и не является предметом рассмотрения каких-либо стандартов комплекса ИИЭР 1451, за исключением стандарта ИИЭР 1451.4—2004.

5.1.13 Преобразователь

Преобразователь не является предметом рассмотрения каких-либо стандартов комплекса ИИЭР 1451.

5.1.14 ЭТДП

В настоящем стандарте определены основные ЭТДП и методы доступа к ним. Тем не менее информация, касающаяся физического интерфейса, и соответствующие ЭТДП определены в других стандартах комплекса ИИЭР 1451. Подробное описание ЭТДП приведено в разделе 8.

5.1.15 Применение стандарта ИИЭР 1451.4—2004 для базовой модели

Уровень ИИЭР 1451.0 задан для передачи цифровой информации между модулями системы. ЭТДП ИИЭР 1451.0 используются для полного описания ИМП, включая сам преобразователь, блок формирования сигналов и блоки преобразования данных. Передаваемая через такие интерфейсы аналоговая информация не является предметом рассмотрения стандартов настоящего комплекса. Тем не менее стандарт ИИЭР 1451.4—2004 дополняет аналоговый выход преобразователя ЭТДП. В этих ЭТДП приведены характеристики аналоговой информации, которая может быть получена от датчика ИИЭР 1451.4. Использование преобразователей ИИЭР 1451.4 не требуется ни одним из стандартов комплекса ИИЭР 1451, но допускается, что и показано в описании базовой модели. Совместное использование устройства ИИЭР 1451.4 и совместимого с ним устройства ИИЭР 1451.0 требует наличия соответствующего блока формирования сигналов для преобразователя и транслятора ЭТДП ИИЭР 1451.4. Транслятор ЭТДП требуется для согласования данных ЭТДП ИИЭР 1451.4 и характеристик блока формирования сигнала и позволяет полностью описать ИМП с помощью ЭТДП ИИЭР 1451.0. Оба типа ЭТДП описываются в стандартах данного комплекса, но описание метода совмещения ЭТДП не является предметом рассмотрения какого-либо из данных стандартов.

Для преобразователей ИИЭР 1451.4 часто используются интерфейсы, не соответствующие остальным стандартам комплекса ИИЭР 1451. В таком случае данная базовая модель не применяется.

5.2 Возможность автоматической конфигурации (plug-and-play)

ИМП и СПП, разрабатываемые для работы по настоящему стандарту, должны иметь возможность соединения с использованием совместимых физических средств связи и быть способны работать без каких-либо изменений системного программного обеспечения. При этом не должно быть необходимости в различных драйверах, профилях или прочих изменениях в программном обеспечении для выполнения базовых функций преобразователя. Данное утверждение не подразумевает приложений, использующих, создающих или отображающих относящиеся к преобразователю данные. Использование компонентов, выходящих за пределы рассмотрения настоящего стандарта, допускается, но такие компоненты могут оказывать влияние на способность конфигурации plug-and-play для ИМП или СПП.

5.3 Адреса

В настоящем стандарте применяется два уровня адресации. Первый уровень адресации связан с физическим уровнем и детально описан в стандартах комплекса ИИЭР 1451, определяющих средства связи. API модульных связей делает привязку данного адреса к «destId» (к «идентификатору получателя») с использованием «discoverDestination call» («запроса для определения получателя»), описанного в 11.3.13. На данном уровне адресации возможен обмен сообщениями (см. раздел 6) между ИМП и СПП или между модулями ИМП. Детальное рассмотрение данного вида адресации представлено в стандарте для связей физического уровня.

Второй уровень адресации — обращение к номеру канала преобразователя, присваиваемому заданному каналу преобразователя в рамках ИМП. Данные номера каналов преобразователя, имеющие длину в 16 битов, применяются в командных сообщениях в качестве номера канала-получателя (см. 6.1.2) или в ответных сообщениях (см. 6.1.4) в качестве номера канала-отправителя. Данный уровень используется для информирования ИМП о направлении отправки сообщения или для информирования СПП о том, откуда поступило сообщение в рамках ИМП. Правила, связанные с трактовкой номера канала преобразователя, приведены в таблице 5. Каждая строка данной таблицы содержит имя определенного класса адресов, используемое на протяжении всего стандарта.

Не всем командам можно присвоить любой класс адреса. Классы адресов, присваиваемые определенным командам, как и определения самих команд, приведены в разделе 7.

Таблица 5 — Правила использования номера канала преобразователя

Класс адреса	Значение	Описание
Глобальный	0xFFFF	К глобальной адресации относится особый адрес группы (GroupAddress), принадлежащий всем каналам ИМП
Адресная группа (AddressGroup)	0x8000<A≤0xFFFE	Как показано в таблице 6, существуют две адресные группы. Адресные группы с побитовым отображением применяются в случае, когда есть необходимость в небольшом числе адресных групп и желательна отправка команды к нескольким адресным группам одновременно. Двоичные адресные группы применяются в случае, когда нужно использовать большое число различных адресных групп. Адрес 0x8000 не используется
Канал преобразователя	1≤A≤0x7FFF	Адрес, старший значащий бит которого равен нулю. Оставшиеся 15 битов определяют канал преобразователя, которому направлено сообщение
ИМП	0	Адрес в нулевом сообщении означает, что сообщение предназначено для ИМП, а не для отдельного канала преобразователя

Таблица 6 — Адресные группы

Класс адреса	Значение	Описание
Адресные группы с побитовым отображением	0x8000<A≤0xBFFF	Установленное значение старшего значащего бита равно единице, следующего старшего значащего бита — нулю, и как минимум еще одного бита — единице. Данная последовательность значений двух старших значащих битов (битовая комбинация) сигнализирует, что это адрес группы. Оставшийся бит или биты указывают на группу (группы)
Двоичные адресные группы	0xC000≤A≤0xFFFE	Значения двух старших значащих битов установлены равными единице, а из оставшиеся 14 битов формируется двоичная последовательность, определяющая одиночную адресную группу (группу с одним адресом)

5.3.1 Глобальные адреса

Команды, отправленные на глобальный адрес, должны быть получены и выполнены всеми каналами преобразователя модулей ИМП, которым была адресована команда. В случае если полученная команда не выполняется ИМП или каналом преобразователя, команда должна быть проигнорирована без генерации сигнала ошибки.

5.3.2 Адреса адресных групп

Команды, отправленные адресным группам, должны быть исполнены всеми каналами преобразователя, которые опознаны как члены адресной группы. Определения команд адресной группы и способы их присваивания или удаления определенным адресным группам рассмотрены в 7.1.2.3. В случае если полученная команда не выполняется ИМП или каналом преобразователя, команда должна быть проигнорирована, при этом должен быть установлен бит «Отказ от выполнения» в ИПМ или канале преобразователя.

Использование адресных групп с несколькими модулями ИМП требует установки метода множественной адресации ИМП до присвоения адресных групп каналам преобразователя. Более подробно указанный метод присвоения описан в 10.1.3 и 10.2.4.

5.3.2.1 Адреса адресных групп с побитовым отображением

Каждая адресная группа содержит в себе старший значащий бит, равный единице, следующий старший значащий бит, равный нулю, и еще один бит, установленный в классе адреса в поле «Номер канала преобразователя». Битовая комбинация из единицы и нуля в двух старших значащих битах в поле «Номер канала преобразователя» сигнализирует, что адрес относится к адресной группе с побитовым отображением. При использовании побитовой операции «OR» («логическое ИЛИ») одна команда

может быть отправлена нескольким адресным группам одновременно. Возможна отправка сообщений 14 различным адресным группам.

5.3.2.2 Адреса двоичных адресных групп

Каждая адресная группа содержит в себе два старших значащих бита, равных единице. Оставшиеся 14 битов в поле «Номер канала преобразователя» данного класса адреса обозначают адресную группу. Возможна маркировка 16 383 различных адресных групп.

5.3.3 Номер канала преобразователя

Команды и ответные сообщения в поле «Номер канала преобразователя», имеющие номер от единицы до 0x7FFF включительно, адресованы каналу преобразователя. Данные команды и сообщения должны быть получены и выполнены тем каналом преобразователя, которому они адресованы. Если полученная команда не выполняется каналом преобразователя, она должна быть проигнорирована, и при этом должен быть сгенерирован сигнал ошибки, как предписано для отдельной команды.

5.3.4 Адреса ИМП

Команды и ответные сообщения, имеющие значение «ноль» в поле «Номер канала преобразователя», адресуются ИМП. Они должны быть получены и выполнены тем ИМП, который помечен при помощи «destId» («идентификатор получателя»). Команды и ответные сообщения предназначены непосредственно для интерфейсного модуля, а не для какого-либо канала преобразователя внутри ИМП. Если полученная команда не исполняется ИМП, она должна быть проигнорирована, и при этом должен быть сгенерирован сигнал ошибки, как предписано для отдельной команды.

5.4 Общие характеристики

Рабочие характеристики, общие для всех типов каналов преобразователя, определены в 5.4.1—5.4.4.

5.4.1 Рабочие режимы

На рисунке 3 приведена диаграмма высокоуровневого состояния канала преобразователя. После инициализации канала преобразователя существуют два основных рабочих режима. После инициализации канал преобразователя входит в «режим ожидания преобразователя». Большую часть команд канал преобразователя получает в режиме ожидания. В рабочий режим канал преобразователя входит только после получения команды «Transducer Channel Operate» («Перевести канал преобразователя в рабочий режим») (см. 7.1.4.1) и остается в этом режиме до получения команды «Transducer Channel Idle» («Перевести канал преобразователя в режим ожидания») (см. 7.1.4.2).

Как показано на рисунке 4, ИМП может находиться в трех состояниях. ИМП может войти в состояние инициализации при получении команды «Reset» («Перезагрузка») (см. 7.1.7.1) или в случае подачи питания. После завершения процесса инициализации ИМП переходит в активное состояние. В спящий режим ИМП переходит при получении команды «TIM Sleep» («Перевести ИМП в спящий режим») (см. 7.1.6.2). Единственной командой, которую ИМП должен принимать в спящем режиме, является команда «TIM Wake-up» («Вывести ИМП из спящего режима»), переводящая модуль в активное состояние.

Подробное описание команд, принимаемых каналом преобразователя в каждом из режимов, приведено в разделе 7.

5.4.2 Включение и отключение триггеров

В канале преобразователя может быть предусмотрена возможность включения или выключения триггеров при помощи команд. Если не указано обратное, все операции, описанные в настоящем стандарте, рассматриваются для канала преобразователя с включенными триггерами.

5.4.3 Диагностика

Одной из главных причин ввода в эксплуатацию интеллектуальных преобразователей является их способность к самостоятельной проверке и диагностике. В настоящем стандарте представлен механизм запуска процесса диагностики, но требований к логике диагностики не предъявляется.

5.4.4 Структуры хранения и передачи данных

В настоящем стандарте используются три структуры для хранения и передачи данных: набор данных, сообщение и пакет. Каждая из этих трех структур описывается в нижеследующих подпунктах. Приложения уровня выше стека (набора) протоколов работают с наборами данных. Более высокие уровни стека протокола работают с сообщениями. Если в наборе данных содержится больше байтов, чем может быть передано в одном сообщении, то он должен быть разбит приложением на несколько сообщений.

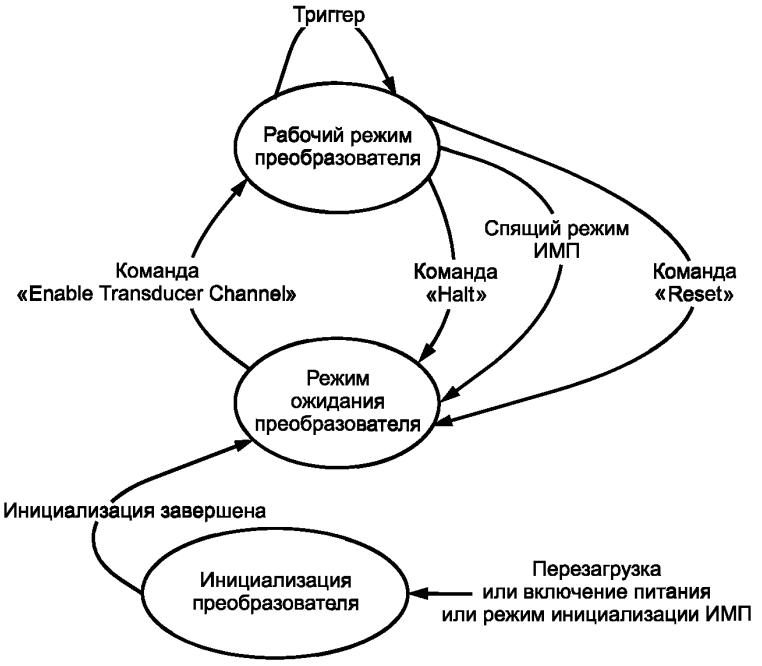


Рисунок 3 — Режимы работы канала преобразователя

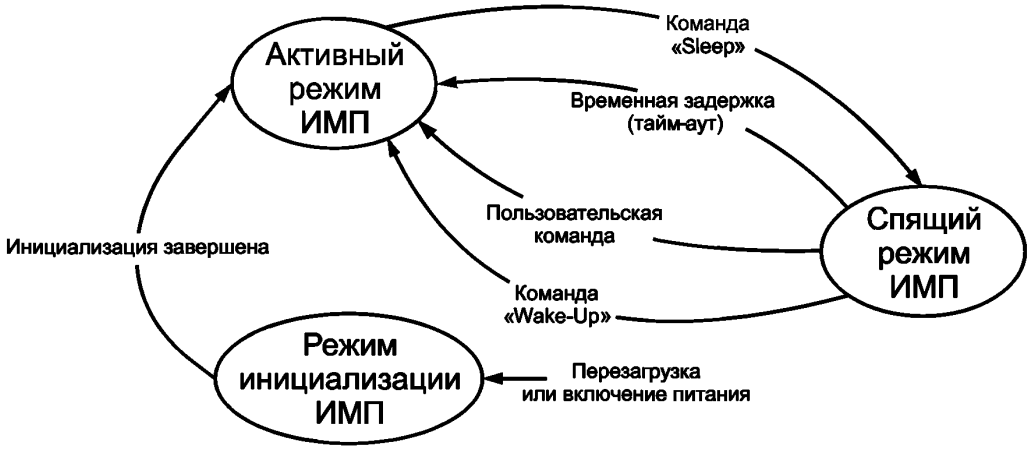


Рисунок 4 — Режимы работы интерфейсного модуля преобразователя

5.4.4.1 Наборы данных

Все каналы преобразователя работают с наборами данных. Набор данных определяется тремя полями ЭТДП канала преобразователя. Поле «Максимальное повторение данных» (см. 8.5.2.28) определяет максимальное число отдельных выборок данных из набора данных. Фактическое число выборок может быть уменьшено по сравнению с числом выборок в поле «Максимальное повторение данных» за счет установки дополнительной команды «Set Transducer Channel data repetition count» («Установить число повторений данных канала преобразователя») (см. 7.1.2.1). При перезагрузке или исходной подаче питания используются значения по умолчанию. В случае если число повторения данных является программируемым, значение по умолчанию должно быть равно нулю. Если число повторения данных не является программируемым, значение по умолчанию должно быть таким же, как и значение

поля «Максимальное повторение данных» ЭТДП. Второе поле — поле «Шаг дискретизации серии» (см. 8.5.2.30). Данное поле используется для определения интервалов между данными и может быть изменено заложенной изготовителем командой или встроенным исполнительным механизмом. Третье поле — поле «Единицы измерения серии» (см. 8.5.2.31). Данное поле определяет единицы измерения для поля «Шаг дискретизации серии». Данное поле создано таким образом, что единицами измерения шага дискретизации серии не обязательно должны быть значения времени, и в таком случае временные интервалы между данными могут быть неодинаковыми.

Пример — Единица измерения серии — кельвин, а шаг дискретизации серии составляет 0,5. При такой комбинации датчик будет собирать данные с интервалом в 0,5 К. И таким образом, данные будут получены с равным температурным интервалом вместо равного временного интервала.

5.4.4.2 Сообщения и пакеты

Сообщения могут содержать до 65 535 байтов, не считая байтов в заголовках. Тем не менее канал передачи данных и физические уровни стека протоколов работают с пакетной передачей данных. Максимальная длина пакетов определяется в стандарте для физических уровней и каналов данных. Если сообщение не помещается в один пакет, то на уровне канала данных стека протоколов оно разбивается на несколько пакетов для передачи.

5.5 Электронная таблица данных преобразователя (ЭТДП, TEDS)

ЭТДП являются блоками информации, представленными в одном из форматов, определенных в разделе 8. ЭТДП должны храниться в энергонезависимой памяти ИМП. Тем не менее в случаях, если определенные применения не позволяют выполнить данное требование, стандарт позволяет хранение ЭТДП в других местах внутри пользовательской системы. В случае если ЭТДП хранится не в ИМП, она называется «виртуальной» ЭТДП. Изготовитель канала передатчика может поставлять виртуальные ЭТДП, чтобы не хранить их в ИМП. В этом случае задача пользовательской системы заключается в обеспечении взаимодействия между такой гарантированно доступной и находящейся в ИМП информацией, как УУИД, и файлом, содержащим виртуальную ЭТДП в системе. Сервис предоставляется СПП или главным процессором. ИМП может одновременно использовать различные ЭТДП: хранящиеся непосредственно в ИМП и виртуальные. Ответ на команду «Query TEDS» («Запросить ЭТДП») (см. 7.1.1.1) содержит указатели (флажки), которые сигнализируют, является ли ЭТДП поддерживаемой и является ли она виртуальной. Виртуальная ЭТДП считается «поддерживаемой» в случае, если изготовителем предоставлен файл с соответствующей информацией на совместимом носителе, не принимая во внимания факт загрузки данного файла в систему пользователем.

Как правило, содержание ЭТДП, записанное изготовителем или пользователем в первый раз, не меняется. Тем не менее существует возможность создания каналов преобразователя, изменяющих содержание ЭТДП во время работы. Необходимость изменения содержания ЭТДП может быть вызвана наличием встроенного исполнительного устройства, которое требует изменения информации ЭТДП, ранее считаемой постоянной. Изменения могут быть вызваны логикой построения ИМП или канала преобразователя, если изменения внутри устройства требуют изменений в ЭТДП. Для проведения таких изменений в атрибутах ЭТДП имеется бит «Adaptive» («Адаптивность»), который задает возможность изменения внутренними логическими операциями ИМП или канала преобразователя. Если содержание ЭТДП изменяется без записи (сохранения), устанавливается бит «Status» («Статус»).

Четыре вида ЭТДП являются обязательными для всех ИМП, остальные виды обязательными не являются. Список обязательных ЭТДП следующий:

- мета-ЭТДП;
- ЭТДП канала преобразователя;
- ЭТДП с именем преобразователя, заданным пользователем;
- ЭТДП физического уровня.

Технические характеристики всех видов ЭТДП приведены в разделе 8 настоящего стандарта.

5.5.1 Обязательные ЭТДП

Технические характеристики обязательных ЭТДП и их назначения приведены в 5.5.1.1—5.5.1.4.

5.5.1.1 Мета-ЭТДП

Мета-ЭТДП содержит ряд аварийных временных параметров, доступных СПП для установления значений времени ожидания (тайм-аута) в коммуникационном программном обеспечении, что позволяет определить, когда ИМП не отвечает. Оставшаяся часть ЭТДП содержит описание связей между каналами преобразователя внутри ИМП. Технические характеристики данного вида ЭТДП представлены в 8.4.

5.5.1.2 ЭТДП канала преобразователя

В ЭТДП канала преобразователя содержится подробная информация об определенном преобразователе: измеряемый или контролируемый физический параметр, рабочий диапазон канала преобразователя, характеристики цифрового входа-выхода, рабочие режимы блока и временные характеристики. Технические характеристики данного вида ЭТДП представлены в 8.5.

5.5.1.3 ЭТДП с именем преобразователя, заданным пользователем

ЭТДП с именем преобразователя, заданным пользователем, предоставляет пользователю преобразователя место для хранения имени, по которому система сможет опознавать преобразователь. Рекомендуется составлять ЭТДП в соответствии с приведенной в настоящем стандарте структурой, однако данное требование не является обязательным, так как содержание ЭТДП определяется самим пользователем. При этом изготовителю ИМП необходимо предоставить область энергонезависимой памяти, в которой пользователь может создать запись, используя стандартные методы доступа к ЭТДП. Технические требования к данного вида ЭТДП представлены в 8.11[1)].

Примечание — Предполагается, что ЭТДП с именем преобразователя, заданным пользователем, должна поддерживать «Object Tags» («объектные метки»), как определено в стандарте ИИЭР 1451.1—1999, или другие подобные применения.

5.5.1.4 ЭТДП физического уровня

ЭТДП физического уровня зависит от физических средств связи, используемых для связи ИМП и СПП. Данная ЭТДП не является предметом рассмотрения настоящего стандарта, в стандарте рассмотрены только методы доступа к ней.

5.5.2 Необязательные ЭТДП

Технические характеристики и назначение необязательных ЭТДП приведены в 5.5.2.1—5.5.2.10.

5.5.2.1 ЭТДП калибровки

ЭТДП калибровки содержат калибровочные постоянные, необходимые для преобразования сигнала на выходе датчика в инженерные единицы измерения или для преобразования значений, заданных в инженерных единицах измерения, в форму, требуемую для исполнительного устройства. Для осуществления таких преобразований поддерживаются два метода. Первый — обычный линейный метод с применением формулы общего вида $y = mx + b$. Второй — метод с использованием формулы еще более общего вида, имеющей универсальное применение. Технические характеристики данного вида ЭТДП приведены в 8.6.

5.5.2.2 ЭТДП частотной характеристики

В ЭТДП частотной характеристики используется таблица для задания частотной характеристики канала преобразователя. Технические характеристики данного вида ЭТДП приведены в 8.7.

5.5.2.3 ЭТДП передаточной функции

ЭТДП передаточной функции описывает способ объединения нескольких отдельных передаточных функций для описания частотной характеристики канала преобразователя в алгоритмической форме. Технические характеристики данного вида ЭТДП приведены в 8.8.

5.5.2.4 Текстовые ЭТДП

Текстовые ЭТДП принадлежат комплексу ЭТДП, предоставляющих текстовую информацию об ИМП или канале преобразователя. Такие ЭТДП могут быть составлены на одном или более языках. Они состоят из директории, облегчающей доступ к определенному языковому подблоку внутри ЭТДП, за которым следуют блоки ЭТДП, составленные с учетом требований языка XML. Технические характеристики данного вида ЭТДП приведены в 8.9.

5.5.2.5 Командная ЭТДП

Командная ЭТДП относится к текстовым ЭТДП, предоставляющим изготовителю способ установить дополнительные команды, помимо уже включенных в стандарт. Данные команды в первую очередь предназначены для того, чтобы при необходимости настроить определенные преобразователи и обработчики сигналов. Пример схемы подобной ЭТДП приведен в приложении D, пример командной ЭТДП приведен в приложении Н.

5.5.2.6 Идентификационные ЭТДП

В стандарте ИИЭР 1451.2—1997 определены ЭТДП мета-идентификации, ЭТДП идентификации канала, а также ЭТДП идентификации калибровки. Данные ЭТДП предназначены для того, чтобы предоставлять текстовую информацию об интеллектуальном ИМП, канале преобразователя и калибровке. В настоящем стандарте все виды подобных ЭТДП были объединены в категорию текстовых ЭТДП, а не определены

[1)] В оригинале ISO/IEC/IEEE 21450:2010 допущена ошибка. Ошибочно приведена ссылка на 5.5.1.3.

отдельно. Схема в приложении D и примеры в приложениях E, F и G могут быть использованы для дублирования информации о предназначении таких ЭТДП, определенных в стандарте ИИЭР 1451.2—1997. Наименования ЭТДП в стандарте ИИЭР 1451.2—1997 соответствуют их наименованиям в настоящем стандарте.

5.5.2.7 ЭТДП географического места нахождения

ЭТДП географического места нахождения относятся к текстовым ЭТДП, содержащим информацию о статичном географическом месте нахождения. Данные о месте нахождения, в котором установлен ИМП, вводятся пользователем. Содержание данной ЭТДП должно соответствовать требованиям к текстовой ЭТДП приведенным в 8.9. Блок данных должен быть написан на языке географической разметки (Geography Markup Language, GML), разработанном открытым геопространственным консорциумом (Open Geospatial Consortium, OGC) и описанном в стандарте ИСО 19136.

ЭТДП географического места нахождения устанавливает данные о месте нахождения датчика, которое на языке GML должно быть задано в соответствии со схемой для точки: «gml:point».

Схема «Gml:point» является частным случаем схемы «gml:PointType», включенной в общую схему «GML: geometryBasic0d1d.xsd».

Схема «GML geometryBasic0d1d.xsd» идентифицируется следующим URI: urn:opengis:specification:gml:schema-xsd:geometryBasic0d1d:v3.1.0.

Применение схемы «gml:point» в настоящей спецификации должно быть ограничено следующими способами:

- схема «gml:point» должна использовать «DirectPositionType», то есть «gml:pos»;
- схема «gml:point» не должна использовать «gml:coordinates» или «gml:coord»;
- схема «gml:pos» должна включать атрибут «gml:SRSReferenceGroup».

Примечание — Величины «gml:pos» ограничены системой отсчета координат (Coordinate Reference System, CRS), определенной в опорной группе SRSReferenceGroup:

- «gml:SRSReferenceGroup» должна включать «srsName»;
- «gml:SRSReferenceGroup» может включать «srsDimension»;
- «gml:SRSReferenceGroup» может включать «gml:SRSInformationGroup».

В данной спецификации для «srsName» должно быть присвоено имя из набора возможных имен Унифицированного ресурса имен (Uniform Resource Name, URN). Пример CRS для Мировой геодезической системы 1984 года (World Geodesic System, WGS 84) выглядит следующим образом:

urn:ogc:def:crs:OGC:1.3:CRS84

В системе WGS 84 CRS содержит данные о географической широте, затем данные о географической долготе; при этом ось X соответствует широте, а ось Y соответствует долготе.

Предполагается, что изготовитель предоставляет место для данной ЭТДП, а вся актуальная информация для размещения вводится пользователем. Требуемый объем памяти зависит от выбранного типа места нахождения для языка GML. Настоящий стандарт не регламентирует объем памяти, который необходимо выделить для данного вида ЭТДП. Тем не менее объем памяти стандартного языкового блока данной ЭТДП составляет от 60 до 80 байтов при использования одного байта для каждого символа.

Для упрощения работы с ЭТДП географического места нахождения следует использовать «GML Point» («Точка языка географической разметки GML»). Данная точка обозначает место нахождения датчика непосредственно после его установки, при этом также должна быть определена система отсчета координат (CRS). Если система отсчета координат не определена, координаты места нахождения точки могут иметь разброс до 100 м из-за разницы начала координат двух географических систем CRS. Поэтому при записи данной ЭТДП рекомендуется включать в нее идентификатор системы отсчета координат (CRS), как определено для «GML Point» («Точки языка географической разметки GML»).

5.5.2.8 ЭТДП с расширенным набором единиц измерения

В некоторых случаях, в особенности при использовании химических датчиков, невозможно полностью выразить физические величины в системе СИ. Например, датчик измеряет процент содержания определенного химического соединения в образце и работает с соотношением величин, соответствующим нумерации 1 в поле «Тип величины», а сам тип измеряемой величины — моль. Разрешить это противоречие возможно при использовании данной текстовой ЭТДП, которая предоставляет место для включения текста, расширяющего ограниченный набор величин в системе СИ.

5.5.2.9 ЭТДП для специальных приложений конечного пользователя

ЭТДП для специальных приложений конечного пользователя подобна ЭТДП с именем преобразователя, заданным пользователем, так как она является блоком памяти, в котором пользователь может хранить любую информацию по собственному усмотрению в любом формате. Технические характеристики данного вида ЭТДП приведены в 8.10.

5.5.2.10 ЭТДП, заданная изготовителем

ЭТДП данного типа включена в стандарт для того, чтобы дать изготовителям возможность задавать ЭТДП, не включенные в настоящий стандарт. Область применения и структура ЭТДП, заданных изготовителем, полностью определяются самим изготовителем. Изготовитель не обязан делать такие ЭТДП доступными для пользователей. Технические характеристики данного вида ЭТДП приведены в 8.12[1)].

5.6 Описание типов канала преобразователя

В данном подразделе определены основные свойства трех видов каналов преобразователей. Атрибуты ЭТДП каналов преобразователей описывают свойства различных каналов преобразователей в системе. Более подробные временные и контрольные параметры этих каналов преобразователей приведены в нижеследующих подразделах настоящего стандарта. Существуют следующие виды каналов преобразователя:

- датчики;
- датчики событий;
- исполнительные устройства.

Термин «канал преобразователя» в данном подразделе применяется к физическому преобразователю и ко всем электронным устройствам, обеспечивающим преобразователю выполнение связующих функций.

5.6.1 Датчик

Датчик должен измерять какой-либо физический параметр и отправлять обратно цифровые данные для этого параметра. При срабатывании триггера, если функция триггера активна, датчик начинает сбор и хранение наборов данных внутри ИМП. Величина временных интервалов между отдельными отсчетами в наборе данных должна контролироваться ИМП и является функцией рабочего режима датчика. Датчик в рабочем режиме преобразователя должен отвечать на команды «Read Transducer Channel data-set segment» («Считать сегмент набора данных канала преобразователя») (см.7.1.3.1), отправляя в ответ соответствующий набор данных. Если новый набор данных недоступен, канал преобразователя должен отправить в ответ те же самые данные, которые были переданы при предыдущей команде «Read Transducer Channel data-set segment» («Считать сегмент набора данных канала преобразователя»).

5.6.2 Датчик событий

В отличие от обычного датчика датчик событий не определяет величину какого-либо физического параметра, а определяет изменение состояния. Такое изменение состояния может быть аналоговым сигналом, переходящим определенный порог, или набором дискретных битов, совпадающим или не совпадающим с заданной битовой комбинацией. Таким образом, выходной сигнал датчика событий отражает состояние на его входе. Существуют всего два допустимых состояния: ноль и единица. На выходе датчика событий может быть получена информация двух видов:

- текущее состояние на входе;
- время, в которое произошло изменение состояния.

Для датчика событий действительно такое же определение ЭТДП, как и для любого другого преобразователя.

5.6.2.1 Выходной сигнал датчика события

Модель данных на выходе датчика события определена в ЭТДП канала преобразователя аналогично модели данных на выходе для любого другого датчика. Тем не менее величина выходного сигнала должна быть равна нулю или единице.

5.6.2.2 Время события

Датчик событий сообщает только о факте произошедшего события. Для определения времени, в которое событие произошло, необходимы другие элементы системы. Существует множество путей решения данной задачи, и выбор определенного метода зависит от требований конкретного приложения. Если для определения времени события используется метод опроса, время события может быть определено только в пределах интервала опроса. В режиме потоковой передачи данных СПП может определить время события по времени поступления сообщения. Главным фактором в определении задержки во времени между самим событием и его распознаванием СПП является физический уровень. Время, в которое произошло событие, может быть определено в пределах одной микросекунды или менее. Для этого совместно с датчиком событий применяются встроенные в ИМП датчики временных

[1)] В оригинале ISO/IEC/IEEE 21450:2010 допущена ошибка. Ошибочно приведена ссылка на 8.11.2.3.

интервалов или датчики временного события (TimeInstance). Обсуждение датчиков временных интервалов и датчиков временного события представлено в М.2 и М.3 (приложение М).

5.6.2.3 Аналоговые датчики событий

Аналоговые датчики событий получают аналоговый входной сигнал. Событие считается произошедшим, когда величина входного сигнала переходит пороговое значение. Для аналоговых датчиков событий определены верхнее и нижнее пороговые значения. Нижний порог отличается от верхнего на величину гистерезиса. Значение гистерезиса должно быть более или равным нулю. Во время стабильной работы повышающий переход (то есть условия на входе датчика, вызывающие изменение сигнала на выходе со значения «ноль» до значения «один») должен происходить при значении «ноль» сигнала на выходе и когда значение аналогового сигнала на входе переходит верхнее пороговое значение. Понижающий переход (то есть условия на входе датчика, вызывающие изменение сигнала на выходе со значения «один» до значения «ноль») должен происходить при значении «один» сигнала на выходе и когда значение аналогового сигнала на входе переходит нижнее пороговое значение. Графически такие переходы изображены на рисунке 5. Изменения состояния на выходе должны регистрироваться только после того, как устройство получило запускающий импульс. Верхнее пороговое значение и величина гистерезиса могут быть фиксированы при изготовлении датчика событий или могут быть программируемыми. Для того чтобы сделать верхний порог или гистерезис программируемыми, рекомендуется использовать встроенные исполнительные устройства для установки их значений.

Несмотря на то что инициализация датчика событий зависит от его устройства, рекомендуется провести описанную ниже процедуру. Выходной сигнал с датчика событий, сообщающего только о понижающих переходах, следует инициализировать в состояние «один», если значение аналогового входного сигнала больше или равно значению верхнего порога. Если значение входного сигнала меньше значения верхнего порога, то выходной сигнал следует инициализировать в состояние «ноль». Выходной сигнал с датчика событий, сообщающего только о повышающих переходах, следует инициализировать в состояние «ноль», если значение входного аналогового сигнала меньше или равно значению нижнего порога. Если значение на входе больше значения нижнего порога, то выходной сигнал следует инициализировать в состояние «один». Для датчиков событий, сообщающих об обоих переходах, следует выбрать один из вышеуказанных методов. При этом следует обратить внимание на то, что если значение входного сигнала находится в пределах гистерезиса, то первый переход может дать сбой. Этого можно избежать путем контроля за величиной входного аналогового сигнала во время инициализации.

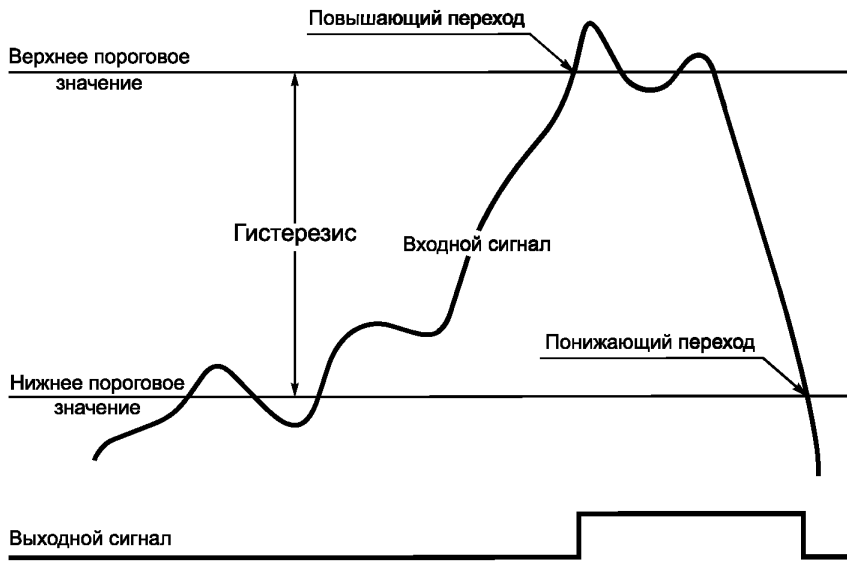


Рисунок 5 — Характеристика аналогового датчика событий

5.6.2.4 Цифровые датчики событий

Цифровые датчики событий имеют один или более дискретных входных сигналов. Событие считается произошедшим, когда входной сигнал совпадает с определенной последовательностью за определенный фиксированный изготовителем период времени. Повышающий переход должен произойти, когда входные сигналы совпадают с определенной цифровой последовательностью и данное совпаде-

ние длится фиксированный период времени. Данный период времени определяется изготовителем для соответствия характеристикам события, но может также контролироваться при помощи заложенных изготовителем уникальных команд или при помощи встроенного исполнительного устройства. Понижающий переход должен произойти, когда входные сигналы не совпадают с определенной последовательностью за фиксированный период времени.

Цифровой датчик событий может иметь несколько связанных встроенных каналов преобразователя, а также может включать в себя датчик(и), возвращающий(ие) входной цифровой сигнал по команде, и исполнительные устройства, позволяющие сконфигурировать сложный датчик событий. Встроенные исполнительные устройства могут быть использованы для определения последовательностей, масок, временных задержек и комбинаций из нескольких дискретных входных сигналов, объединенных в одно событие. ЭТДП таких встроенных каналов преобразователя должны соответствовать их функционалу. Функционал каждого встроенного канала преобразователя определяется контрольными группами в мета-ЭТДП, которые представлены в 8.4.2.8.

5.6.2.5 Сообщения о переходах состояний

Датчик событий может быть разработан для сообщений о повышающих либо понижающих переходах или об обоих вариантах переходов. Команда «Edge-to-Report» («Пороговое значение для посылки отчета») (см. 7.1.2.9) позволяет системе выбирать, какие переходы считать событиями. Возможности команды «Edge-to-Report» («Пороговое значение для посылки отчета») для датчика событий наряду с условиями по умолчанию определены в разделе ЭТДП канала преобразователя (см. 8.5). Если датчик событий находится в рабочем режиме преобразователя в момент времени, когда происходит событие, он должен установить бит «data/event» («данные/событие») в регистре состояний и отправить сервисное сообщение запроса, если протокол состояний событий подключен (см. 7.1.1.11). Если протокол состояний событий отключен, бит «data/event» («данные/событие») в регистре состояния устанавливается, но сообщение не отправляется.

Проверка бита состояния «data/event» («данные/событие») представлена в 5.13.6 и позволяет определить, произошло ли событие с момента последнего считывания состояния.

5.6.2.6 Состояние датчика событий

Датчик событий должен установить бит состояния «data/event» («данные/событие») канала преобразователя сразу после того, как событие произошло, независимо от того, находится преобразователь в рабочем режиме или режиме ожидания. Если датчик событий находится в рабочем режиме преобразователя, но при этом датчиком не был получен запускающий импульс, должен быть установлен бит состояния «missed data or event» («пропущенные данные/событие») канала преобразователя одновременно с любым произошедшим событием. Бит состояния «missed data or event» («пропущенные данные/событие») канала преобразователя не устанавливается, если датчик событий работает с преобразователем в режиме ожидания.

ИМП может проводить проверки на согласованность в случае, если битовая комбинация цифрового датчика событий является изменяемой или если являются изменяемыми значение верхнего порога или гистерезиса аналогового датчика событий. При наличии несогласованных величин устанавливается бит состояния «hardware error» («ошибка аппаратного обеспечения») (см. 5.13.7). Проверка несогласованных величин, если она проводится, должна быть проведена сразу после изменения любого из этих параметров. Проверка должна состоять из верификации следующих отношений:

Для аналоговых величин:

максимальное измеряемое значение на входе > верхний порог ≥ (верхний порог минус гистерезис) ≥ минимальное измеряемое значение на входе.

Для цифровых последовательностей (комбинаций):

последовательность является разрешенной комбинацией цифровых сигналов.

Примечание — При проверке на согласованность ИМП должен использовать только данные из модели данных, содержащейся в ЭТДП канала преобразователя для встроенных каналов преобразователя.

5.6.2.7 Датчики событий с непрерывной выборкой

Датчики событий, также как прочие датчики и исполнительные устройства, могут работать в режиме непрерывной выборки (см. 5.10.1.6). В режиме непрерывной выборки датчик событий, находящийся в рабочем режиме преобразователя, должен регистрировать и помещать в буфер информацию о том, что событие произошло. Триггеры в результате события не запускаются.

5.6.2.8 Датчики событий в режиме потоковой передачи данных

Датчики событий, также как прочие датчики и исполнительные устройства, могут работать в режиме потоковой передачи данных. При потоковой передаче данных, при буфере, работающем в полном ре-

жиме (см. 5.10.2.2), датчик событий в рабочем режиме преобразователя должен передавать сообщение каждый раз после того, как событие произошло. При потоковой передаче в режиме с фиксированным интервалом передачи данных (см. 5.10.2.3) датчик событий должен хранить информацию о произошедшем событии или событиях в буфере и передавать информацию из буфера в соответствующее время.

5.6.2.9 Время между событиями

Минимальное время между событиями, которые может обнаружить датчик событий, зависит от устройства датчика событий. Минимальное время между событиями, которые может обнаружить датчик событий, должно быть указано в поле «Период дискретизации канала преобразователя» ЭТДП канала преобразователя (см. 8.5.2.36).

5.6.3 Исполнительное устройство

Исполнительное устройство вызывает физическое или внутреннее выходное действие. Когда происходит запускающее триггерное событие, состояние на выходе исполнительного устройства меняется для приведения в соответствие определенному набору данных. Если в наборе данных содержится больше одного заданного значения, то интервал между отработкой отдельных входных значений должен контролироваться ИМП.

Исполнительное устройство может быть спроектировано таким образом, что запись набора данных самим устройством перед совершением действия не требуется. Такой тип устройств всегда либо использует набор данных по умолчанию, либо вообще не использует набор данных и производит заранее определенное действие при получении запускающего сигнала.

5.7 Встроенные каналы преобразователя

Встроенные каналы преобразователя — это каналы преобразователя, функции которых выполняются исключительно внутри ИМП. Встроенный канал преобразователя не воспринимает и не контролирует какие-либо функции вне ИМП. К примеру, встроенные исполнительные устройства могут использоваться для установки значения порога или гистерезиса датчика событий. Встроенный цифровой датчик событий может быть настроен на обнаружение и сообщение о любых изменениях состояния ИМП. Встроенные каналы преобразователя распознаются системой как обычные каналы преобразователя. Они отвечают на команды, имеют ЭТДП и учитываются при подсчете числа каналов преобразователя ИМП.

Примечание — Преимущество использования «встроенных» каналов преобразователя над контрольными командами заключается в том, что «встроенные» каналы преобразователя снабжены ЭТДП, предоставляющими всей системе стандартный способ получения данных о работе ИМП. Их стоит использовать, когда возникают трудности в отслеживании этапов выполнения контрольной команды стандартным способом.

5.8 Группы каналов преобразователя

В настоящем стандарте определены две группы каналов преобразователя: «ControlGroups» («Контрольные группы») и «VectorGroups» («Векторные группы»). Они сходны в реализации, но используются для двух различных функций.

5.8.1 Контрольные группы

Контрольные группы применяются для определения набора каналов преобразователя, когда один из каналов преобразователя является главным, а оставшиеся каналы группы либо обеспечивают дополнительную информацию о главном канале преобразователя, либо используются для контроля определенного аспекта главного канала преобразователя. Например, при помощи контрольной группы можно определить до трех дополнительных каналов преобразователя, связанных с аналоговым датчиком событий. Первый канал — датчик, измеряющий значения аналоговых входных сигналов для датчика событий. Второй канал — исполнительное устройство, устанавливающее порог для датчика событий. Третий — исполнительное устройство, устанавливающее значение гистерезиса для датчика событий.

5.8.2 Векторные группы

Векторные группы применяются для определения взаимоотношений между каналами преобразователя внутри одного мультиканального ИМП, обеспечивающего отображение или математическое взаимодействие между каналами преобразователя. Например, при помощи векторной группы можно определить взаимоотношения между компонентами трехосевого акселерометра. Векторные группы используются программным обеспечением СПП или главным процессором для группирования выходных сигналов отдельных каналов преобразователя в векторы с целью отображения или вычислений.

Примечание — Векторные группы в некотором роде аналогичны прокси-каналам преобразователя. Прокси-каналы преобразователя применяются для определения каналов преобразователя, сгруппированных для

более эффективной передачи данных и/или одновременного запуска. Прокси-каналы могут представлять из себя или не представлять векторные группы для отображения или вычисления. Тем не менее рекомендуется реализовывать все векторные группы как прокси-каналы, особенно в пространственных векторных приложениях (таких как скорость или ускорение), где измерения трех компонент в различных точках и во времени могут привести к неверной интерпретации.

5.9 Прокси-канал преобразователя

Прокси-канал преобразователя — искусственная конструкция, используемая для объединения множества выходов датчиков или множества входов исполнительных устройств в единую структуру. Прокси-канал преобразователя имеет номер канала преобразователя, который может быть считан или записан, но прочих характеристик канала преобразователя не имеет. Это означает, что прокси-канал не имеет ЭТДП канала преобразователя, ЭТДП калибровки, ЭТДП передаточной функции или ЭТДП частотной характеристики. Прокси-каналы преобразователя, существующие внутри ИМП, определяются в мета- ЭТДП (см. 8.4.2.14).

Прокси-канал преобразователя, как и любой другой канал преобразователя, имеет постоянный номер канала преобразователя, присвоенный изготовителем. Следовательно, ИМП может обращаться к нему по этому адресу тем же образом, которым он обращается ко всем остальным связанным с ним каналам преобразователя.

Прокси-канал не должен включать в себя несовместимые типы преобразователей. Иными словами, прокси-канал представляет собой группу датчиков или группу исполнительных устройств, но не должен представлять группу, содержащую оба вида преобразователей.

Возможность записи или считывания данных отдельных представителей прокси-канала может быть разрешена или не разрешена изготовителем по его собственному усмотрению. Если изготовитель запрещает возможность считывания или записи отдельных устройств прокси-канала, то ответом на одну из таких команд в слове состояния канала преобразователя должен быть записан соответствующий бит «command rejected» («отказ от выполнения команды») (см. 5.13.4).

Существуют два метода комбинирования наборов данных для прокси-канала: метод «block» (метод «блоков») и метод «interleave» (метод «чередования»). Оба эти метода изображены графически на рисунке 6. Метод «блоков» позволяет использование наборов данных разной длины. При использовании метода «чередования» наборы данных должны иметь одинаковый размер.

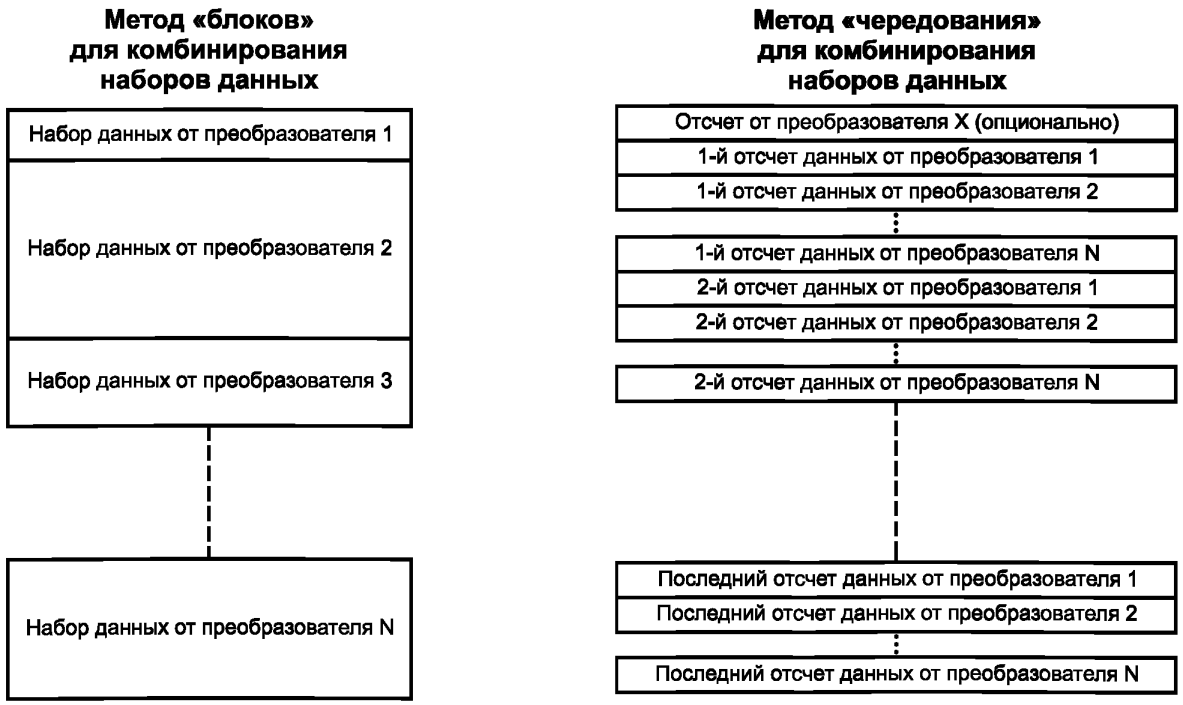


Рисунок 6 — Методы комбинирования наборов данных

5.10 Атрибуты и рабочие режимы

В 5.10.1—5.10.7.3 определены рабочие режимы, характеризующиеся атрибутами, содержащимися в ЭТДП канала преобразователя. Каждый атрибут показывает, поддерживает ли канал преобразователя соответствующий рабочий режим. Некоторые из этих режимов являются взаимоисключающими, а некоторые могут выполняться одновременно.

5.10.1 Режимы выборки данных

Датчик или исполнительное устройство могут работать в одном из пяти режимов выборки данных. В каждом режиме выборки данных существуют различные взаимоотношения между срабатыванием триггера и выборкой датчиком или приложением, обрабатывающим собранные данные и передающим сигнал на выход исполнительного устройства. Режим или режимы работы канала преобразователя определяются атрибутами, приведенными в ЭТДП канала преобразователя. Допустимые значения данных атрибутов представлены в 8.5.2.44.

Режимы «Trigger-initiated» («Выборка данных по срабатыванию триггера») и «Free-running» («Автономный режим выборки данных») являются взаимоисключающими. Канал преобразователя должен работать в одном из двух этих режимов. Оставшиеся три режима представляют собой вариации этих двух основных рабочих режимов.

5.10.1.1 Режим «Trigger-initiated» («Выборка данных по срабатыванию триггера»)

В режиме «Trigger-initiated» («Выборка данных по срабатыванию триггера») датчик должен начинать запрашивать набор данных сразу после получения запускающего сигнала триггера. Исполнительное устройство должно начинать выдавать данные сразу после получения запускающего сигнала триггера. Обработка выборок данных продолжается до тех пор, пока все данные из набора данных не будут обработаны с частотой, определенной ИМП, после чего обработка прекращается.

5.10.1.2 Режим «Free-running without pre-trigger» («Автономная выборка данных без предварительно заданного счетчика триггера»)

В режиме автономной выборки данных, если преобразователь находится в рабочем режиме, датчик непрерывно и автономно измеряет какой-либо физический параметр. Полученные и преобразованные данные аннулируются до получения запускающего сигнала триггера. Следующий отсчет, преобразованный сразу после получения запускающего сигнала, сохраняется в канале преобразователя или ИМП как первое слово в наборе данных. Последующие отсчеты сохраняются до тех пор, пока не закончится заполнение всего набора данных. После этого канал преобразователя снова начинает аннулировать отсчеты до получения следующего запускающего сигнала. Возможность осуществления выборок данных каналом преобразователя, когда преобразователь находится в режиме ожидания, оставлена на усмотрение изготовителя.

Исполнительное устройство, работающее в режиме автономной выборки данных, должно использовать полученный до прихода запускающего сигнала триггера набор данных в соответствии с его рабочим режимом «End-of-data-set» («Набор данных закончен»).

5.10.1.3 Режим «Free-running with pre-trigger» («Автономная выборка данных с предварительно заданным счетчиком триггера»)

В автономном режиме выборки данных, если преобразователь находится в рабочем режиме, датчик непрерывно и автономно измеряет какой-либо физический параметр. Полученные и преобразованные данные сохраняются до получения запускающего сигнала или достижения предварительного значения счетчика триггера (см. 7.1.2.2 и 8.5.2.32). После того как число сохраненных отсчетов данных достигает предварительно заданного значения счетчика триггера, получение следующего отсчета должно вызвать аннулирование старшего по времени отсчета, при этом новое значение должно быть сохранено. Следующий отсчет, преобразованный сразу после получения запускающего сигнала, сохраняется в ИМП как следующее слово в наборе данных. Последующие отсчеты сохраняются до тех пор, пока не закончится заполнение всего набора данных. Набор данных будет полным, когда после приема запускающего сигнала будет получено число выборок (отсчетов), равное размеру набора данных за вычетом предварительного значения счетчика триггера. Порядок действий после заполнения набора данных приведен в 5.10.1.4—5.10.1.5. Отклик на дополнительные запускающие сигналы, полученные до завершения отработки первого запускающего сигнала, описан в 5.11.5.

Исполнительное устройство может не работать в режиме «Free-running with pre-trigger» («Автономная выборка данных с предварительно заданным счетчиком триггера»).

5.10.1.4 Режим «Free-running with pre-trigger without buffers enabled» («Автономная выборка данных с предварительно заданным счетчиком триггера без доступа к буферу»)

Когда набор данных завершен, канал преобразователя начинает аннулировать отсчеты до поступления следующего запускающего сигнала или до завершения считывания набора данных. После считывания набора данных канал преобразователя должен возобновить сохранение отсчетов до получения нового запускающего сигнала. В этом режиме набор данных может быть считан только один раз. При последующих считываниях до получения следующего запускающего сигнала будет возвращено 0 байтов. Возможность осуществления выборок данных каналом преобразователя, когда преобразователь находится в режиме ожидания, оставлена на усмотрение изготовителя.

5.10.1.5 Режим «Free-running with pre-trigger and buffers enabled» («Автономная выборка данных с предварительно заданным счетчиком триггера с доступом к буферу»)

Когда набор данных завершен, канал преобразователя должен переключиться на следующий пустой буфер и начать получать выборки данных для следующего набора данных. Если свободных буферов не осталось, то все полученные выборки должны быть аннулированы до появления доступного буфера. Буфер будет считаться доступным после его считывания. Возможность осуществления выборок данных каналом преобразователя, когда преобразователь находится в режиме ожидания, оставлена на усмотрение изготовителя.

5.10.1.6 Режим «Continuous sampling» («Непрерывная выборка»)

В режиме «Continuous sampling» («Непрерывная выборка») датчик должен начать получать и сохранять отсчеты (выборки) в одном из буферов после получения запускающего сигнала. Работа в этом режиме аналогична работе в режиме «Free-running without pre-trigger» («Автономная выборка данных без предварительно заданного счетчика триггера»), приведенном в 5.10.1.2, за исключением того, что после заполнения набора данных канал преобразователя не прекращает работу, а переключается на следующий доступный буфер и продолжает получать данные. Для работы в данном режиме датчику требуется несколько доступных буферов для хранения выборок данных. При заполнении всех буферов данные в старшем буфере аннулируются вне зависимости от того, были ли они переданы СПП или нет, а буфер получает на хранение новые данные. Если поток данных поступает в режиме передачи с фиксированным интервалом, описанным в 5.10.2.3, то датчик должен переключиться на свободный буфер в начале нового интервала передачи вне зависимости от того, заполнен ли текущий буфер. Тем не менее если число выборок, полученных за один интервал передачи, больше числа «Max Data Repetitions» («Максимального повторения данных»), указанного в ЭТДП канала преобразователя (см. 8.5.2.28), то набор данных урезается до числа максимального повтора данных, и канал преобразователя устанавливает бит «Missed data or event» («Пропущенные данные или событие») (см. 5.13.5).

При получении начального запускающего сигнала датчик событий, работающий в режиме «Continuous sampling» («Непрерывная выборка»), должен обнаружить изменение состояния на его входе, сохранить это изменение состояния для передачи и продолжить поиск дополнительных изменений состояния на его входе. Для этого датчику событий требуется несколько буферов, которые он использует так же, как и любой другой датчик. Если поток данных поступает в режиме передачи с фиксированным интервалом, то канал преобразователя должен переключиться на свободный буфер в начале нового интервала передачи вне зависимости от того, заполнен ли текущий буфер.

В режиме «Continuous sampling» («Непрерывная выборка») после получения первого запускающего сигнала исполнительное устройство должно отработать все данные из его текущего буфера с частотой, контролируемой каналом преобразователя. После того как все данные из этого буфера отработаны, устройство переключается на буфер, заполненный ранее остальных, и продолжает отрабатывать данные. Если другой заполненный буфер недоступен, то исполнительное устройство должно предпринять действия, описанные в 5.10.4 и контролируемые установкой атрибута операции «End-of-data-set» («Набор данных закончен»), как указано в 8.5.2.48. Если предпринято действие «recirculate» («зациклить»), то устройство не будет искать новый заполненный буфер, а продолжит повторно отрабатывать данные текущего буфера. Если предпринято действие «hold»(«удержать»), то устройство должно переключиться на новый буфер, как только он будет получен и сохранен в памяти. Если будет предпринята попытка записать данные в такой канал преобразователя, и при этом не окажется доступных пустых буферов, то входящие данные должны быть проигнорированы, а канал преобразователя должен установить бит «Missed data or event» («Пропущенные данные или событие»).

5.10.1.7 Режим «Immediate operation» («Немедленное выполнение»)

Датчик, работающий в данном режиме выборки, немедленно получает наборы данных и передает их в виде ответа на команду «Read Transducer Channel data-set segment» («Считать сегмент набора данных канала преобразователя»). Получение команды «Read Transducer Channel data-set segment»

(«Считать сегмент набора данных канала преобразователя») будет работать как запускающий сигнал триггера.

Исполнительное устройство в этом режиме выборки немедленно отрабатывает набор данных, полученный после команды «Write Transducer Channel data-set segment» («Записать сегмент набора данных канала преобразователя»). Получение команды «Write Transducer Channel data-set segment» («Записать сегмент набора данных канала преобразователя») будет работать как запускающий сигнал триггера.

5.10.2 Режим «Data transmission» («Передача данных»)

Режимы «Data transmission» («Передача данных») определены в настоящем стандарте согласно таблице 7.

Таблица 7 — Режимы «Data transmission» («Передача данных»)

Нумерация	Наименование аргумента	Режим передачи данных	Описание
0	XmitMode.reserved[0]	Зарезервировано	
1	XmitMode.OnCommand	Только по команде	ИМП должен передавать набор данных только в ответ на команду «Read Transducer Channel data-set segment» («Считать сегмент набора данных канала преобразователя») (см. 7.1.3.1)
2	XmitMode.BufferFull	Потоковый при заполненном буфере	Передача данных осуществляется сразу после заполнения буфера, без ожидания от СПП команды «Read Transducer Channel data-set segment» («Считать сегмент набора данных канала преобразователя») (см. 7.1.3.1)
3	XmitMode.Interval	Потоковый с фиксированным интервалом	Данные из буфера передаются с фиксированным интервалом. Канал преобразователя должен прекратить использование текущего буфера вне зависимости от степени его заполнения данными, должен начать сохранять данные в другом буфере и передать данные из ранее используемого буфера без ожидания команды «Read Transducer Channel data-set segment» («Считать сегмент набора данных канала преобразователя») (см. 7.1.3.1)
4—127	XmitMode.reserved[N] 4≤N≤127	Зарезервировано	
128—255	XmitMode. open[N]128≤N≤255	Открыто для изготовителей	

5.10.2.1 Режим «XmitMode.OnCommand» («Передача данных только по команде»)

В данном режиме передачи данных канал преобразователя должен передавать набор данных только в ответ на команду «Read Transducer Channel data-set segment» («Считать сегмент набора данных канала преобразователя») (см. 7.1.3.1).

5.10.2.2 Режим «XmitMode.BufferFull» («Потоковый режим передачи данных при заполненном буфере»)

В данном режиме передачи данных датчик или датчик событий должен передать данные сразу после получения полного набора данных. Исполнительное устройство не работает с потоковыми данными и не может работать в этом режиме. Эквивалентным режимом для исполнительного устройства является режим «Continuous sampling» («Непрерывная выборка»), описанный в 5.10.1.6.

5.10.2.3 Режим «XmitMode.Interval» («Потоковый режим передачи данных с фиксированным интервалом»)

В данном режиме передачи данных канал преобразователя должен передавать полный или частичный набор данных с фиксированным интервалом. При работе в данном режиме повторы данных канала преобразователя не должны быть использованы для определения числа выборок в наборе данных. Число выборок в наборе данных определяется по частоте выборки и периоду передачи данных. Тем не менее если число выборок, собранное в течение одного интервала передачи, больше числа «Max Data Repetitions» («Максимального повторения данных»), указанного в ЭТДП канала преобразователя

(см. 8.5.2.28), то набор данных должен быть ограничен до числа «Max Data Repetitions» («Максимального повторения данных»), а также должен быть установлен бит «Hardware error» («Аппаратная ошибка») канала преобразователя (см. 5.13.7).

Метод определения интервала выборки определен в соответствующем стандарте для средств передачи и не является предметом рассмотрения настоящего стандарта.

5.10.3 Режим «Buffered operation» («Передача данных через буфер»)

Датчик или исполнительное устройство могут иметь возможность работы в режиме передачи данных с использованием и без использования буфера. В режиме «Передача данных через буфер» один из буферов доступен для считывания информации от датчика или для отработки данных на выходе исполнительного устройства. В свою очередь, другие буферы доступны для заполнения данными. Характеристикой канала преобразователя, работающего в режиме «Передача данных через буфер», является то, что данные, доступные для считывания или получения, — это всегда те данные, которые были доступны в буфере до получения предыдущего запускающего сигнала. В режиме передачи данных без использования буфера для хранения набора данных доступен только один буфер. Возможности буферизации заданного канала преобразователя описаны в атрибутах буфера в ЭТДП канала преобразователя. Допустимые значения данного атрибута представлены в 8.4.2.47.

При работе преобразователя в режиме ожидания и перед переходом в рабочий режим преобразователя датчик в первый раз после инициализации должен отправлять 0 байтов данных в ответ на любую команду «Считать сегмент набора данных канала преобразователя» (см. 7.1.3.1). Если при возвращении в режим ожидания преобразователя датчик имеет полностью или частично заполненные буферы, то в ответ на команду «Read Transducer Channel data-set segment» («Считать сегмент набора данных канала преобразователя») датчик должен вернуть содержимое буфера с информацией, записанной раньше. В ответ на последующие команды «Read Transducer Channel data-set segment» («Считать сегмент набора данных канала преобразователя») должно быть возвращено содержимое оставшихся буферов. После того как считано все содержимое каждого буфера, имевшего непрочитанные данные в момент получения каналом преобразователя команды перехода в режим ожидания (см. 7.1.4.2), канал преобразователя или прокси-канал преобразователя должен возвращать 0 байтов в ответ на команды «Read Transducer Channel data-set segment» («Считать сегмент набора данных канала преобразователя»).

Когда канал преобразователя переходит из режима ожидания в рабочий режим, все буферы должны быть очищены.

5.10.4 Режим «End-of-data-set» («Набор данных закончен»)

Данный режим используется только для исполнительных устройств. Атрибут операции «End-of-data-set» («Набор данных закончен») ЭТДП канала преобразователя (см. 8.5.2.48) определяет возможные операции, которые может производить исполнительное устройство после окончания набора данных. Два возможных режима работы приведены в 5.10.4.1—5.10.4.2.

Примечание — В данных рабочих режимах описываются способы, благодаря которым исполнительные устройства могут осуществлять плавные переходы от одного набора данных к другому. Несмотря на то что в большинстве случаев предпочтительны плавные переходы между наборами данных, иногда необходимо совершить быстрый переход, например, в случае аварийного выключения. В таких случаях для перехода в новое состояние может быть использован канал преобразователя второго исполнительного устройства того же ИМП. Механизм контроля переключения выходов одного исполнительного устройства на выходы другого содержится внутри ИМП и не является предметом рассмотрения настоящего стандарта.

5.10.4.1 Режим «Hold» («Удержать»)

Исполнительное устройство, работающее в режиме «Hold» («Удержать»), должно отработать все выборки из набора данных и затем должно продолжать отрабатывать последнее значение из набора данных до получения нового запускающего сигнала.

Примечание — Данный режим подходит для исполнительных устройств, содержащих клапан или иной механический механизм позиционирования. Данный режим также подходит для устройств с двумя устойчивыми состояниями выходного сигнала логического типа, но при этом часто связан с режимом выборки данных по срабатыванию триггера.

5.10.4.2 Режим «Recirculate» («Зациклить»)

Исполнительное устройство, работающее в режиме «Recirculate» («Зациклить»), должно отработать все выборки из набора данных в выходные сигналы, затем должно вернуться к началу набора данных и повторять отработку набора данных до тех пор, пока не будет получен другой запускающий сигнал или не будет отключен канал преобразователя. При переходе к началу повторного набора данных или

к новому набору данных должен поддерживаться соответствующий интервал между выборками. При получении другого запускающего сигнала триггера канал преобразователя должен переключиться на новый набор данных после окончания текущего набора данных.

5.10.5 Режим «Streaming operation» («Потоковый режим работы»)

Данный режим работы достигается сочетанием режима «Continuous sampling» («Непрерывная выборка») (см. 5.10.1.6) с режимом «XmitMode.BufferFull» («Потоковый режим передачи данных при заполненном буфере») (см. 5.10.2.2) либо режимом «XmitMode.Interval» («Потоковый режим передачи данных с фиксированным интервалом») (см. 5.10.2.3). Датчик или датчик событий, работающий в данном режиме, должен получать и передавать данные без дополнительных команд от СПП. Исполнительное устройство в потоковом режиме отрабатывает полученные данные без получения запускающей триггерной команды для каждого набора данных.

5.10.6 Рабочий режим «Edge-to-report» («Пороговое значение для посылки отчета»)

Данный рабочий режим существует только для датчиков событий.

Датчик событий может работать в одном из трех таких режимов. В режиме понижающих переходов датчик сообщает только о понижающих переходах. В режиме повышающих переходов датчик сообщает только о повышающих переходах. Определения повышающих и понижающих переходов см. в 5.6.2.3—5.6.2.4. В режиме любых переходов датчик сообщает как о повышающих, так и о понижающих переходах. Поле атрибута «Edge-to-report» («Пороговое значение для посылки отчета») ЭТДП канала преобразователя определяет способность датчика событий поддерживать данный рабочий режим. Если включены настройки по умолчанию, то существует шесть возможных значений данного атрибута. Более детальное описание см. в 8.5.2.50.

5.10.7 Режим «Actuator-halt» («Остановка исполнительного устройства»)

Данный атрибут применяется только к исполнительным устройствам. Данный режим определяет действия, которые совершает исполнительное устройство при получении команды о переходе канала преобразователя в режим ожидания. Возможные действия определяются атрибутом «Actuator-halt» («Остановка исполнительного устройства») в ЭТДП канала преобразователя (см. 8.5.2.51).

5.10.7.1 Режим «Halt immediate» («Немедленная остановка»)

Исполнительное устройство должно удерживать текущее состояние на его выходе до возвращения преобразователя в рабочий режим или до перехода в спящий режим.

5.10.7.2 Режим «Halt at the end of the data set» («Остановка по окончании набора данных»)

Исполнительное устройство должно завершить отработку текущего набора данных и сохранить последнее значение сигнала на его выходе до возвращения преобразователя в рабочий режим или до перехода в спящий режим.

5.10.7.3 Режим «Ramp to a predefined state» («Переход в заданное состояние»)

Исполнительное устройство должно использовать заданный изготовителем процесс для перевода своего выхода в заданное состояние.

5.11 Использование триггеров

Триггером является сигнал, применяемый к одному каналу преобразователя или набору каналов преобразователя и приводящий к совершению ими определенного действия. На диаграмме состояний на рисунке 7 изображены триггерные состояния датчика, а на рисунке 8 показаны триггерные состояния исполнительного устройства.

5.11.1 Использование триггеров для датчиков

Переход от одного состояния к другому зависит от полученных команд, определенных в разделе 7, состояния канала преобразователя (см. рисунок 3), состояния ИМП (см. рисунок 4), происходящих событий и состояния набора данных «полный/пустой».

Если ИМП выводится из активного состояния, то датчик должен немедленно вернуться в состояние «Sensor Trigger Initialize State» («Инициализация датчика по триггеру»). Если канал преобразователя выводится из рабочего состояния, то датчик должен немедленно вернуться в состояние «Sensor Trigger Initialize State» («Инициализация датчика по триггеру»). При получении команды перезагрузки датчик должен немедленно вернуться в состояние «Sensor Trigger Initialize State» («Инициализация датчика по триггеру»).

5.11.2 Использование триггеров для исполнительных устройств

Переход от одного состояния к другому при использовании триггеров для исполнительного механизма зависит от полученных команд, определенных в разделе 7, состояния канала преобразователя

(см. рисунок 3), состояния ИМП (см. рисунок 4), происходящих событий и состояния набора данных «полный/пустой».

Если ИМП выводится из активного состояния, а исполнительное устройство находится в состоянии «Transverse Data-set» («Поперечный набор данных»), то исполнительное устройство должно немедленно через состояние «Actuator-halt» («Остановка исполнительного устройства») перейти в состояние «Actuator Trigger Initialize» («Инициализация исполнительного устройства по триггеру»). Если преобразователь не находится в активном состоянии, а исполнительное устройство находится в режиме «Transverse Data-set» («Поперечный набор данных»), то исполнительное устройство должно немедленно через состояние «Actuator-halt» («Остановка исполнительного устройства») перейти в состояние «Actuator Trigger Initialize» («Инициализация исполнительного устройства по триггеру»). Если получена команда «Reset» («Перезагрузка»), а исполнительное устройство находится в режиме «Transverse Data-set» («Поперечный набор данных»), то исполнительное устройство должно немедленно через состояние «Actuator-halt» («Остановка исполнительного устройства») перейти в состояние «Actuator Trigger Initialize» («Инициализация исполнительного устройства по триггеру»). В состоянии «Actuator-halt» («Остановка исполнительного устройства») для обработки набора данных используется рабочий режим «Actuator-halt» («Остановка исполнительного устройства») (см. 5.10.7).

Если триггерный сигнал получен во время работы канала исполнительного устройства в состоянии «Transverse Data-set» («Поперечный набор данных») (см. рисунок 8), а режим работы «End-of-data-set» («Набор данных закончен») (см. 5.10.4) установлен в положение «Recirculate» («Зациклить»), и при этом устройство получает новый набор данных, то канал преобразователя должен переключиться на новый набор данных по окончании текущего набора данных.

Для корректного выхода из набора данных в состоянии «Actuator-haltt» («Остановка исполнительного устройства») должен быть использован аргумент команды рабочего режима «Actuator-halt» («Остановка исполнительного устройства»).

В настоящем стандарте приведено несколько методов запуска триггеров. Это заданные в явном виде триггерные команды от СПП, команды СПП для доступа к ИМП, события внутри ИМП, которые могут быть использованы как триггеры, и триггерные команды, которые могут быть инициализированы ИМП с подключенным СПП, если событие произошло внутри ИМП.

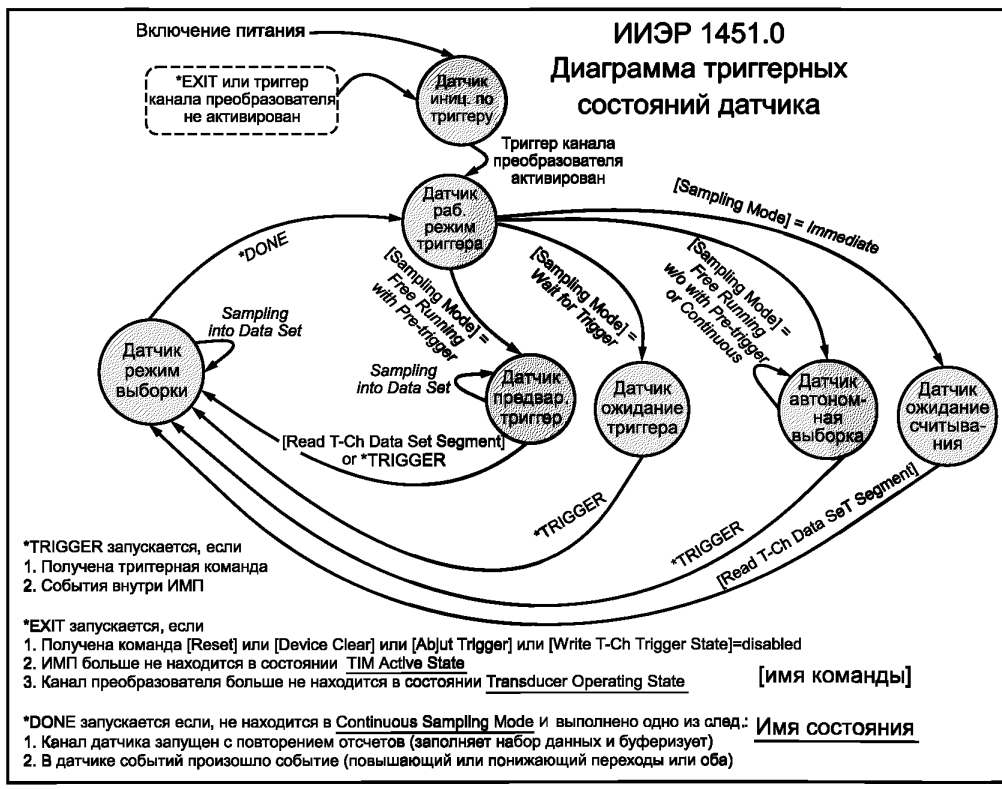


Рисунок 7 — Триггерные состояния датчика

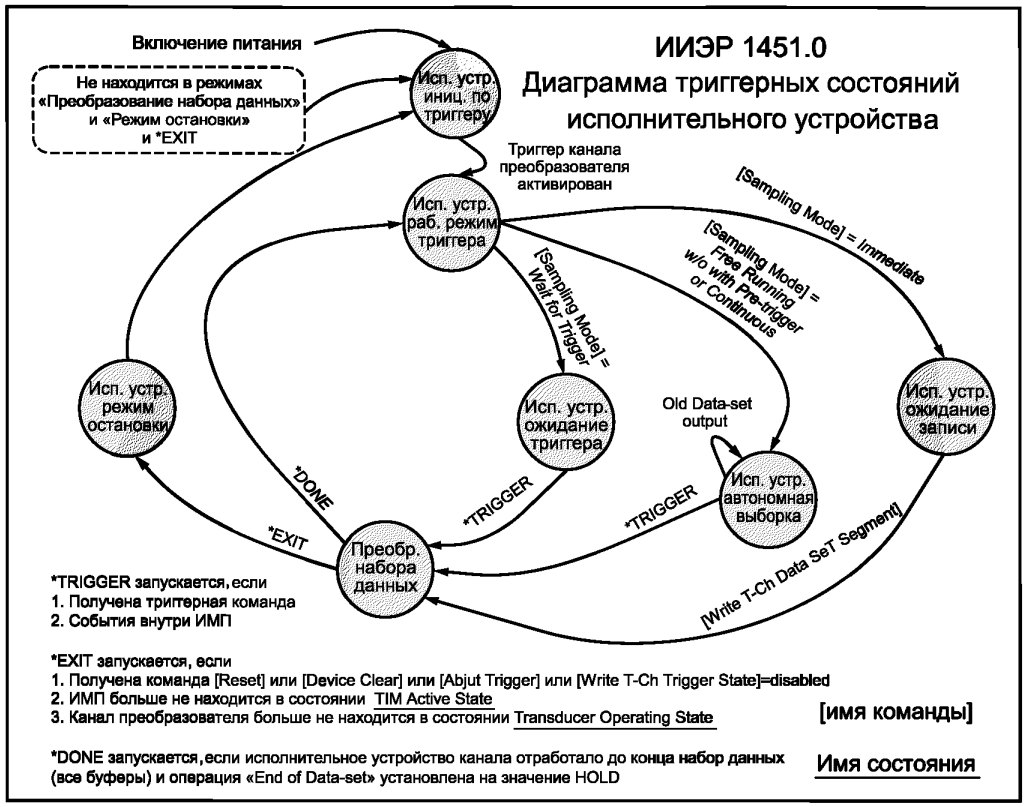


Рисунок 8 — Триггерные состояния исполнительного устройства

5.11.2.1 Триггерные команды

Триггерные команды посылаются от СПП одному или более каналам преобразователя в обычной среде передачи данных. Определение триггерной команды приведено в 7.1.3.3. Триггерная команда может быть адресована любому из следующих элементов:

- канал преобразователя;
- прокси-канал преобразователя;
- ИМП;
- адресная группа;
- глобальный адрес.

Триггерная команда, адресованная определенному каналу преобразователя, передается одному каналу преобразователя одного ИМП.

Прокси-канал преобразователя является ресурсом, имеющим адрес внутри одного ИМП (см. 5.9), и способен «представлять» один или более каналов преобразователя внутри данного ИМП. Триггерная команда, адресованная прокси-каналу преобразователя, исполняется для каждого канала преобразователя, входящего в прокси-канал преобразователя.

Триггерная команда, адресованная ИМП, запускает все доступные для сигналов триггера каналы преобразователя, входящие в такой ИМП.

Пользователь системы может определить адресную группу (см. 5.3). В настроенной системе каждый канал преобразователя, включенный в адресную группу, программируется реагировать на идентификатор данной адресной группы. Триггерная команда, отправленная такой адресной группе, запускает все элементы данной адресной группы. Если триггерная команда, составленная с применением технологии, приведенной в 10.2.3—10.2.4, отправлена нескольким ИМП, то она запускает все каналы преобразователя для всех ИМП, которым она была отправлена.

Глобальная триггерная команда запускает все доступные триггерным командам каналы преобразователя внутри ИМП. Для создания глобальной триггерной команды система использует отправку триггерной команды на глобальный адрес. Если триггерная команда, составленная с применением

технологии, приведенной в 10.2.3—10.2.4, отправлена нескольким ИМП, то она запускает все каналы преобразователя для всех ИМП, которым она была отправлена. Независимо от режима адресации канал преобразователя должен принимать триггерные команды только в том случае, если функция приема триггерных команд канала преобразователя активна, то есть канал преобразователя доступен для триггерных команд.

5.11.2.2 События, используемые как триггеры

События могут использоваться как триггеры для других каналов преобразователя непосредственно внутри одного ИМП. События такого характера не являются триггерными командами, но выполняют такие же функции. Событие, являющееся триггером, может формально рассматриваться как датчик событий. В результате события датчик событий может начать передачу триггерного сообщения другим ИМП в той же среде связи.

Примечание — В системах, где есть датчики событий, используемые в качестве триггеров, и где важно определить точное время события, необходимо дополнить датчик событий датчиком временных интервалов или датчиком временного события («TimeInstance»).

5.11.3 Номинальная триггерная логика

Функциональная блок-схема триггерной логики внутри ИМП приведена на рисунке 9.

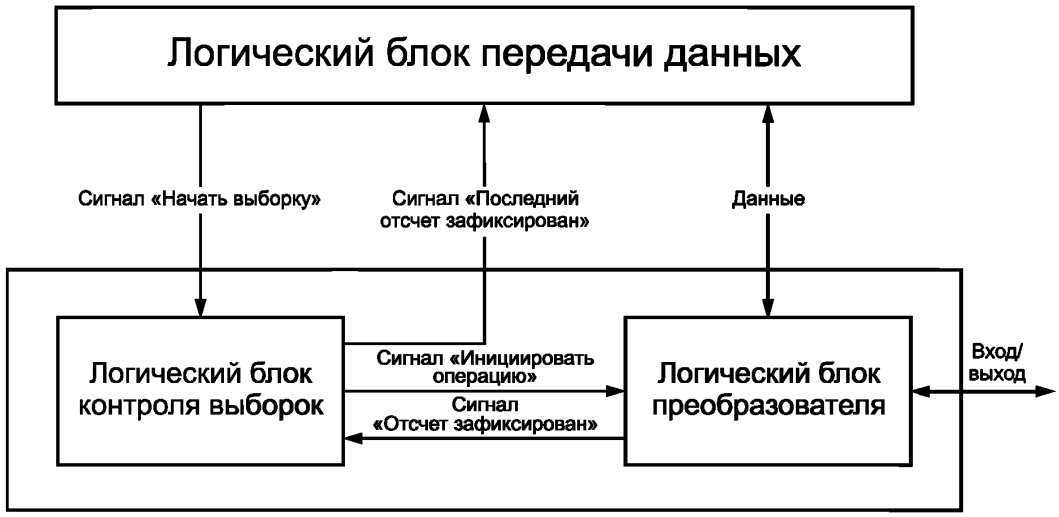


Рисунок 9 — Элементарные функциональные блоки канала преобразователя

Логический блок передачи данных отвечает на триггерные команды. После получения триггерной команды при условии, что функция получения триггерных команд каналом преобразователя активна, логический блок передачи данных передает сигнал «StartSamp» («Начать выборку»).

При получении сигнала «StartSamp» («Начать выборку») обычный датчик должен начать собирать новый набор данных. Для простейших датчиков это может означать обновление состояния одноступенчатого триггера или начало конвертации сигнала аналого-цифровым преобразователем. Более сложные датчики в ответ на триггерный сигнал могут выполнять многочисленные отсчеты, разнесенные в пространстве или во времени либо в каком-либо другом пространстве измеряемых величин.

При получении сигнала «StartSamp» («Начать выборку») датчик событий должен начать мониторинг определенного события. После того как событие произошло, логический блок контроля выборок должен создать сигнал фиксации последнего отсчета (выборки).

При получении сигнала «StartSamp» («Начать выборку») исполнительное устройство канала преобразователя должно отработать полученный набор данных, преобразовав его в выходные сигналы. В случае простейших исполнительных устройств это может означать обновление состояния одноступенчатого триггера или начало конвертации сигнала цифро-аналоговым преобразователем. Более сложные исполнительные устройства в ответ на триггерный сигнал могут отрабатывать многочисленные выборки данных, разнесенные в пространстве или во времени либо в каком-либо другом пространстве полученных величин.

Логический блок контроля выборок отвечает за организацию последовательности операций, необходимых для получения или отработки набора данных. Логический блок преобразователя обеспечивает формирование всех сигналов, преобразование данных, а также логику буферизации данных. Сигнал «Initiate Operation» («Инициализировать операцию») применяется для инициализации процесса преобразования данных на примере одной выборки, а сигнал «Sample Latched» («Выборка (отсчет) зафиксирован») показывает, что выборка была получена или преобразована в выходной сигнал.

Для каналов преобразователей, функционирующих с применением номинальной триггерной модели, определение времени фиксации выборки выполняется исключительно СПП или главным процессором и основано на информации, предоставленной ИМП. Для этого существуют два метода. При этом метод определения T_N, рассмотренный в 8.5.2.40, должен быть определен в ЭТДП канала преобразователя в графе «Source» («Источник») для поля «Time of sample» («Время выборки»).

В результате таких расчетов можно получить время последней выборки в наборе данных. Приведенные ниже вычисления показывают, как рассчитать время любой другой выборки из набора данных.

$$T_i = T_N - (N - i)t_{si} \quad (1)$$

для i = от 1 до N,

где N — число выборок в наборе данных,
T_j — время i-выборки,
T_N — время последней выборки в наборе данных,
t_{si} — интервал между выборками (период дискретизации).

Если нумерованное значение источника для времени выборки показывает, что время задержки между срабатыванием триггера и последующей фиксацией значения выборки постоянно, а в ИМП отсутствует встроенное аппаратное оборудование для обеспечения маркировки (сопровождения) времени, то время выборки должно рассчитываться от времени запуска триггерной команды путем прибавления поправки, равной времени задержки на распространение входящего сигнала, представленного в ЭТДП канала преобразователя в поле «Логика передачи данных». Данный метод пренебрегает величиной задержки распространения за счет среды транспортировки.

Расчетная формула имеет следующий вид:

$$T_1 = T_{tm} + t_{pdi}, \quad (2)$$

где T_{tm} — время дня, когда триггерная команда была отправлена от СПП,
t_{pdi} — время задержки на распространение входного сигнала представленного в ЭТДП канала преобразователя в поле «Логика передачи данных»,
T_1 — время дня, в которое была зафиксирована первая выборка из набора данных.

Такие временные расчеты позволяют получить время первой выборки в наборе данных. В следующей формуле показан способ расчета времени любой другой выборки в наборе данных:

$$T_i = T_m + i \cdot t_{si} \quad (3)$$

для i = от 1 до N,

где N — число выборок в наборе данных,
T_j — время i-выборки,
T_m — время первой выборки в наборе данных,
t_{si} — интервал между выборками.

Другие возможные конфигурации триггерной логики приведены в приложении L.

5.11.4 Триггерная логика, основанная на определении события

Функциональная блок-схема, приведенная на рисунке 9, изображает канал преобразователя с датчиком событий в качестве триггера. Датчик событий может содержать дополнительные встроенные исполнительные устройства, позволяющие обеспечить выполнение таких функций, как программируемый аналоговый порог срабатывания и гистерезис, или программируемая цифровая последовательность (цифровая комбинация).

Датчик событий связан на аппаратном уровне с одним или более каналами преобразователя, определенными в мета-ЭТДП как «ChannelGroup» («Группа каналов»). При получении триггерной команды датчик событий начинает мониторинг. Произошедшее событие также может служить триггером для соответствующего(их) канала(ов) преобразователя.

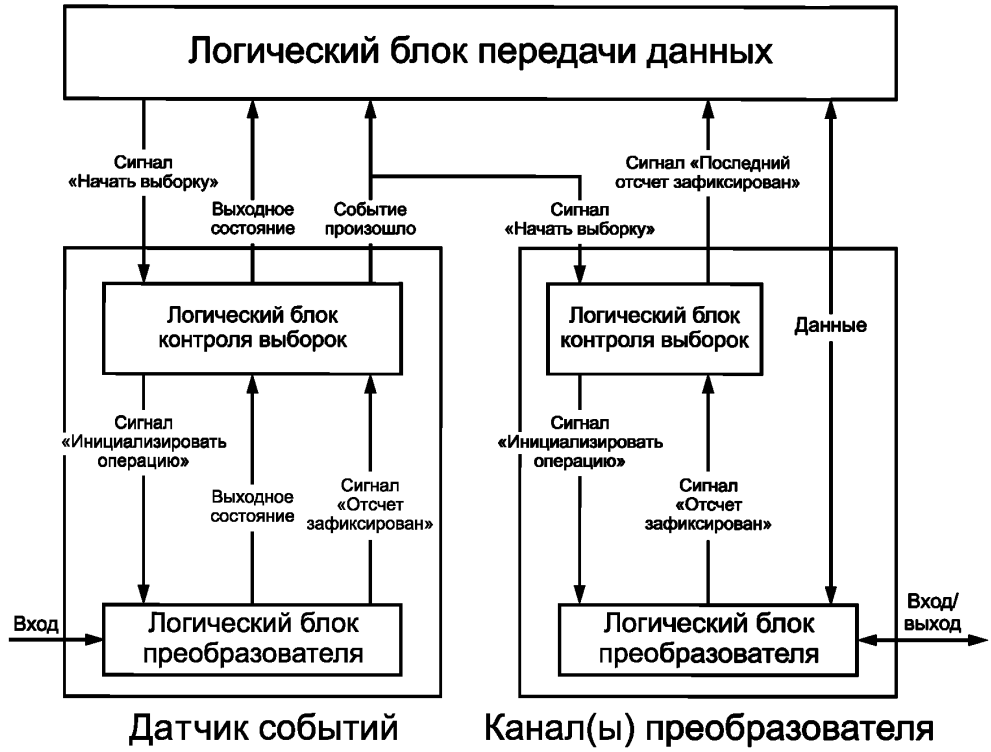


Рисунок 10 — Выход датчика событий, использующийся в качестве триггера

Канал преобразователя может не принимать триггерные команды, адресованные каналам преобразователя, сконфигурированным таким образом.

5.11.5 Избыточный запуск триггерных команд канала преобразователя

Независимо от времени, необходимого для считывания данных канала преобразователя, для успешного получения триггерных команд СПП должен отправлять их с интервалом времени не меньшим, чем интервал между выборками канала преобразователя. Интервал между выборками канала преобразователя приведен в ЭТДП канала преобразователя (см. 8.5.2.36).

Термин «избыточный запуск триггерных команд» относится к последующему триггеру, выпущенному до окончания времени интервала между выборками канала преобразователя. В этом случае канал преобразователя может не успеть полностью обработать предыдущий триггер. В таблице 8 приведен список ответов на избыточный запуск триггерных команд канала преобразователя, находящегося в рабочем режиме, описанном в 5.10.

Таблица 8 — Ответ на избыточные триггерные команды

Рабочий режим		Ответ на триггер
Потоковый	Буферный	
Да	Не важно	Игнорировать все, кроме первого триггера
Нет	Да	Воспринимать как предварительный триггер
Нет	Нет	Игнорировать триггеры до завершения предыдущей операции

В непотоковом буферном режиме устройство воспринимает триггер, полученный до фиксации всех выборок в наборе данных, как предварительный триггер. Иначе говоря, устройство принимает триггер в качестве команды для запуска другого буфера после завершения обработки данных текущего

буфера таким образом, что все получаемые или отрабатываемые выборки проходят с сохранением одинакового интервала. Если во время обработки буфера получено более одного избыточного триггера, то такие триггеры игнорируются. В этом случае время выборки не может быть рассчитано из времени отправки триггера. В любом случае, если триггерная команда проигнорирована, в соответствующем регистре состояния устанавливается бит «Command Rejected» («Отказ от выполнения команды») (см. 5.13.4).

5.12 Синхронизация

Основным требованием для крупномасштабного сбора данных с распределенных многоточечных массивов датчиков является обеспечение в системе функции запуска синхронизированных во времени триггерных сигналов для различных преобразователей данных. Так как для точного временного «снимка состояния» измеряемых параметров принципиально важно обеспечить одновременный сбор широкого спектра точек данных, то отдельные выборки от распределенных в пространстве датчиков нужно координировать в довольно точных временных интервалах. Для выполнения этой задачи в настоящем стандарте описано несколько функций, приведенных в 5.12.1—5.12.2.

5.12.1 Методы получения синхронизации

Основным инструментом для обеспечения синхронизации является триггер. Так как триггеры могут быть адресованы группам каналов преобразователя, то выборка данных для членов группы проходит синхронно в пределах времени задержки сигнала и с разницей на время задержки между получением триггерного сигнала ИМП и его загрузкой в канал преобразователя.

5.12.1.1 Применение триггеров

Самым простым путем достижения синхронизации выборок данных являются адресные группы и глобальные триггеры, описанные в 5.3. Если синхронизация выборок данных должна быть достигнута для нескольких ИМП, то требуется использовать метод адресации для нескольких ИМП, приведенный в 10.2.3—10.2.4. При использовании данного метода максимально достижимая временная точность зависит от базовых средств и протокола связи. Данный метод хорошо работает для каналов преобразователей, загружающих или обрабатывающих одиночную выборку после получения одного триггерного сигнала. Метод является менее точным при работе с наборами данных, содержащими более одной выборки. Метод также имеет ограничения при очень большой длине кабелей или при очень строгих требованиях к синхронизации.

5.12.1.2 Применение отсроченного триггера

Задержки в системе, возникающие по причине различной длины кабелей между СПП и ИМП, а также по причине задержек внутри ИМП, могут быть устранены при использовании функции отсроченного триггера. Более детально данный процесс описан в приложении L. Данная функция ИМП может быть использована, если СПП обладает методом определения задержек для различных ИМП. Данный метод может быть применен в том случае, если в стандарте для физической среды передачи данных оговорен метод определения задержек передачи информации.

5.12.1.3 Применение сигнала синхронизации

Сигнал синхронизации обеспечивает синхронный тактовый импульс с одинаковой частотой для всех ИМП в обычной среде. Применение сигнала синхронизации позволяет улучшить работу с адресными группами или глобальными триггерами каналов преобразователя, работающих с наборами данных, содержащих более одной выборки. Тем не менее применение сигнала синхронизации не устраняет напрямую различные задержки, связанные с длиной кабеля.

5.12.2 Сигнал синхронизации

В некоторых из стандартов настоящего комплекса описаны СПП, способные генерировать тактовые импульсы синхронизации и передавать их ИМП. Модули ИМП могут получать данные тактовые импульсы, загружать их в буфер и использовать для генерации соответствующих тактовых импульсов внутри самого ИМП. Детальная информация о передаче тактовых импульсов от СПП к ИМП приведена в соответствующем стандарте комплекса ИИЭР 1451.

5.13 Состояние (статус)

Как показано на рисунке 11, существуют два вида регистра состояния, и оба вида имеют длину в 32 бита. Первый регистр носит название «Condition Register» («Регистр условия»). Его можно считать при помощи команды «Read Status-Condition Register» («Считать статус состояния регистра условия»)

(см. 7.1.1.9). Данный регистр содержит информацию о текущем состоянии передаваемых атрибутов. Второй регистр называется «Event Register» («Регистр события»). Данный регистр содержит значение «True» при условии, если «Condition Register» («Регистр условия») изменил значение на значение «True» с момента последней очистки регистра события. Регистр можно считать при помощи команды «Read Status-Event Register» («Считать статус состояния регистра события») (см. 7.1.1.8). Некоторые биты регистра события представляют собой фактические события, такие как ошибки выполнения команд. В регистре условия такие биты всегда должны быть равны нулю, поскольку иначе в модели будет отображено изменение условия с его немедленной очисткой.

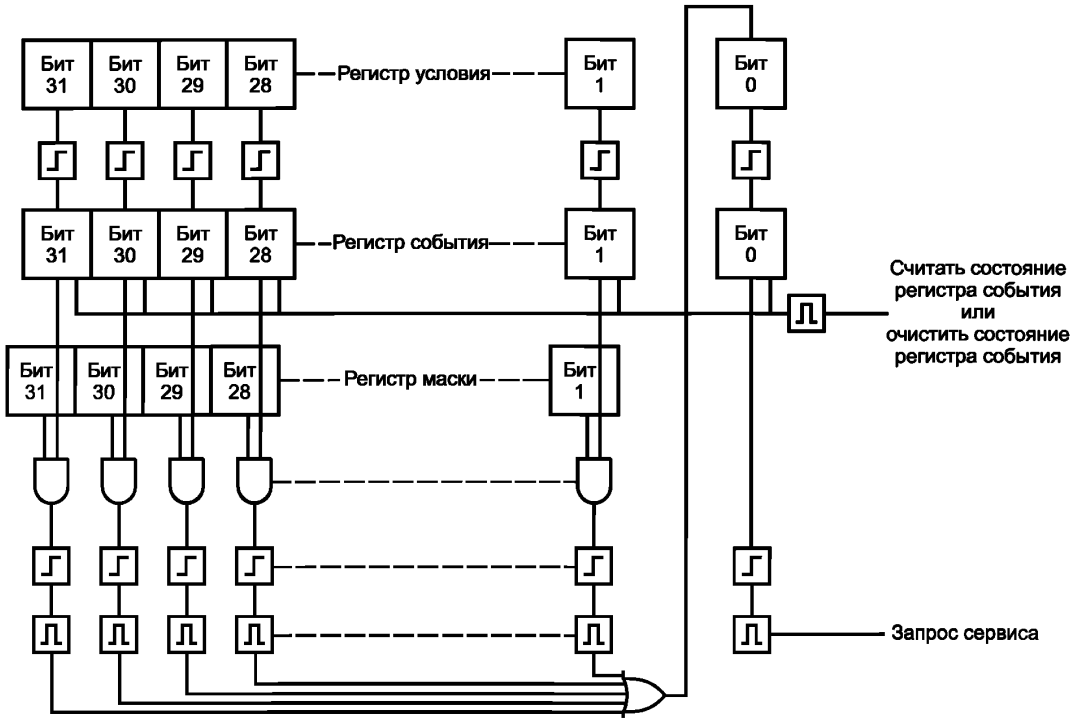


Рисунок 11 — Логика создания сообщений состояния

Оба регистра состояний должны быть применимы к ИМП и к каждому каналу преобразователя внутри ИМП. Возвращенное состояние-событие представляет собой состояние ИМП в целом. Во многих случаях бит состояния-события ИМП представляет собой «OR» («логическое ИЛИ») соответствующих битов для всех каналов преобразователя.

Каждый бит регистра состояния-события означает наличие или отсутствие определенного события. Значение «1» в соответствующей позиции бита должно означать, что условие выполняется или что событие произошло с момента последнего считывания или очистки статуса.

Каждый регистр состояния-события используется в сочетании с соответствующей маской сервисного запроса (см. 5.14.1), позволяющей проконтролировать выбор битов для формирования сервисного запроса.

Биты состояния, определенные в настоящем стандарте, приведены в таблице 9.

Некоторые биты состояния зарезервированы для будущих версий настоящего стандарта. Некоторые биты являются необязательными и могут быть не предусмотрены изготовителем. Биты, обозначенные как «открытые для изготовителей», могут использоваться для сообщения о состояниях, не представленных уже определенными битами. Описания новых битов регистра состояния-события ИМП должны отражать состояния внутри ИМП в целом. Биты состояния, отмеченные как «зарезервированные», или не использующиеся «необязательные», или «открытые для изготовителей», при считывании должны быть равны нулю.

Таблица 9 — Биты состояния

Бит	Биты состояния ИМП	Биты состояния канала преобразователя	Обязательный/ необязательный
0	Service request («Запрос сервиса»)	Service request («Запрос сервиса»)	Обязательный
1	TEDS changed («ЭТДП изменена»)	TEDS changed («ЭТДП изменена»)	Необязательный
2	Invalid command («Неверная команда»)	Зарезервировано	Обязательный
3	Command rejected («Отказ от выполнения команды»)	Command rejected («Отказ от выполнения команды»)	Обязательный
4	Missed data or event («Пропущенные данные или событие»)	Missed data or event («Пропущенные данные или событие»)	См. 5.13.5[1)]
5	Data/event («Данные/событие»)	Data/event («Данные/событие»)	См. 5.13.6[2)]
6	Hardware error («Ошибка аппаратного оборудования»)	Hardware error («Ошибка аппаратного оборудования»)	См. 5.13.7[3)]
7	Not operational («Нерабочее состояние»)	Not operational («Нерабочее состояние»)	Обязательный
8	Protocol error («Ошибка протокола»)	Зарезервировано	Обязательный
9	Data available/data processed («Данные доступны/данные обработаны»)	Data available/data processed («Данные доступны/данные обработаны»)	Необязательный
10	Busy («Занято»)	Busy («Занято»)	Необязательный
11	Failed calibration («Сбой калибровки»)	Failed calibration («Сбой калибровки»)	Необязательный
12	Failed self-test («Сбой самоконтроля»)	Failed self-test («Сбой самоконтроля»)	Необязательный
13	Data over range or under range («Данные выходят за верхнюю или нижнюю границы диапазона»)	Data over range or under range («Данные выходят за верхнюю или нижнюю границы диапазона»)	Необязательный
14	Corrections disabled («Коррекции отключены»)	Corrections disabled («Коррекции отключены»)	Необязательный
15	Consumables exhausted («Закончились расходные материалы»)	Consumables exhausted («Закончились расходные материалы»)	Необязательный
16	Зарезервировано	Not-the-first-read-of-this-data-set («Не первое считывание из этого набора данных»)	Обязательный
17—23	Зарезервировано	Зарезервировано	—
23—31	Открыто для изготовителей	Открыто для изготовителей	—

Очистка битов в регистре состояния-события может быть проведена одним из следующих способов:

- очистка битов «Summary» («Сводка») должна быть проведена немедленно после считывания соответствующего нижележащего (базового) регистра состояния-события. Любой бит в регистре состояния-события ИМП, определенный как «OR» («логическое ИЛИ») для соответствующего бита регистров состояния-события канала преобразователя, должен быть очищен таким же образом;
- очистка прочих битов должна происходить сразу после того, как изменилось условие, о котором они сообщали, и произошло считывание регистра состояния-события или была получена команда очистки состояния. Данные биты не должны очищаться при получении протокола очистки устройства;
- биты состояния также могут меняться при изменении рабочих режимов, приведенных в 5.4.1.

[1)] В оригинале ISO/IEC/IEEE 21450:2010 допущена ошибка. Ошибочно приведена ссылка на 5.13.4.

[2)] В оригинале ISO/IEC/IEEE 21450:2010 допущена ошибка. Ошибочно приведена ссылка на 5.13.5.

[3)] В оригинале ISO/IEC/IEEE 21450:2010 допущена ошибка. Ошибочно приведена ссылка на 5.13.6.

При инициализации во время включения питания или перезагрузки время задержки передачи данных может превысить значения времени задержки, оговоренные в ЭТДП. ИМП должен провести проверку считываний регистра состояния и убедиться, что они возвращают точные данные о состоянии ИМП. В противном случае ИМП должен отложить любое считывание регистра состояния.

Считывание регистров состояния должно быть доступно при помощи команд «Read status-event register» («Считать статус состояния регистра события») и «Read status-condition register» («Считать статус состояния регистра условия»), определенных в 7.1.1.8—7.1.1.9. Полученный в ответ статус должен иметь размер 4 байта. Информация, полученная в ответ на обращение к ИМП, определена в таблице 9, в графе «Биты состояния ИМП». Биты, полученные в ответ на обращение к каналу преобразователя, определены в графе «Биты состояния канала преобразователя».

Регистр состояния-события также может быть передан в запущенном ИМП сообщении (см. 6.4) при условии, что маска установлена для бита регистра состояния-события, который был установлен, и протокол состояния-события был активирован при помощи команды «Write status-event protocol state» («Записать состояние протокола состояния-события») (см. 7.1.1.11). После того как данный протокол активирован, должен быть инициирован поток, отправляющий 32-битовый регистр состояния каждый раз после установки бита «Service request» («Запрос сервиса»). При запросе сервиса каналом должен быть отправлен регистр канала. Если запрос совершает ИМП, то должен быть отправлен регистр состояния ИМП. Информация, отправленная через протокол состояния-события, должна быть идентична информации, полученной в ответ на команду «Read status-event register» («Считать статус состояния регистра события»).

5.13.1 Бит «Service request» («Запрос сервиса»)

Бит «Service request» («Запрос сервиса») для любого канала преобразователя должен быть установлен при запросе сервиса этим каналом преобразователя. Указанный бит должен быть очищен после считывания, после отправки сообщения протокола состояния или после отправки команды «Clear status-event register» («Очистить регистр состояния-события») (см. 7.1.1.10) для данного канала. Для определения условий, при которых канал преобразователя запрашивает сервис, применяются маски сервисного запроса.

Описание маски сервисного запроса приведено в 5.14.1.

Бит «Service request» («Запрос сервиса») для ИМП должен быть установлен в случае любого запроса сервиса ИМП, как определено масками ИМП. Он должен быть очищен после считывания, после отправки сообщения протокола состояния или после отправки команды «Clear Transducer status-event register» («Очистить регистр события-состояния преобразователя»). Бит «Service request» («Запрос сервиса») для ИМП не должен очищаться при получении протокола очистки устройства.

Бит «Service request» («Запрос сервиса») для канала преобразователя должен быть применим для каждого канала преобразователя внутри ИМП. Бит «Service request» («Запрос сервиса») для ИМП должен быть применим для каждого ИМП.

Бит «Service request» («Запрос сервиса») очищается при изменении рабочего режима при условии, что бит состояния, вызвавший это изменение, изменяется или очищается.

Бит «Service request» («Запрос сервиса») должен быть определен при включении питания, а также он должен быть установлен при пометке бита активизации какого-либо состояния, в особенности если активизирован бит состояния «Power on» («Включение питания»).

5.13.2 Бит «TEDS changed» («ЭТДП изменена»)

Бит «TEDS changed» («ЭТДП изменена») для модуля ИМП должен быть установлен при любых производимых ИМП изменениях содержания адаптивных ЭТДП. Бит «TEDS changed» («ЭТДП изменена») для канала преобразователя должен быть установлен при любых изменениях содержания адаптивных ЭТДП, производимых каналом преобразователя. Бит должен быть очищен после считывания.

Данный бит не зависит от изменения рабочего режима преобразователя.

5.13.3 Бит «Invalid command» («Неверная команда»)

Бит «Invalid command» («Неверная команда») для ИМП должен быть установлен в случае обнаружения модулем ИМП любой невыполнимой команды или при считывании или записи неприменимой функции. Бит должен быть очищен после считывания.

Данный бит не зависит от изменения рабочего режима преобразователя.

5.13.4 Бит «Command rejected» («Отказ от выполнения команды»)

Бит «Command rejected» («Отказ от выполнения команды») должен быть установлен в случае обнаружения ИМП действительной команды, которая не может быть выполнена в текущем режиме работы ИМП или канала преобразователя. Также данный бит должен быть установлен, если по какой-либо причине аргумент команды является неприемлемым. Бит должен быть очищен после считывания.

Бит «Command rejected» («Отказ от выполнения команды») является обязательным и должен быть применим к ИМП и всем каналам преобразователя.

Данный бит не зависит от изменения рабочего режима.

5.13.5 Бит «Missed data or event» («Пропущенные данные или событие»)

Бит «Missed data or event» («Пропущенные данные или событие») должен быть установлен одновременно с временем выборки данных (временем снятия отсчета) датчика события или датчика, находящегося в режиме автономной выборки, если канал преобразователя находится в рабочем режиме преобразователя, но еще не получил запускающей триггерной команды. Исключение составляет случай, когда после того как бит был очищен, он не должен быть установлен до получения первого триггерного сигнала для любого канала преобразователя. Данный бит должен быть очищен после считывания.

Бит «Missed data or event» («Пропущенные данные или событие») для ИМП должен быть установлен, если установлен какой-либо бит «Missed data or event» («Пропущенные данные или событие») канала преобразователя.

Данный бит состояния должен быть применим для любого датчика событий или датчика, способного работать в режиме автономной выборки. Бит также должен быть применим для любого ИМП, содержащего подобный датчик или датчик событий.

Бит «Missed data or event» («Пропущенные данные или событие») должен быть очищен при переходе канала преобразователя из режима остановки в рабочий режим преобразователя. Описание рабочих режимов приведено в 5.4.1.

Примечание — Если канал преобразователя в режиме ожидания преобразователя не собирает или не потребляет данные, то бит должен оставаться установленным до тех пор, пока не будет считан. С другой стороны, если бит был установлен, то он должен быть очищен при переходе канала преобразователя из рабочего режима в режим ожидания.

5.13.6 Бит «Data/event» («Данные/событие»)

Бит «Data/event» («Данные/событие») для канала преобразователя должен быть установлен во время получения выборки данных (отсчета) датчика в рабочем режиме преобразователя, работающего в автономном режиме выборки данных или при обнаружении события датчиком события. Бит «Data/event» («Данные/событие») должен быть очищен после считывания. Бит «Data/event» («Данные/событие») для ИМП должен быть установлен, если установлены какие-либо из битов «Data/event» («Данные/событие») для канала преобразователя.

Данный бит состояния должен быть применим для любых датчиков событий или датчиков, способных работать в режиме автономной выборки. Бит также должен быть применим для любого ИМП, содержащего подобный датчик или датчик событий.

Бит «Data/event» («Данные/событие») для канала преобразователя должен быть очищен при переходе канала преобразователя из режима ожидания в рабочий режим преобразователя. Описание рабочих режимов представлено в 5.4.1.

Примечание — Если канал преобразователя в режиме ожидания преобразователя не собирает или не потребляет данные, то бит должен оставаться установленным до тех пор, пока не будет считан. С другой стороны, если бит был установлен, то он должен быть очищен при переходе канала преобразователя из рабочего режима в режим ожидания.

5.13.7 Бит «Hardware error» («Аппаратная ошибка»)

Бит «Hardware error» («Аппаратная ошибка») для канала преобразователя должен быть установлен, если выполняются условия, о которых он сообщает. Бит «Hardware error» («Аппаратная ошибка») должен быть очищен после считывания, если условия, о которых он сообщает, больше не выполняются. Датчик событий должен установить данный бит после неудавшихся проверок на совместимость. Бит «Hardware error» («Аппаратная ошибка») для ИМП должен быть установлен, если установлены какие-либо биты «Hardware error» («Аппаратная ошибка») для канала преобразователя. Изготовитель ИМП может определить дополнительные критерии аппаратной ошибки.

Бит «Hardware error» («Аппаратная ошибка») для ИМП при возвращении статуса ИМП и бит «Hardware error» («Аппаратная ошибка») для канала преобразователя должны быть применимы для любого ИМП, содержащего датчик событий и проводящего проверки на совместимость (см. 5.6.1).

Данный бит не зависит от изменения рабочего режима.

5.13.8 Бит «Not operational» («Нерабочее состояние»)

Бит «Not operational» («Нерабочее состояние») для канала преобразователя должен быть установлен, если канал преобразователя не соответствует спецификациям изготовителя. Бит «Not operational»

(«Нерабочее состояние») должен быть установлен при прогреве или любом другом состоянии, когда канал преобразователя не соответствует спецификациям или не функционирует.

Бит «Not operational» («Нерабочее состояние») для ИМП должен представлять результат операции «OR» («логическое ИЛИ») для всех битов нерабочего состояния канала преобразователя. Бит «Not operational» («Нерабочее состояние») также должен быть установлен, если ИМП находится в нерабочем состоянии по любой другой причине. Бит «Not operational» («Нерабочее состояние») для ИМП должен быть очищен после того, как причина, приведшая к его установке, устранена.

Данный бит не зависит от изменения рабочего режима.

5.13.9 Бит «Protocol error» («Ошибка протокола»)

Бит «Protocol error» («Ошибка протокола») устанавливается, если в сообщении, полученном через интерфейсную среду преобразователя, содержатся ошибки протокола.

Данный бит не зависит от изменения рабочего режима.

5.13.10 Бит «Data available/data processed» («Данные доступны/данные обработаны»)

Бит «Data available/data processed» («Данные доступны/данные обработаны») для канала преобразователя должен быть установлен, если у данного канала преобразователя есть данные, доступные для считывания датчиком, или если канал преобразователя завершил обработку данных в исполнительном устройстве. Бит «Data available/data processed» («Данные доступны/данные обработаны») для ИМП должен быть установлен, если у какого-либо канала преобразователя для данного ИМП есть данные, доступные для считывания датчиком, или если канал преобразователя завершил обработку данных в исполнительном устройстве. Бит «Data available/data processed» («Данные доступны/данные обработаны») должен оставаться установленным до тех пор, пока условия сохраняются. В случае датчика это означает, что данный бит должен оставаться установленным, пока данные не будут считаны. В случае исполнительного устройства это означает, что данный бит должен оставаться установленным, пока на устройство не будут записаны новые данные или не будет считано состояние.

Бит «Data available/data processed» («Данные доступны/данные обработаны») должен быть очищен при переходе канала преобразователя из режима остановки в рабочий режим преобразователя. Описание рабочих режимов приведено в 5.4.1.

Примечание — Если канал преобразователя в режиме ожидания преобразователя не собирает или не потребляет данные, то бит должен оставаться установленным до тех пор, пока не будет считан. С другой стороны, если бит был установлен, то он должен быть очищен при переходе канала преобразователя из рабочего режима в режим ожидания.

5.13.11 Бит «Busy» («Занято»)

Бит «Busy» («Занято») для канала преобразователя должен быть установлен, если канал преобразователя не может поддерживать доступ к записи или считыванию данных канала преобразователя через транспортировку данных. Существует ряд команд, в результате которых канал преобразования может оказаться занят: например, команда «Run self-test» («Запустить самодиагностику») (см. 7.1.1.5), команда «Calibrate Transducer Channel» («Калибровать канал преобразователя») (см. 7.1.2.10), команда «Reset» («Перезагрузка») (см. 7.1.7.1), команда «Zero Transducer Channel» («Обнулить канал преобразователя») (см. 7.1.2.11) или любая другая команда, изменяющая нормальный режим работы. Данный бит должен оставаться установленным, только пока сохраняются условия, указывающие на то, что канал занят. Бит «Busy» («Занято») для ИМП должен представлять результат операции «OR» («логическое ИЛИ») всех битов «Busy» («Занято») каналов преобразователя внутри ИМП.

Данный бит не зависит от изменения рабочего режима.

5.13.12 Бит «Failed calibration» («Сбой калибровки»)

Бит «Failed calibration» («Сбой калибровки») для канала преобразователя должен быть установлен, если во время «калибровочной» проверки (см. 7.1.2.10) не были получены ожидаемые результаты. Бит «Failed calibration» («Сбой калибровки») должен быть очищен после считывания. Бит «Failed calibration» («Сбой калибровки») для ИМП должен представлять результат операции «OR» («логическое ИЛИ») всех битов «Failed calibration» («Сбой калибровки») каналов преобразователя внутри ИМП.

Данный бит не зависит от изменения рабочего режима.

5.13.13 Бит «Failed self-test» («Сбой самодиагностики»)

Бит «Failed self-test» («Сбой самодиагностики») для канала преобразователя и бит «Failed self-test» («Сбой самодиагностики») для ИМП должны быть установлены, если в результате запуска команд «Run self-test» («Запустить самодиагностику») не были получены ожидаемые результаты (см. 7.1.1.5). Бит «Failed self-test» («Сбой самодиагностики») должен быть очищен после считывания. Бит «Failed

self-test» («Сбой самодиагностики») для ИМП должен представлять результат операции «OR» («логическое ИЛИ») всех битов «Failed self-test» («Сбой самодиагностики») каналов преобразователя внутри ИМП.

Данный бит не зависит от изменения рабочего режима.

5.13.14 Бит «Data over-range or under-range» («Данные выходят за верхнюю или нижнюю границы диапазона»)

Бит «Data over-range or under-range» («Данные выходят за верхнюю или нижнюю границы диапазона») для канала преобразователя должен быть установлен, когда канал преобразователя обнаруживает условия выхода сигналов за верхнюю или нижнюю границы диапазона. Бит «Data over-range or under-range» («Данные выходят за верхнюю или нижнюю границы диапазона») для ИМП должен представлять результат операции «OR» («логическое ИЛИ») всех битов «Data over-range or under-range» («Данные выходят за верхнюю или нижнюю границы диапазона») каналов преобразователя внутри ИМП. Данный бит должен быть очищен после считывания.

Условие выхода за верхнюю границу диапазона выполняется, если величина сигнала на выходе канала преобразователя больше (или имеет большее по модулю положительное значение) величины, указанной в поле верхней границы рабочего диапазона устройства в ЭТДП канала преобразователя (см. 8.5.2.19). Условие выхода за нижнюю границу диапазона выполняется, если величина сигнала на выходе канала преобразователя меньше (или имеет большее по модулю отрицательное значение) величины, указанной в поле нижней границы рабочего диапазона устройства в ЭТДП канала преобразователя (см. 8.5.2.7).

Бит «Data over-range or under-range» («Данные выходят за верхнюю или нижнюю границы диапазона») канала преобразователя должен быть очищен при переходе канала преобразователя из режима остановки в рабочий режим. Описание рабочих режимов приведено в 5.4.1.

Примечание — Если канал преобразователя в режиме ожидания преобразователя не собирает или не потребляет данные, то бит должен оставаться установленным до тех пор, пока не будет считан. С другой стороны, если бит был установлен, то он должен быть очищен при переходе канала преобразователя из рабочего режима в режим ожидания.

5.13.15 Бит «Corrections disabled» («Коррекции отключены»)

Бит «Corrections disabled» («Коррекции отключены») должен быть установлен для любого ИМП или канала преобразователя, для которых корректировка данных возможна, но отключена. Бит «Corrections disabled» («Коррекции отключены») должен быть применим для любого канала преобразователя или ИМП, содержащего возможность корректировки данных. Данный бит должен быть равен нулю для любого канала преобразователя или ИМП, не имеющего возможности корректировки данных.

Данный бит не зависит от изменения рабочего режима.

5.13.16 Бит «Consumables exhausted» («Закончились расходные материалы»)

Бит «Consumables exhausted» («Закончились расходные материалы») для канала преобразователя должен быть установлен, если канал преобразователя не может больше получать какие-либо расходные материалы, необходимые для продолжения работы. Бит «Consumables exhausted» («Закончились расходные материалы») для ИМП должен представлять результат операции «OR» («логическое ИЛИ») всех битов «Consumables exhausted» («Закончились расходные материалы») каналов преобразователя внутри ИМП. Данный бит должен быть установлен, только пока выполняются описанные в данном пункте условия.

Данный бит не зависит от изменения рабочего режима.

5.13.17 Бит «Not-the-first-read-of-this-data-set» («Не первое считывание из этого набора данных»)

Бит «Not-the-first-read-of-this-data-set» («Не первое считывание из этого набора данных») для любого канала преобразователя должен быть установлен при считывании набора данных более одного раза. Данный бит должен быть очищен после считывания.

Данный бит не зависит от изменения рабочего режима.

5.14 Логика сервисного запроса

В случае если внутри ИМП существует какое-либо требующее внимания условие, то от ИМП к СПП отправляются сервисные запросы, представляющие собой грубую аналогию с компьютерными сигналами прерывания. Тем не менее отличие от компьютерных сигналов прерывания заключается в том, что сервисные запросы не имеют независимого механизма для передачи СПП информации о

существующем условии, кроме случаев, когда протокол состояния-события включен (см. 7.1.1.11). Бит «Service request» («Запрос сервиса») может быть считан при помощи команды «Read Status-Event Register» («Считать регистр состояния-события») (см. 7.1.1.8). Это означает, что СПП не способен в минимальное время ответить на сервисный запрос. Также запрос может быть отправлен обратно, если он был подтвержден через сообщение протокола состояния.

5.14.1 Маски сервисного запроса

ИМП должен содержать регистр маски сервисного запроса для самого ИМП и для каждого используемого канала преобразователя внутри ИМП. Ширина таких регистров составляет 4 байта. Для того чтобы установить бит «Service request» («Запрос сервиса») после установки соответствующего бита в регистре состояния, нужно записать единицу в любой бит регистра маски сервисного запроса. Детальное описание представлено на рисунке 11.

Положение битов регистра маски сервисного запроса полностью соответствует положению битов в регистре состояния-события, определенных в таблице 9. Значение, расположенное в 0-битовой позиции регистра маски сервисного запроса, не используется, так как данное положение соответствует биту «Service request» («Запрос сервиса»). Бит «Service request» («Запрос сервиса») не может быть маскирован, так как он непосредственно создает сервисный запрос. При включенном питании значение по умолчанию для регистров масок сервисного запроса равно только нулям (то есть ни один бит состояния не может создавать сервисный запрос). На значение данного регистра не должно оказывать влияние ни получение протокола «Device Clear» («Очистка устройства»), ни команда «Reset» («Перезагрузка»).

5.14.2 Сервисные запросы

Сигнал сервисного запроса применяется в сочетании с регистрами состояния и регистрами масок сервисного запроса для отображения особых условий ИМП. Как правило, СПП считывает состояние всех каналов преобразователя для того, чтобы определить, какой канал преобразователя запрашивает сервис и по какой причине. СПП не обязательно должен немедленно отвечать на сервисный запрос.

Если установлено автоматическое сообщение о состоянии, то ИМП или любой канал, запрашивающий сервис, должен вернуть регистр состояния-события через сообщение протокола состояния-события.

На рисунке 12 показана логика генерирования сервисного запроса ИМП.

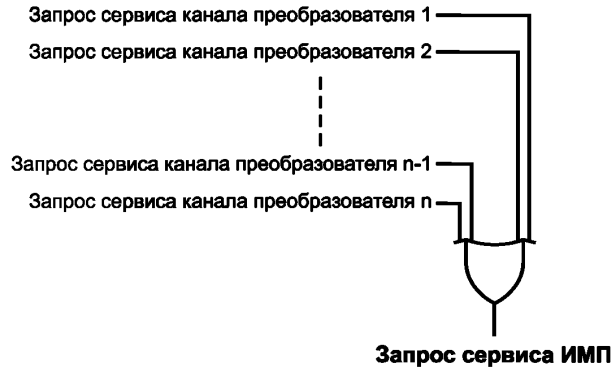


Рисунок 12 — Генерация сервисного запроса ИМП

5.15 Возможность горячей замены

В системе должна существовать возможность отсоединения ИМП от интерфейса или подсоединения ИМП к интерфейсу без отключения питания системы и без повреждения подключаемого ИМП или другого устройства, подключенного к среде. При подсоединении ИМП к среде связи во время передачи данных или его отсоединении допускаются кратковременные воздействия. При этом необратимые повреждения недопустимы.

Возможность СПП обнаруживать добавленный или отсоединенный ИМП поддерживается настоящим стандартом, но при этом зависит от способности физических слоев более низкого уровня обнаруживать подобные изменения и вызывать соответствующие методы в API модульных связей (см. 11.5—11.6).

6 Структура сообщений

Данный раздел определяет структуру сообщений, пересылаемых через интерфейс модульных связей.

6.1 Порядок передачи данных. Значимость битов

Порядок передачи заголовков и данных, описанных в настоящем стандарте, решен на уровне байтов. В случае если диаграмма представляет группу байтов, они передаются в обычном порядке, то есть в порядке их прочтения в русском или английском языке. Например, представленные в таблице 10 байты передаются в порядке их нумерации.

Примечание — Данный порядок передачи данных применим только к интерфейсу модульных связей и является исключительно абстрактным. Порядок передачи данных и значимость битов для физического уровня могут отличаться.

Таблица 10 — Порядок передачи байтов

1 байт								1 байт								1 байт								1 байт							
7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
1 (первый передаваемый байт)								2								3								4							
5								6								7								8							
9								—																							

В случае если байт представляет числовое значение, крайний левый бит, представленный в таблице 11, является битом высшего порядка или старшим значащим битом, то есть бит под номером 7 является старшим значащим битом. Например, числовое значение в таблице 11 представляет собой число 170 (десятичное число) или 0xAA (шестнадцатеричное число).

Таблица 11 — Пример значимости битов

	Биты							
	7	6	5	4	3	2	1	0
Значение = 170 (десятичное)	1	0	1	0	1	0	1	0

6.2 Структура командных сообщений

Формат командных сообщений представлен в таблице 12.

Таблица 12 — Структура командных сообщений

1 байт							
7	6	5	4	3	2	1	0
Номер канала-получателя (старший значащий байт)							
Номер канала-получателя (младший значащий байт)							
Класс команды							
Функция команды							
Длина (старший значащий байт)							
Длина (младший значащий байт)							
Командно-зависимые байты . . .							

6.2.1 Поле «Номер канала-получателя для преобразователя»

Данное поле содержит 16-битный номер канала преобразователя, который является получателем сообщения.

6.2.2 Поле «Класс команд»

Класс команд определен в 7.1, главный индекс классов команд представлен в таблице 15.

6.2.3 Поле «Функция команд»

Функция команд определена в разделе 7. Функция команды должна рассматриваться в соответствии с классом команды, как описано в рамках раздела 7.

6.2.4 Поле «Длина»

Длина — это число командно-зависимых байтов в данном сообщении. Если длина полученного сообщения не совпадает с заданной в данном поле длиной получаемых сообщений, то такое сообщение должно быть отклонено и должен быть установлен бит «Protocol error» («Ошибка протокола») в регистре состояний (см. 5.13.9).

6.2.5 Поле «Командно-зависимые байты»

Данное поле содержит информацию, которая должна быть определена для команд. Подробное описание команд и указанной информации представлено в разделе 7.

6.3 Ответные сообщения

Ответные сообщения используются для ответа на полученные команды. Формат ответных сообщений приведен в таблице 13.

Таблица 13 — Структура ответных сообщений

1 байт							
7	6	5	4	3	2	1	0
Флаг (индикатор) удача/сбой							
Длина (старший значащий байт)							
Длина (младший значащий байт)							
Байты, зависимые от ответного сообщения . . .							

6.3.1 Поле «Флаг «удача/сбой»

Ненулевое значение данного байта указывает на успешное завершение команды. Нулевое значение байта показывает, что для данной команды произошел сбой, при этом системе следует провести проверку состояния для определения причины сбоя.

6.3.2 Поле «Длина»

Длина — это число зависимых от ответного сообщения байтов в данном сообщении. Если длина полученного сообщения не совпадает с заданной в данном поле длиной получаемых сообщений, то такое сообщение должно быть отклонено и должен быть установлен бит «Protocol error» («Ошибка протокола») в регистре состояний (см. 5.13.9).

6.3.3 Поле «Зависимые от ответного сообщения байты»

Данное поле содержит информацию, которая должна быть определена для команд. Подробное описание команд и указанной информации представлено в разделе 7.

6.4 Структура сообщений, инициируемых ИМП

Формат сообщений, запускаемых ИМП, представлен в таблице 14. Примерами таких сообщений являются потоковые данные или сообщения о статусе.

Таблица 14 — Структура сообщений, инициируемых ИМП

1 байт							
7	6	5	4	3	2	1	0
Номер канала-отправителя (старший значащий байт)							
Номер канала-отправителя (младший значащий байт)							
Класс команды							
Функция команды							
Длина (старший значащий байт)							
Длина (младший значащий байт)							
Командно-зависимые байты . . .							

6.4.1 Поле «Номер канала-отправителя для преобразователя»

Данное поле содержит 16-битный номер канала преобразователя, который является источником (отправителем) сообщения.

6.4.2 Поле «Класс команд»

Данное поле полностью повторяет поле, описанное в 6.2.2.

6.4.3 Поле «Функция команд»

Данное поле полностью повторяет поле, описанное в 6.2.3.

6.4.4 Поле «Длина»

Данное поле полностью повторяет поле, описанное в 6.2.4.

6.4.5 Поле «Командно-зависимые байты»

Данное поле полностью повторяет поле, описанное в 6.2.5.

7 Команды

Команды подразделяются на две категории: стандартные и определенные изготовителем. Вне зависимости от категории команда разделена на два байта. Старший значащий байт должен использоваться для определения класса команды. Младший значащий байт, называемый функцией, должен определять конкретную команду в рамках определенного класса. Например, в случае если старший значащий байт определяет класс команд режима ожидания преобразователя, то младший значащий байт определяет конкретную команду внутри данного класса, при этом их возможные варианты перечислены в таблице 25.

Различные классы команд перечислены в таблице 15.

Таблица 15 — Классы стандартных команд

cmdClassId (идентификатор класса команды)	Наименование атрибута	Категория
0	Reserved	Зарезервировано
1	CommonCmd	Общие команды для ИМП и канала преобразователя
2	XdcrIdle	Режим ожидания преобразователя
3	XdcrOperate	Рабочий режим преобразователя
4	XdcrEither	Режим ожидания или рабочий режим преобразователя
5	TIMsleep	Спящий режим ИМП
6	TIMActive	Команды активного режима ИМП

Окончание таблицы 15

cmdClassId (идентификатор класса команды)	Наименование атрибута	Категория
0	Reserved	Зарезервировано
7	AnyState	Любой режим
8—127	ReservedClass	Зарезервировано
128—255	ClassN	Открыто для изготовителей, N = номер класса

ИМП может генерировать ответ на команды в случае выполнения одного из двух условий. Первое условие: сама команда требует ответа. Например, в случае получения команды «Query TEDS» («Запросить ЭТДП»). Второе условие: активирован протокол состояния-события. В данном случае канал преобразователя или ИМП передает ответы каждый раз, когда в регистре состояния происходит не скрытое маской изменение.

Информация, описываемая как «передаваемая», передается с использованием структуры командных сообщений, приведенной в 6.2. В следующих пунктах описывается содержание командно-зависимых октетов, приведенных в таблице 12. В случае приема или передачи ответа на команду применяется структура ответных сообщений (см. 6.3).

В таблицах данного раздела присутствует нумерация для полей, «зарезервированных» для будущих версий настоящего стандарта. Некоторые типы нумерации являются необязательными, и возможность их применения остается на усмотрение изготовителя ИМП. Нумерация, обозначенная как «открыто для изготовителя», может быть использована для описания условий, лежащих за пределами рассмотрения настоящего стандарта.

7.1 Стандартные команды

Контрольные функции позволяют отправлять команды ИМП в целом либо каждому каналу преобразователя по отдельности, в результате чего происходит изменение их режима работы. Список классов стандартных команд приведен в таблице 15.

ИМП должен отвечать на любые невыполнимые команды установкой бита «Invalid command» («Неверная команда») ИМП в регистре состояния. Более полное описание данного бита см. в 5.13.3[1)].

7.1.1 Общие команды для ИМП и канала преобразователя

Команды данного класса могут быть адресованы и для ИМП, и для канала преобразователя. Для того чтобы определить, кому предназначается команда — ИМП или каналу преобразователя, используется адрес (см. 5.3). Команды данного класса перечислены в таблице 16. Команды данного класса не должны быть адресованы какой-либо адресной группе, прокси-каналу преобразователя или иметь глобальный адрес. В случае если команда данного класса адресована адресной группе, прокси-каналу преобразователя или имеет глобальный адрес, она должна быть проигнорирована, а в регистре состояния-условия ИМП должен быть установлен бит «Command rejected» («Отказ от выполнения команды»).

Таблица 16 — Общие команды

cmdFunctionId (идентификатор функции команды)	Команда	Состояние		Ответ ожидается	Обязательная/ необязательная
		Канала преобразователя	ИМП		
0	Зарезервировано	—	—	—	—
1	Запросить ЭТДП	Любое	Активен	Да	Обязательная
2	Считать сегмент ЭТДП	Любое	Активен	Да	Обязательная
3	Записать сегмент ЭТДП	Любое	Активен	Нет	Обязательная

[1)] В оригинале ISO/IEC/IEEE 21450: 2010 допущена ошибка. Ошибочно приведена ссылка на 5.13.2.

Окончание таблицы 16

cmdFunctionId (идентификатор функции команды)	Команда	Состояние		Ответ ожидается	Обязательная/ необязательная
		Канала преобразователя	ИМП		
4	Обновить ЭТДП	Любое	Активен	Да	Обязательная
5	Запустить самодиагностику	Режим ожидания	Активен	Нет	Обязательная
6	Записать маску сервисного запроса	Любое	Активен	Нет	Обязательная
7	Считать маску сервисного запроса	Любое	Активен	Да	Обязательная
8	Считать регистр состояния-события	Любое	Активен	Да	Обязательная
9	Считать регистр состояния-условия	Любое	Активен	Да	Обязательная
10	Очистить регистр состояния-события	Любое	Активен	Нет	Обязательная
11	Записать состояние протокола состояния-события	Режим ожидания	Активен	Нет	Обязательная
12	Считать состояние протокола состояния-события	Любое	Активен	Да	Обязательная
13—127	Зарезервировано	—	—	—	—
128—255	Открыто для изготовителей	—	—	—	—

7.1.1.1 Команда «Query TEDS» («Запросить ЭТДП»)

Имя атрибута аргумента: TEDSAccessCode data type UInt8.

Данная команда используется СПП для запроса информации, необходимой для считывания или записи ЭТДП. Для данной команды существует один-единственный аргумент: «TEDS Access Code» («Код доступа к ЭТДП»), который определяет ЭТДП, к которой должен быть осуществлен доступ. Коды доступа к ЭТДП, определенным в настоящем стандарте, перечислены в таблице 17.

Ответ на команду «Query TEDS» («Запросить ЭТДП») должен содержать информацию, приведенную в таблице 18. Требуется, чтобы ИМП обеспечивал ответ на все команды запроса ЭТДП вне зависимости от того, выбрана ли кодом доступа выполняемая ЭТДП. Поле атрибутов в ответном сообщении (см. таблицу 19) должно отображать, поддерживается данная ЭТДП или нет, может ли она быть изменена, верно ли ее текущее содержание, а также является ли ЭТДП встроенной в ИМП или расположенной удаленно (виртуальной). Описание поля состояния ЭТДП приведено в таблице 20.

В случае если установлен атрибут неподдерживаемой ЭТДП, то ИМП в ответном сообщении должен установить «ноль» для атрибутов «TEDSSize» («Размер ЭТДП») и «MaxTEDSSize» («Максимальный размер ЭТДП»).

В случае если установлен атрибут виртуальной ЭТДП, то также должен быть установлен атрибут «Read-only» («Только для чтения»), и ИМП в ответном сообщении также должен установить «ноль» для атрибутов «TEDSSize» («Размер ЭТДП»), «TEDSCkSum» («Контрольная сумма ЭТДП») и «MaxTEDSSize» («Максимальный размер ЭТДП»). Определение размеров и атрибутов в ответном сообщении для приложения, отправившего запрос, осуществляет СПП или главный процессор. В случае если файл не может быть найден, то должен быть установлен атрибут «Unsupported» («Не поддерживается»), и ИМП в ответном сообщении должен установить «ноль» для атрибутов «TEDSSize» («Размер ЭТДП»), «TEDSCkSum» («Контрольная сумма ЭТДП») и «MaxTEDSSize» («Максимальный размер ЭТДП»). В случае если файл найден, атрибуты «Invalid» («Недействительный») и «Unsupported» («Не поддерживается») должны быть очищены, а атрибуты «Read-Only» («Только для чтения»), «TEDSSize» («Размер ЭТДП»), «TEDSCkSum» («Контрольная сумма ЭТДП») и «MaxTEDSSize» («Максимальный размер ЭТДП») должны быть определены согласно атрибутам удаленного файла, в котором хранится ЭТДП.

Таблица 17 — Коды доступа к ЭТДП

Код доступа к ЭТДП	Наименование атрибута ЭТДП	ЭТДП	Обязательная/ необязательная
0	—	Зарезервировано	—
1	MetaTEDS	Мета-ЭТДП1	Обязательная
2	MetaIdTEDS	ЭТДП2 мета-идентификации	Необязательная
3	ChanTEDS	ЭТДП1 канала преобразователя	Обязательная
4	ChanIdTEDS	ЭТДП2 идентификации канала преобразователя	Необязательная
5	CalTEDS	ЭТДП1 калибровки	Необязательная
6	CalIdTEDS	ЭТДП2 идентификации калибровки	Необязательная
7	EUASTEDS	ЭТДП3 специальных приложений конечного пользователя	Необязательная[1)]
8	FreqRespTEDS	ЭТДП1 частотной характеристики	Необязательная
9	TransferTEDS	ЭТДП1 передаточной функции	Необязательная
10	CommandTEDS	Командная ЭТДП2	Необязательная
11	TitleTEDS	ЭТДП2 места нахождения и заголовка	Необязательная
12	XdcrName	ЭТДП2 имени преобразователя, установленного пользователем	Обязательная
13	PHYTEDS	ЭТДП1 физического уровня	Обязательная
14	GeoLocTEDS	ЭТДП2 географического места нахождения	Необязательная
15	UnitsExtention	ЭТДП2 с расширенным набором единиц измерения	Необязательная
16—127	—	Зарезервировано	—
128—255	—	ЭТДП, заданная изготовителем	Необязательная

Примечание —
ЭТДП1 — двоичная ЭТДП;
ЭТДП2 — текстовая ЭТДП;
ЭТДП3 — содержание, определяемое пользователем.

Таблица 18 — Поле данных ответного сообщения на запрос ЭТДП

Поле	Тип данных	Наименование атрибута поля	Функция
1	UInt8	TEDSAttrib	Атрибуты ЭТДП (см. таблицу 19)
2	UInt8	TEDSStatus	Состояние (статус) ЭТДП
3	UInt32	TEDSSize	Текущий размер ЭТДП
4	UInt16	TEDSCkSum	Контрольная сумма ЭТДП
5	UInt32	MaxTEDSSize	Максимальный размер ЭТДП

[1)] В оригинале ISO/IEC/IEEE 21450:2010 допущена ошибка. Ошибочно приведено значение «обязательная», что не соответствует описанию в 5.5.2.9 и 5.5.2.

Таблица 19 — Атрибуты ЭТДП

Бит	Тип данных	Наименование атрибута поля	Определение
0 (МЗБ, lsb)	Логический	TEDSAttrib.ReadOnly	Только для чтения — устанавливается на значение «True» («Истина»), если ЭТДП может быть считана, но не может быть записана
1	Логический	TEDSAttrib.NotAvail	Не поддерживается — устанавливается на значение «True» («Истина»), если ЭТДП не поддерживается данным каналом преобразователя
2	Логический	TEDSAttrib.Invalid	Недействительная — устанавливается на значение «True» («Истина»), если текущее отображение ЭТДП не верно
3	Логический	TEDSAttrib.Virtual	Виртуальная ЭТДП — данный бит устанавливается на значение «True» («Истина»), если данная ЭТДП является виртуальной. (Виртуальной ЭТДП является любая ЭТДП, которая не хранится внутри ИМП. За доступ к виртуальной ЭТДП отвечает СПП или главный процессор)
4	Логический	TEDSAttrib.TextTEDS	Текстовая ЭТДП — устанавливается на значение «True» («Истина»), если ЭТДП является текстовой
5	Логический	TEDSAttrib.Adaptive	Адаптивная — устанавливается на значение «True» («Истина»), если содержание ЭТДП может быть изменено ИМП или каналом преобразователя без получения от СПП команды «Записать сегмент ЭТДП»
6	Логический	TEDSAttrib.MfgrDefine	MfgrDefine (определен изготовителем) — устанавливается на значение «True» («Истина»), если содержание ЭТДП определяется изготовителем. Соответствие структурам, определенным в настоящем стандарте, достигается, если также установлен атрибут текстовой ЭТДП
7 (СЗБ, msb)	Логический	TEDSAttrib.Reserved[7]	Зарезервировано

Таблица 20 — Состояние (статус) ЭТДП

Бит	Тип данных	Наименование атрибута поля	Определение
0 (МЗБ, lsb)	Логический	TEDSStatus.TooLarge	Слишком большой размер — размер последнего полученного отображения ЭТДП превышает объем памяти, отведенный для данной ЭТДП
1	Логический	TEDSStatus Reserved[1]	Зарезервировано
2	Логический	TEDSStatus Reserved[2]	Зарезервировано
3	Логический	TEDSStatus Reserved[3]	Зарезервировано
4	Логический	TEDSStatus Open[4]	Открыто для изготовителей
5	Логический	TEDSStatus Open[5]	Открыто для изготовителей
6	Логический	TEDSStatus Open[6]	Открыто для изготовителей
7 (СЗБ, msb)	Логический	TEDSStatus Open[7]	Открыто для изготовителей

7.1.1.2 Команда «Read TEDS segment» («Считать сегмент ЭТДП»)

Данная команда используется для считывания ЭТДП в СПП. Аргументы для данной команды приведены в таблице 21. Так как максимальный размер байтового массива меньше максимального раз-

мера ЭТДП, то для доступа к месту ЭТДП, с которого должно начинаться считывание, используется смещение сегмента ЭТДП относительно ее начала.

Примечание — Размер большинства ЭТДП не превышает размера одного сегмента. В таких случаях смещение сегмента ЭТДП должно быть равно нулю. Тем не менее допускаются ЭТДП с размером больше максимального размера байтового массива данных. ЭТДП такого объема требуют сегментирования для передачи.

Таблица 21 — Поле данных команды «Read TEDS segment» («Считать сегмент ЭТДП»)

Поле	Тип данных	Наименование атрибута поля	Функция
1	UInt8	TEDSAccessCode	Код доступа к ЭТДП, как определено в таблице 17
2	UInt32	TEDSOffset	Смещение сегмента ЭТДП (значения от 0 до [текущий размер минус 1]) — это адрес относительно начала ЭТДП, с которого должно происходить считывание блока данных

Для ответа на команду «Read TEDS segment» («Считать сегмент ЭТДП») используется ответное сообщение (см. 6.3). Изменяемые байты в ответном сообщении должны соответствовать таблице 22. Первое поле содержит смещение в ЭТДП, для которого было произведено считывание блока данных, и в большинстве случаев величина данного смещения совпадает со значением смещения сегмента ЭТДП для команды «Read TEDS segment» («Считать сегмент ЭТДП»). Оставшиеся байты содержат данные, считанные с соответствующей ЭТДП. Ответное сообщение должно содержать только «единицы» в поле смещения сегмента ЭТДП и «нулевые» байты данных в случае виртуальных, не поддерживаемых или недействительных ЭТДП. Число байтов в ответном сообщении зависит от строения и определено в заголовке сообщения (см. 6.3). Если значение «TEDSOffset» («Смещение сегмента ЭТДП») больше длины ЭТДП, то значение «TEDSOffset» («Смещение сегмента ЭТДП») в ответном сообщении должно быть равно длине ЭТДП и ответное сообщение должно содержать «ноль» байтов.

Таблица 22 — Поле данных ответного сообщения для команды «Read TEDS segment» («Считать сегмент ЭТДП»)

Поле	Тип данных	Наименование атрибута поля	Функция
1	UInt32	TEDSOffset	Смещение сегмента ЭТДП (значения от 0 до [текущий размер минус 1])
2	OctetArray	RawTEDSBlock	Байты данных ЭТДП

7.1.1.3 Команда «Write TEDS segment» («Записать сегмент ЭТДП»)

Данная команда используется для записи части ЭТДП. Аргументы для данной команды приведены в таблице 23. Так как максимальный размер байтового массива данных меньше максимального размера ЭТДП, то для доступа к месту, с которого должна начаться запись данных поля 3, используется смещение сегмента ЭТДП. Если значение «TEDSOffset» («Смещение сегмента ЭТДП») больше максимальной длины ЭТДП, то данные должны быть отклонены, и в слове состояния должен быть установлен бит «Command rejected» («Отказ от выполнения команды») (см. 5.13.4).

Примечание — Размер большинства ЭТДП не превышает размера одного сегмента. В таких случаях содержание поля смещения сегмента ЭТДП должно быть равно нулю. Тем не менее допускаются размеры ЭТДП, превышающие максимальный размер байтового массива данных. ЭТДП такого объема требуют сегментирования для передачи.

Таблица 23 — Поле данных команды «Write TEDS segment» («Записать сегмент ЭТДП»)

Поле	Тип данных	Наименование атрибута поля	Функция
1	UInt8	TEDSAccessCode	Код доступа к ЭТДП, как определено в таблице 17
2	UInt32	TEDSOffset	Смещение сегмента ЭТДП (значения от 0 до [текущий размер минус 1])
3	OctetArray	RawTEDSBlock	Содержание ЭТДП

Для команды «Write TEDS segment» («Записать сегмент ЭТДП») используется структура командного сообщения (см. 6.2). В случае если превышен максимальный размер ЭТДП, дополнительные данные не должны записываться в память, а текущий размер ЭТДП должен быть установлен равным нулю.

Команда «Write TEDS segment» («Записать сегмент ЭТДП») должна создать новую ЭТДП с соответствующим кодом доступа, в случае если она еще не существует. В случае если устройство ИМП не позволяет создание ЭТДП, команда «Write TEDS segment» («Записать сегмент ЭТДП») не должна записывать какие-либо данные в память ЭТДП, так как данная ЭТДП не поддерживается.

Когда ИМП начинает перезапись существующей ЭТДП, перезаписываемая ЭТДП должна быть помечена как недействительная. Отметка о недействительной ЭТДП должна сохраняться до тех пор, пока не будет получена команда «Update TEDS» («Обновить ЭТДП»), описанная в 7.1.1.4.

Команда «Write TEDS segment» («Записать сегмент ЭТДП») не генерирует ответного сообщения.

7.1.1.4 Команда «Update TEDS» («Обновить ЭТДП»)

Наименование атрибута аргумента: TEDSAccessCode data type UInt8.

Данная команда используется для вызова предварительно записанной в канал преобразователя ЭТДП для ее проверки и копирования в энергонезависимую память (в случае если это не было сделано при получении ЭТДП). После проверки ЭТДП она может быть отмечена как действительная. Если проверка дает отрицательный результат, то отметка о недействительной ЭТДП должна сохраняться.

Для данной команды существует один аргумент: код доступа к ЭТДП, как определено в таблице 17.

Ответное сообщение для команды «Update TEDS» («Обновить ЭТДП») должно содержать информацию, которая имеется в ответном сообщении для команды «Query TEDS» («Запросить ЭТДП»), описанной в 7.1.1.1.

7.1.1.5 Команда «Run self-test» («Запустить самодиагностику»)

Наименование атрибута аргумента: Test2Run data type UInt8.

Данная команда используется для запуска процесса самодиагностики для устройства, которому она адресована. Для данной команды имеется один аргумент: 8-битовая нумерация, определяющая тип запускаемой диагностики, как показано в таблице 24. Изготовитель может установить дополнительные виды проверок, доступных для определенного устройства. В случае запроса неприменимой нумерации в соответствующем регистре состояния должен быть установлен бит состояния «Invalid command» («Неверная команда») (см. 5.13.3)[1)].

Таблица 24 — Нумерация для команды «Run self-test» («Запустить самодиагностику»)

Нумерация	Наименование атрибута аргумента	Диагностика
0	Test2Run.ConfidenceTest	Короткая проверка на достоверность
1	Test2Run.TestAll	Полная проверка
2—255	Test2Run.RunTest[N], 2≤N≤255	Определено изготовителем

Если изготовитель применяет дополнительные проверки, они должны быть перечислены в командных ЭТДП для канала преобразователя или ИМП.

Данная команда должна выполняться, только если канал преобразователя находится в режиме ожидания. Если данная команда получена, когда канал преобразователя находится в любом другом режиме, то должен быть установлен бит «Command rejected» («Отказ от выполнения команды») (см. 5.13.4).

Если изготовитель намеревается записать единственную программу диагностики, проверяющую ИМП и каналы преобразователя, то такая команда должна быть адресована ИМП, при этом все каналы преобразователя должны находиться в режиме ожидания. В противном случае команда должна быть отклонена, и установлен бит «Command rejected» («Отказ от выполнения команды»).

7.1.1.6 Команда «Write service request mask» («Записать маску сервисного запроса»)

Наименование атрибута аргумента: SRMask data type UInt32.

Данная команда используется для записи маски сервисного запроса ИМП или канала преобразователя, которым она была адресована. Маска представляет собой 32-битовое слово, которое используется согласно описанию, приведенному в 5.14.1.

[1)] В оригинале ISO/IEC/IEEE 21450:2010 допущена ошибка. Ошибочно приведена ссылка на 5.13.2.

7.1.1.7 Команда «Read service request mask» («Считать маску сервисного запроса»)

Наименование атрибута аргумента ответа: SRMask data type UInt32.

Данная команда используется для считывания маски сервисного запроса с ИМП или канала преобразователя, которым она была адресована. Данная команда не имеет аргументов. Тем не менее она вызывает ответное сообщение, имеющее единственный аргумент. Данный аргумент передает текущую активную маску сервисного запроса согласно 5.14.1 для канала преобразователя, которому была адресована исходная команда.

7.1.1.8 Команда «Read status-event register» («Считать регистр состояния-события»)

Наименование атрибута аргумента ответа: SERegister data type UInt32.

Данная команда используется для считывания состояния ИМП или канала преобразователя, которым она была адресована. Данная команда не имеет аргументов. Тем не менее она вызывает ответное сообщение, имеющее единственный аргумент. Данный аргумент сообщает текущее содержание регистра состояния-события согласно 5.13 для канала преобразователя, которому была адресована исходная команда.

7.1.1.9 Команда «Read status-condition register» («Считать регистр состояния-условия»)

Наименование атрибута аргумента ответа: SCRegister data type UInt32.

Данная команда используется для считывания регистра состояния-условия ИМП или канала преобразователя, которым она была адресована. Данная команда не имеет аргументов. Тем не менее она вызывает ответное сообщение, имеющее единственный аргумент. Данный аргумент сообщает текущее содержание регистра состояния-условия согласно 5.13 для канала преобразователя, которому была адресована исходная команда.

7.1.1.10 Команда «Clear status-event register» («Очистить регистр состояния-события»)

Данная команда используется для очистки регистра состояния-события ИМП или канала преобразователя, которым она адресована. Если команда адресована для всего ИМП, должны быть очищены все регистры состояния-события самого ИМП и все регистры состояния-события всех каналов преобразователя. Команда не очищает соответствующие маски и регистры условия. Данная команда не имеет аргументов.

7.1.1.11 Команда «Write status-event protocol state» («Записать состояние протокола состояния-события»)

Наименование атрибута аргумента: SEProtocol data type Boolean.

Данная команда используется для включения или отключения протокола состояния-события, приведенного в 5.13. Если протокол включен, должен быть инициализирован поток данных для отправки 32-битового регистра состояния при каждой отметке бита «Service request» («Запрос сервиса»). Следует заметить, что если канал запрашивает сервис, то должен быть отправлен регистр канала, а в случае если сервис запрашивает сам ИМП, должен быть отправлен регистр состояния ИМП.

Данная команда имеет единственный аргумент. Если его значение «True» («Истина»), то протокол состояния-события включен. В противном случае протокол состояния-события отключен.

7.1.1.12 Команда «Read status-event protocol state» («Считать состояние протокола состояния-события»)

Наименование атрибута аргумента ответа: SEProtocol data type Boolean.

Данная команда используется для считывания состояния протокола состояния-события, описанного в 5.13. Данная команда не имеет аргументов. Тем не менее она вызывает ответное сообщение, имеющее единственный аргумент. Данный аргумент содержит текущее состояние протокола состояния-события. Если его значение «True» («Истина»), протокол состояния-события включен. В противном случае протокол состояния-события отключен.

7.1.2 Команды для преобразователя в режиме ожидания

Команды из класса команд режима ожидания, перечисленные в таблице 25, должны выполняться, только когда канал преобразователя находится в режиме ожидания. Если какая-либо из таких команд получена, когда канал преобразователя находится в любом другом режиме, то она должна быть проигнорирована, и в регистре состояния-условия канала преобразователя (см. 5.13) должен быть установлен бит «Command rejected» («Отказ от выполнения команды»).

Для всех команд данного класса требуется, чтобы значение номера канала-получателя для преобразователя было больше нуля. В случае если байтовый массив номеров каналов-получателей преобразователя (см. 5.3) содержит нулевое значение, команда должна быть проигнорирована, а в регистре состояния-условия канала преобразователя (см. 5.13) должен быть установлен бит «Command rejected» («Отказ от выполнения команды»).

Если команда для преобразователя в режиме ожидания была отправлена каналу преобразователя, который не поддерживает такую функцию или ее изменение, то такая команда должна быть проигнорирована, и в слове состояния должен быть установлен бит «Invalid command» («Неверная команда») (см. 5.13.2).

Для получения какой-либо из таких команд каналом преобразователя, ИМП должен находиться в активном режиме. Если такая команда была получена, когда ИМП находился в неактивном режиме, то команда должна быть проигнорирована, и в регистре состояния-условия канала преобразователя (см. 5.13) должен быть установлен бит «Command rejected» («Отказ от выполнения команды»).

Команды, обозначенные в таблице 25 как «необязательные», становятся обязательными в случае, если функция, которую они поддерживают, является программируемой.

Некоторые команды данного класса могут быть отправлены для прокси-канала преобразователя (если в таблице 25 в графе «Прокси» отмечено «Да»). Если какая-либо из таких команд отправлена прокси-каналу преобразователя, она должна действовать также, как если бы была отправлена каждому члену прокси-канала преобразователя по отдельности.

Таблица 25 — Команды для канала преобразователя в режиме ожидания

cmdFunctionId (идентификатор функции команды)	Команда	Класс адреса			Ответ ожидается	Обязательная/ необязательная
		Канал преобразователя	Прокси	Группа		
0	Зарезервировано	—	—	—	—	—
1	Установить число повторений данных канала преобразователя	Да	Нет	Нет	Нет	Необязательная
2	Установить предварительное значение счетчика триггера	Да	Нет	Нет	Нет	Необязательная
3	Установить адресную группу	Да	Да	Нет	Нет	Обязательная
4	Режим выборки данных	Да	Да	Нет	Нет	Необязательная
5	Режим передачи данных	Да	Да	Да	Нет	Необязательная
6	Состояние буферизации	Да	Да	Нет	Нет	Необязательная
7	Рабочий режим «набор данных закончен»	Да	Да	Нет	Нет	Необязательная
8	Режим остановки исполнительного устройства	Да	Да	Нет	Нет	Необязательная
9	Сообщить о достижении порогового значения	Да	Да	Нет	Нет	Необязательная
10	Калибровка канала преобразователя	Да	Да	Да	Да	Необязательная
11	Обнулить канал преобразователя	Да	Да	Да	Да	Необязательная
12	Записать состояние коррекции данных	Да	Да	Нет	Нет	См. 7.1.2.12
13	Считать состояние коррекции данных	Да	Да	Нет	Нет	См. 7.1.2.13
14	Записать состояние канала преобразователя с инициализацией триггера	Да	Нет	Нет	Нет	Необязательная
15	Записать конфигурацию канала преобразователя с инициализацией триггера	Да	Нет	Нет	Нет	Необязательная
16—127	Зарезервировано	—	—	—	—	—
128—255	Открыто для изготовителей	—	—	—	—	—

Примечание — Необязательные команды становятся обязательными, если функции, которые они поддерживают, являются программируемыми.

7.1.2.1 Команда «Set Transducer Channel data repetition count» («Установить число повторений данных канала преобразователя»)

Наименование атрибута аргумента: RepCount data type UInt16.

Данная команда используется для изменения числа выборок в наборе данных от нуля до значения в поле «Максимальное повторение данных» в ЭТДП канала преобразователя (см. 8.5.2.28).

Данная команда имеет единственный аргумент: 16-битовое целое число, представляющее собой число выборок в наборе данных. Если данное значение превышает значение максимального повторения данных, то число повторений данных не должно меняться, а в регистре состояния канала преобразователя должен быть установлен бит «Command rejected» («Отказ от выполнения команды») (см. 5.13.4).

7.1.2.2 Команда «Set Transducer Channel pre-trigger count» («Установить предварительное значение счетчика триггера»)

Наименование атрибута аргумента: PreTrigCount data type UInt16.

Данная команда используется каналами преобразователя, способными работать в режиме автономной выборки данных с предварительно заданным счетчиком триггера (см. 5.10.1.3), для изменения числа выборок в наборе данных, которые могут быть получены и сохранены до получения триггерного сигнала. Число выборок может иметь значение от нуля до значения в поле «Максимальное число выборок с предварительным триггером» в ЭТДП канала преобразователя (см. 8.5.2.32).

Данная команда имеет единственный аргумент: 16-битовое целое число, представляющее собой число выборок в наборе данных, которые должны быть собраны до получения триггерного сигнала. Если данное число превышает значение поля «Максимальное число выборок с предварительным триггером» (см. 8.5.2.32), то предварительное значение счетчика триггера не должно меняться, а в регистре состояния канала преобразователя должен быть установлен бит «Command rejected» («Отказ от выполнения команды») (см. 5.13.4).

7.1.2.3 Команда «AddressGroup definition» («Установить адресную группу»)

Наименование атрибута аргумента: GroupAdd data type UInt16.

Данная команда приписывает канал преобразователя к адресной группе и имеет единственный аргумент в поле данных, представляющий собой адрес группы. Детальная информация об адресах приведена в таблице 5. Таким образом, обычный или прокси-канал преобразователя, которому адресована команда, приписывается к группе. Если аргумент данной команды равен нулю, канал преобразователя, которому была адресована команда, удаляется из всех адресных групп, к которым он был приписан.

7.1.2.4 Команда «Sampling mode» («Режим выборки данных»)

Наименование атрибута аргумента: SampleMode data type UInt8.

Данная команда используется СПП для изменения режима выборки данных канала преобразователя. Не все допустимые режимы работы, определенные в настоящем стандарте, поддерживаются заданным каналом преобразователя. Допустимые режимы работы для заданного канала преобразователя основаны на атрибутах выборки и приведены в ЭТДП канала преобразователя (см. 8.5.2.44).

Данная команда имеет единственный аргумент. Список допустимых значений для данного аргумента приведен в таблице 26. В случае если команда «Sampling mode» («Режим выборки данных») отправлена какому-либо каналу преобразователя, который не поддерживает такой режим или изменения данного режима, то в слове состояния должен быть установлен бит «Invalid command» («Неверная команда») канала преобразователя (см. 5.13.2).

Таблица 26 — Допустимые значения для аргумента режима выборки

Нумерация	Наименование атрибута аргумента	Рабочий режим
0	SampleMode.Reserved[0]	Зарезервировано
1	SampleMode.TriggerInit	Инициированный триггером
2	SampleMode.FreeNoPre	Автономный без предварительного триггера
3	SampleMode.FreePreTrig	Автономный с предварительным триггером
4	SampleMode.Continuous	Непрерывная выборка
5	SampleMode.Immediate	Немедленная работа
6—127	SampleMode.Reserved[N], 6≤N≤127	Зарезервировано
128—255	SampleMode.Open[N], 128≤N≤255	Открыто для изготовителей

7.1.2.5 Команда «Data transmission mode» («Режим передачи данных»)

Наименование атрибута аргумента: XmitMode data type UInt8.

Данная команда используется для контроля режима передачи данных, описанного в 5.10.2. Сведения о возможности заданного канала преобразователя поддерживать допустимые рабочие режимы передачи данных представлены в атрибуте передачи данных ЭТДП канала преобразователя (см. 8.5.2.49).

Данная команда имеет единственный аргумент. Список допустимых значений для данного аргумента приведен в таблице 7. В случае если команда «Data transmission mode» («Режим передачи данных») отправлена какому-либо каналу преобразователя, который не поддерживает такой режим или изменения данного режима, в слове состояния должен быть установлен бит «Invalid command» («Неверная команда») канала преобразователя (см. 5.13.2).

7.1.2.6 Команда «Buffered state» («Состояние буферизации»)

Наименование атрибута аргумента: BufferOnOff data type Boolean.

Данная команда используется для включения или отключения режима буферизации, описанного в 5.10.3. Сведения о возможности заданного канала преобразователя поддерживать режим буферизации представлены в атрибуте буферизации ЭТДП канала преобразователя (см. 8.5.2.47).

Данная команда имеет единственный аргумент. Если данный аргумент принимает значение «True» («Истина»), то режим буферизации включен. В противоположенном случае режим буферизации отключен.

7.1.2.7 Команда «End-of-Data-Set operation mode» («Рабочий режим «набор данных закончен»)

Наименование атрибута аргумента: EndDataSetMode data type UInt8.

Данная команда используется для исполнительных устройств, чтобы сообщить, какое действие должно предпринять исполнительное устройство по достижении окончания набора данных, как описано в 5.10.4. Сведения о возможности заданного исполнительного устройства поддерживать рабочий режим «End-of-Data-Set» («Набор данных закончен») и допустимые режимы представлены в атрибуте «End-of-Data-Set» («Набор данных закончен») ЭТДП канала преобразователя (см. 8.5.2.48).

Данная команда имеет единственный аргумент. Список допустимых значений для данного аргумента приведен в таблице 27. В случае если команда «End-of-Data-Set operation mode» («Рабочий режим «набор данных закончен») отправлена какому-либо каналу преобразователя, который не поддерживает такой режим или его изменения, в слове состояния должен быть установлен бит «Invalid command» («Неверная команда») канала преобразователя (см. 5.13.2).

Таблица 27 — Список допустимых значений для аргумента рабочего режима «End-of-Data-Set» («Набор данных закончен»)

Нумерация	Наименование атрибута аргумента	Рабочий режим
0	EndDataSetMode.Reserved[0]	Зарезервировано
1	EndDataSetMode.Hold	Удержать
2	EndDataSetMode.Recirc	Зациклить
3—127	EndDataSetMode.Reserved[N], 3≤N≤127	Зарезервировано
128—256	EndDataSetMode.Open[N], 128≤N≤255	Открыто для изготовителей

7.1.2.8 Команда «Actuator halt operating mode» («Режим остановки исполнительного устройства»)

Наименование атрибута аргумента: ActuatorHalt data type UInt8.

Данная команда используется для контроля работы выхода исполнительного устройства при получении команды «Transducer Channel Idle» («Перейти в режим ожидания канала преобразователя») во время обработки устройством набора данных. Возможные варианты работы заданного исполнительного устройства приведены в атрибуте «Actuator halt» («Остановка исполнительного устройства») ЭТДП канала преобразователя (см. 8.5.2.51).

Данная команда имеет единственный аргумент. Список допустимых значений для данного аргумента приведен в таблице 28.

Таблица 28 — Допустимые значения для аргумента режима «Actuator halt» («Остановка исполнительного устройства»)

Нумерация	Наименование атрибута аргумента	Рабочий режим
0	ActuatorHalt.Reserved[0]	Зарезервировано
1	ActuatorHalt.Hold	Удержать
2	ActuatorHalt.Finish	Завершить набор данных
3	ActuatorHalt.Ramp	Линейный
4—127	ActuatorHalt.Reserved[N], 4≤N≤127	Зарезервировано
128—256	ActuatorHalt.Open[N], 128≤N≤255	Открыто для изготовителей

7.1.2.9 Команда «Edge-to-Report» («Сообщить о достижении порогового значения»)

Наименование атрибута аргумента: EdgeReported data type UInt8.

Данная команда используется для назначения порогового значения или пороговых значений, о достижении которых должен сообщать датчик событий. Возможные варианты работы заданного датчика событий приведены в атрибуте «Edge-to-Report» («Сообщить о достижении порогового значения») (см. 8.5.2.50) ЭТДП канала преобразователя.

Данная команда имеет единственный аргумент. Список допустимых значений для данного аргумента приведен в таблице 29.

Таблица 29 — Допустимые величины для аргумента режима работы «Edge-to-Report» («Сообщить о достижении порогового значения»)

Нумерация	Наименование атрибута аргумента	Рабочий режим
0	EdgeReported.Reserved[0]	Зарезервировано
1	EdgeReported.Rising	Сообщать о переходе верхнего порогового значения
2	EdgeReported.Falling	Сообщать о переходе нижнего порогового значения
3	EdgeReported.BothEdges	Сообщать о переходе обоих пороговых значений
4—127	EdgeReported.Reserved[N], 4≤N≤127	Зарезервировано
128—256	EdgeReported.Open[N], 128≤N≤255	Открыто для изготовителей

7.1.2.10 Команда «Calibrate Transducer Channel» («Калибровать канал преобразователя»)

Данная команда используется для запуска каналом преобразователя процедуры проверки состояния на его выходах на предмет соответствия ожидаемым параметрам. Конкретные действия по отработке команды «Calibrate Transducer Channel» («Калибровать канал преобразователя») зависят от устройства канала преобразователя и не являются предметом рассмотрения настоящего стандарта.

Ответное сообщение на данную команду всегда должно генерироваться после завершения работы. Ответное сообщение должно быть таким же, как и для команды «Read Status-Event Register» («Считать регистр состояния-события» (см. 7.1.1.8).

7.1.2.11 Команда «Zero Transducer Channel» («Обнулить канал преобразователя»)

Данная команда используется для запуска процедуры обнуления входов или выходов канала преобразователя. Конкретные действия по отработке команды «Zero Transducer Channel» («Обнулить канал преобразователя») зависят от устройства канала преобразователя и не являются предметом рассмотрения настоящего стандарта.

Ответное сообщение на данную команду всегда должно создаваться после завершения работы. Ответное сообщение должно быть таким же, как и для команды «Read Status-Event Register» («Считать регистр состояния-события» (см. 7.1.1.8).

Примечание — Общим результатом при выполнении данной команды является нулевая величина на выходе датчика, которая затем используется в качестве нуля отсчета.

7.1.2.12 Команда «Write corrections state» («Записать состояние коррекции данных»)

Наименование атрибута аргумента: CorrectState data type UInt8.

Данная команда используется для каналов преобразователя, способных вносить поправки внутри ИМП, как описано в 8.6.1. В результате данной команды канал преобразователя включает процесс коррекции выходных данных физических величин, оговоренных в ЭТДП калибровки. Данная команда является обязательной для всех каналов преобразователя со встроенной функцией внесения поправок.

Данная команда имеет единственный аргумент. В случае если его значение «True» («Истина»), то режим коррекции данных включен. В противоположенном случае режим коррекции данных отключен.

После получения данной команды бит «Corrections disabled» («Коррекции отключены») (см. 5.13.15) должен быть очищен.

7.1.2.13 Команда «Read corrections state» («Считать состояние коррекции данных»)

Наименование атрибута аргумента: CorrectState data type UInt8.

Данная команда используется для каналов преобразователя, способных вносить поправки внутри ИМП. Данная команда является обязательной для всех каналов преобразователя со встроенной функцией внесения поправок.

Ответное сообщение на данную команду имеет единственный аргумент. В случае если его значение «True» («Истина»), то режим коррекции данных включен. В противоположенном случае режим коррекции данных отключен.

7.1.2.14 Команда «Write Transducer Channel initiate trigger state» («Записать состояние канала преобразователя с инициализацией триггера»)

Наименование атрибута аргумента: EventTrig data type Boolean.

Данная команда используется для датчика событий канала преобразователя и позволяет ему инициировать триггерную команду после того, как произошло событие. В случае если данную команду получает датчик или исполнительное устройство, то в слове состояния должен быть установлен бит «Command rejected» («Отказ от выполнения команды») (см. 5.13.4).

Данная команда имеет единственный аргумент. В случае если его значение «True» («Истина»), то для датчика событий доступна возможность инициировать сигнал триггера после того, как произошло событие. В противоположенном случае триггерный сигнал отключен.

7.1.2.15 Команда «Write Transducer Channel initiate trigger configuration» («Записать конфигурацию канала преобразователя с инициализацией триггера»)

Наименование атрибута аргумента: InitTrig data type Boolean.

Данная команда используется для предоставления датчику событий информации, необходимой для инициирования триггерной команды. Аргументы для данной команды приведены в таблице 30.

Таблица 30 — Аргументы команды «Write Transducer Channel initiate trigger configuration» («Записать конфигурацию канала преобразователя с инициализацией триггера»)

Поле	Тип данных	Наименование атрибута аргумента	Функция
1	UInt16	InitTrig.destId	«destId» определяет требуемый идентификатор получателя команды
2	UInt16	InitTrig.channelId	«channelId» определяет требуемый канал преобразователя
3	Struct	InitTrig.qosParams	«qosParams» сообщает о требуемом качестве сервисных параметров. Более подробное описание приведено в 9.3.1.3

7.1.3 Команды для рабочего режима преобразователя

Команды класса команд рабочего режима преобразователя должны выполняться, только когда канал преобразователя находится в рабочем режиме. Если какая-либо из таких команд получена, когда канал преобразователя находился в любом другом режиме, то она должна быть проигнорирована, а в регистре состояния-условия канала преобразователя (см. 5.13) должен быть установлен бит «Command rejected» («Отказ от выполнения команды»).

Команды, допустимые для рабочего режима преобразователя, перечислены в таблице 31.

Для получения какой-либо из таких команд каналом преобразователя ИМП должен находиться в рабочем (активном) режиме. Если такая команда была получена, когда ИМП находился не в рабочем

режиме, то команда должна быть проигнорирована, а в регистре состояния-условия ИМП (см. 5.13) должен быть установлен бит «Command rejected» («Отказ от выполнения команды»).

Для всех команд данного класса требуется, чтобы значение номера канала-получателя преобразователя было больше нуля. В случае если номер канала-получателя преобразователя в сообщении равен нулю, то команда должна быть проигнорирована, а в регистре состояния-условия ИМП (см. 5.13) должен быть установлен бит «Command rejected» («Отказ от выполнения команды»).

Таблица 31 — Команды для рабочего режима преобразователя

cmdFunctionId (идентификатор функции команды)	Команда	Класс адреса			Ответ ожидается	Обязательная/ необязательная
		Канал преобразователя	Прокси	Группа/ глобальный		
0	Зарезервировано	—	—	—	—	—
1	Считать сегмент набора данных канала преобразователя	Да	Да	Нет	Да	См. примечание
2	Записать сегмент набора данных канала преобразователя	Да	Да	Нет	Нет	См. примечание
3	Запустить триггер	Да	Да	Да	Нет	Обязательная
4	Прервать триггер	Да	Да	Да	Нет	Необязательная
5—127	Зарезервировано	—	—	—	—	—
128—255	Открыто для изготовителей	—	—	—	—	—
Примечание — Команда «Read Transducer Channel data-set segment» («Считать сегмент набора данных канала преобразователя») является обязательной для датчиков. Команда «Write Transducer Channel data-set segment» («Записать сегмент набора данных канала преобразователя») является обязательной для исполнительных устройств.						

7.1.3.1 Команда «Read Transducer Channel data-set segment» («Считать сегмент набора данных канала преобразователя»)

Наименование атрибута аргумента: DataSetOffset data type UInt32.

Данная команда используется для считывания сегмента набора данных и имеет единственный аргумент, задающий значение смещения от начала набора данных, откуда должно начинаться считывание. Так как максимальный размер байтового массива данных, который может быть обработан на физическом транспортном уровне, меньше, чем максимальный размер набора данных, то смещение набора данных используется для определения допуска к месту внутри набора данных, с которого должно начинаться считывание.

Примечание — Размер большинства наборов данных не превышает размера одного сегмента. Тем не менее стандарт допускает максимальное число повторений данных канала преобразователя, равное 65 356, и модель данных, определяющую 255 байтов в выборке данных, в результате чего максимальный размер набора данных составляет 16 777 216 байтов. В случаях когда набор данных не превышает размеров одного сообщения, значение поля смещения сегмента должно быть равно нулю. Тем не менее большие наборы данных, превышающие размеры одного сообщения, требуют сегментирования набора данных для передачи, а значения смещения набора данных должны быть соответствующими для каждого сегмента.

Для ответного сообщения на команду «Read Transducer Channel data-set segment» («Считать сегмент набора данных канала преобразователя») используется структура ответного сообщения (см. 6.3). Зависимые от содержания ответа байты, отправленные в ответном датаграммном сообщении, должны иметь вид, указанный в таблице 13. Как показано в таблице 32, первое поле содержит значение смещения набора данных, с которого начинается считывание блока данных. В большинстве случаев его значение будет совпадать со значением смещения набора данных в команде «Read Transducer Channel data-set segment» («Считать сегмент набора данных канала преобразователя»). Оставшиеся байты содержат данные, считанные из набора данных. В случае пустого набора данных ответное сообщение

должно содержать только «единицы» для всех значений смещения набора данных и «ноль» байтов данных. Число байтов в ответном сообщении зависит от конструкции канала преобразователя и определяется СПП с помощью аргументов метода API, используемого для ответных сообщений (см. 11.2 или 11.3). Если значение «ReadSensor.Offset» («Смещение набора данных») больше числа байтов в наборе данных, то значение «ReadSensor.Offset» («Смещение набора данных») в ответном сообщении должно быть равно максимальному числу байтов в наборе данных, и ответное сообщение должно содержать ноль байтов данных.

Таблица 32 — Аргументы для ответного сообщения на команду «Read Transducer Channel data-set segment» («Считать сегмент набора данных канала преобразователя»)

Поле	Тип данных	Наименование атрибута аргумента	Функции
1	UInt32	ReadSensorOffset	Смещение набора данных (от 0 до [текущий размер минус 1]) — адрес относительно начала набора данных, с которого должно начинаться считывание данных
2	N×UInt8	ReadSensorData	Блок данных, считанных с датчика

Если данная команда была получена каналом преобразователя, работающим в потоковом режиме передачи данных (см. 5.10.2), то в регистре состояния должен быть установлен бит «Command rejected» («Отказ от выполнения команды»), а команда должна быть проигнорирована.

Если номер канала-получателя преобразователя в байтовом массиве данных равен нулю, то в регистре состояния-условия ИМП (см. 5.13.4) должен быть установлен бит «Command rejected» («Отказ от выполнения команды»), а команда должна быть проигнорирована.

7.1.3.2 Команда «Write Transducer Channel data-set segment» («Записать сегмент данных канала преобразователя»)

Наименование атрибута аргумента: WriteActuator.

Данная команда используется для перезаписи предыдущего содержания набора данных канала преобразователя. Аргументы для команды «Write Transducer Channel data-set segment» («Записать сегмент данных канала преобразователя») должны соответствовать аргументам, приведенным в таблице 33.

Так как максимальный размер байтового массива данных, который может быть обработан на заданном физическом транспортном уровне, меньше, чем максимальный размер набора данных, то смещение набора данных используется для определения допуска к месту внутри набора данных, с которого должна начинаться запись.

Если значение «WriteActuator.Offset» («Смещение набора данных») для исполнительного устройства больше максимальной длины набора данных, то данные должны быть отклонены, а в слове статуса должен быть установлен бит «Command rejected» («Отказ от выполнения команды») (см. 5.13.4).

Примечание — Размер большинства наборов данных не превышает размера одного сегмента. Тем не менее стандарт допускает максимальное число повторений данных канала преобразователя, равное 65 356, и модель данных, определяющую 255 байтов в выборке данных, в результате чего максимальный размер набора данных составляет 16 777 216 байтов. В случаях когда набор данных не превышает размеров одного сообщения, значение поля смещения сегмента должно быть равно нулю. Тем не менее большие наборы данных, превышающие размеры одного сообщения, требуют сегментирования набора данных для передачи, а значения смещения набора данных должны быть соответствующими для каждого сегмента.

Таблица 33 — Поля данных команды «Write Transducer Channel data-set segment» («Записать сегмент данных канала преобразователя»)

Поле	Тип данных	Наименование атрибута аргумента	Функции
1	UInt32	WriteActuator.Offset	Смещение набора данных (от 0 до [текущий размер минус 1]) — это адрес относительно начала набора данных, с которого должна начинаться запись данных
2	N×UInt8	WriteActuator.DataBlock	Блок данных

В случае если данная команда была получена каналом преобразователя, работающим в потоковом режиме передачи данных, то в регистре состояния должен быть установлен бит «Command rejected» («Отказ от выполнения команды») (см. 5.13), а команда должна быть проигнорирована.

Если номер канала-получателя преобразователя в байтовом массиве данных равен нулю, то в регистре состояния-условия ИМП должен быть установлен бит «Command rejected» («Отказ от выполнения команды») (см. 5.13), а команда должна быть проигнорирована.

7.1.3.3 Команда «Trigger» («Запустить триггер»)

Данная команда вызывает последовательность действий, описанную для триггерных команд в 5.11.2.1.

Данная команда не имеет аргументов. Команда может быть адресована каналу преобразователя, прокси-каналу преобразователя, адресной группе или глобальному адресу. В случае глобальной адресации она должна выполняться каждым каналом преобразователя внутри ИМП, для которого функция триггера включена. Если канал преобразователя является датчиком, то он должен действовать согласно триггерной диаграмме состояния датчика (рисунок 7). Если канал преобразователя является исполнительным устройством, то он должен действовать согласно триггерной диаграмме состояния исполнительного устройства (рисунок 8).

7.1.3.4 Команда «Abort trigger» («Прервать триггер»)

Данная команда используется для прерывания работы триггера. Если канал преобразователя является датчиком, то он должен действовать согласно триггерной диаграмме состояния датчика (рисунок 7). Если канал преобразователя является исполнительным устройством, то он должен действовать согласно триггерной диаграмме состояния исполнительного устройства (рисунок 8).

Данная команда не имеет аргументов. Команда может быть адресована каналу преобразователя, прокси-каналу преобразователя, адресной группе или глобальному адресу. В случае глобальной адресации она должна прервать работу триггеров для каждого канала преобразователя с включенной функцией триггера внутри ИМП.

7.1.4 Общие команды для режима ожидания и рабочего режима преобразователя

Команды данного класса могут быть отправлены каналу преобразователя в любое время после его выхода из режима инициализации преобразователя. Допустимые команды для данного класса приведены в таблице 34.

Таблица 34 — Общие команды для режима ожидания и рабочего режима преобразователя

cmdFunctionId	Команда	Класс адреса			Ответ ожидается	Обязательная/ необязательная
		Канал преобразователя	Прокси	Группа/ глобальный		
0	Зарезервировано	—	—	—	—	—
1	Рабочий режим канала преобразователя	Да	Да	Да	Нет	Обязательная
2	Режим ожидания канала преобразователя	Да	Да	Да	Нет	Обязательная
3	Записать состояние триггера канала преобразователя	Да	Да	Да	Нет	Необязательная
4	Считать состояние триггера канала преобразователя	Да	Да	Да	Нет	Обязательная
5	Считать число повторений данных канала преобразователя	Да	Нет	Нет	Да	См. примечание 1
6	Считать предварительное значение счетчика триггера канала преобразователя	Да	Нет	Нет	Да	См. примечание 2
7	Считать привязку к адресной группе	Да	Да	Нет	Да	Обязательная

Окончание таблицы 34

cmdFunctionId	Команда	Класс адреса			Ответ ожидается	Обязательная/ необязательная
		Канал преобразователя	Прокси	Группа/ глобальный		
8	Считать режим выборки	Да	Да	Нет	Да	Необязательная
9	Считать режим передачи данных	Да	Да	Нет	Да	Необязательная
10	Считать состояние буферизации	Да	Да	Нет	Да	Необязательная
11	Считать режим «набор данных закончен»	Да	Да	Нет	Да	Необязательная
12	Считать режим остановки исполнительного устройства	Да	Да	Нет	Да	Необязательная
13	Считать режим сообщения о достижении порогового значения	Да	Да	Нет	Да	Необязательная
14	Считать состояние канала преобразователя с инициализацией триггера	Да	Нет	Нет	Да	Необязательная
15	Считать конфигурацию канала преобразователя с инициализацией триггера	Да	Нет	Нет	Да	Необязательная
16	Очистить устройство	Да	Да	Да	Нет	Необязательная
17—127	Зарезервировано	—	—	—	—	—
128—255	Открыто для изготовителей	—	—	—	—	—

Примечание 1 — Функция является обязательной, если применяется команда «Установить число повторений данных канала преобразователя».

Примечание 2 — Функция является обязательной, если применяется команда «Установить предварительное значение счетчика триггера канала преобразователя».

В случае если какая-либо необязательная команда из данного класса отправлена не поддерживающему ее каналу преобразователя, то в регистре состояния-условия канала преобразователя должен быть установлен бит «Command rejected» («Отказ от выполнения команды») (см. 5.13.4), а команда должна быть проигнорирована.

Для всех команд данного класса требуется значение номера канала-получателя преобразователя, большее нуля. Если номер канала-получателя преобразователя (см. 5.3) в сообщении равен нулю, то команда должна быть проигнорирована, а в регистре состояния-условия канала преобразователя (см. 5.13.4) должен быть установлен бит «Command rejected» («Отказ от выполнения команды»).

Для получения какой-либо из таких команд каналом преобразователя ИМП должен находиться в активном режиме.

7.1.4.1 Команда «Transducer Channel operate» («Рабочий режим канала преобразователя»)

Данная команда вызывает переход канала преобразователя из режима ожидания в рабочий режим. Если канал преобразователя уже находится в рабочем режиме, то команда игнорируется. Описание рабочих режимов канала преобразователя приведено в 5.4.1.

Данная команда не имеет аргументов.

7.1.4.2 Команда «Transducer Channel idle» («Режим ожидания канала преобразователя»)

Данная команда вызывает переход адресованного канала преобразователя в режим ожидания. Если канал преобразователя уже находится в режиме ожидания, то команда игнорируется. Описание рабочих режимов канала преобразователя приведено в 5.4.1.

Данная команда не имеет аргументов.

7.1.4.3 Команда «Write Transducer Channel trigger state» («Записать состояние триггера канала преобразователя»)

Наименование атрибута аргумента: TrigState data type Boolean.

Данная команда используется для включения или отключения функции триггера канала преобразователя, описанной в 5.4.2 и 5.11.

Данная команда имеет единственный аргумент. Если его значение «True» («Истина»), то функция триггера для адресованного канала преобразователя включена. В противоположенном случае функция триггера для адресованного канала преобразователя отключена.

Если такая команда получена каналом преобразователя, не поддерживающим функцию триггера, то в регистре состояния-условия должен быть установлен бит «Command rejected» («Отказ от выполнения команды») (см. 5.13.4), а команда должна быть проигнорирована.

7.1.4.4 Команда «Read Transducer Channel trigger state» («Считать состояние триггера канала преобразователя»)

Наименование атрибута аргумента ответа: TrigState data type Boolean.

Данная команда используется для считывания состояния триггера канала преобразователя, описанного в 5.4.2.

Ответное сообщение на данную команду имеет единственный аргумент. Если его значение «True» («Истина»), то функция триггера для адресованного канала преобразователя включена. В противоположенном случае функция триггера для адресованного канала преобразователя отключена. Если канал преобразователя не поддерживает функцию триггера, то ответ на такую команду должен быть «False» («Ложь»).

7.1.4.5 Команда «Read Transducer Channel data repetition count» («Считать число повторений данных канала преобразователя»)

Наименование атрибута аргумента ответа: RepCount data type UInt16.

Данная команда используется для считывания текущих значений числа повторений данных канала преобразователя (см. 7.1.2.1), приписанных адресованному каналу преобразователя.

Данная команда не имеет аргументов. Тем не менее она вызывает ответное сообщение с единственным 16-битовым (UInt16) аргументом, представляющим собой текущее значение числа повторений данных канала преобразователя.

7.1.4.6 Команда «Read Transducer Channel pre-trigger count» («Считать предварительное значение счетчика триггера канала преобразователя»)

Наименование атрибута аргумента ответа: PreTrigCount data type UInt16.

Данная команда используется для считывания текущих значений предварительного счетчика триггера канала преобразователя, приписанного адресованному каналу преобразователя (см. 7.1.2.2).

Данная команда не имеет аргументов. Тем не менее она вызывает ответное сообщение с единственным 16-битовым (UInt16) аргументом, представляющим собой текущее значение предварительного счетчика триггера канала преобразователя.

7.1.4.7 Команда «Read AddressGroup assignment» («Считать привязку к адресной группе»)

Наименование атрибута аргумента ответа: GrpAssignment data type UInt16.

Данная команда используется для считывания адресной группы (групп), к которой приписан адресованный канал преобразователя (см. 11.3.14).

Данная команда не имеет аргументов. Тем не менее она вызывает ответное сообщение с единственным 16-битовым (UInt16) аргументом, представляющим собой адреса адресной группы или групп, к которой приписан канал преобразователя. Нулевое значение аргумента означает, что канал преобразователя не приписан к какой-либо адресной группе.

7.1.4.8 Команда «Read sampling mode» («Считать режим выборки»)

Наименование атрибута аргумента ответа: SampleMode data type UInt8.

Данная команда позволяет СПП определить текущий рабочий режим выборки канала преобразователя (см. 5.10.1 и 8.5.2.44).

Данная команда не имеет аргументов. Тем не менее она вызывает ответное сообщение с единственным 8-битовым (UInt8) аргументом, представляющим собой текущий режим выборки канала преобразователя. Список допустимых значений для данного аргумента приведен в таблице 26.

7.1.4.9 Команда «Read data transmission mode» («Считать режим передачи данных»)

Наименование атрибута аргумента ответа: XmitMode data type UInt8.

Данная команда позволяет СПП определить текущий рабочий режим передачи данных канала преобразователя (см. 5.10.2 и 8.5.2.49).

Данная команда не имеет аргументов. Тем не менее она вызывает ответное сообщение с единственным 8-битовым (UInt8) аргументом, представляющим собой текущий режим передачи данных канала преобразователя. Список допустимых значений для аргумента ответного сообщения приведен в таблице 7.

7.1.4.10 Команда «Read buffered state» («Считать состояние буферизации»)

Наименование атрибута аргумента ответа: BufferOnOff data type Boolean.

Данная команда не имеет аргументов. Тем не менее она вызывает ответное сообщение с единственным аргументом логического типа. Если значение аргумента «True» («Истина»), буферизация включена. В противоположенном случае буферизация отключена.

7.1.4.11 Команда «Read end-of-data-set operation mode» («Считать режим «набор данных закончен»)

Наименование атрибута аргумента ответа: EndDataSetMode data type UInt8.

Данная команда позволяет СПП определить действие, совершаемое исполнительным устройством канала преобразователя по окончании текущего набора данных, если во время отработки текущего набора данных не был записан новый набор данных.

Данная команда не имеет аргументов. Тем не менее она вызывает ответное сообщение с единственным аргументом. Список допустимых значений для данного аргумента приведен в таблице 27.

7.1.4.12 Команда «Read actuator halt mode» («Считать режим остановки исполнительного устройства»)

Наименование атрибута аргумента ответа: ActuatorHalt data type UInt8.

Данная команда не имеет аргументов. Тем не менее она вызывает ответное сообщение с единственным аргументом. Список допустимых значений для данного аргумента приведен в таблице 28.

7.1.4.13 Команда «Read Edge-to-Report mode» («Считать режим сообщения о достижении порогового значения»)

Наименование атрибута аргумента ответа: EdgeReported data type UInt8.

Данная команда не имеет аргументов. Тем не менее она генерирует ответное сообщение с единственным аргументом. Список допустимых значений для данного аргумента приведен в таблице 29.

7.1.4.14 Команда «Read Transducer Channel initiate trigger state» («Считать состояние канала преобразователя с инициализацией триггера»)

Наименование атрибута аргумента ответа: EventTrig data type Boolean.

Данная команда позволяет СПП определить, установлена ли для датчика событий канала преобразователя функция инициирования триггера после совершения события. Если такую команду получает датчик или исполнительное устройство, то в слове состояния должен быть установлен бит «Command rejected» («Отказ от выполнения команды») (см. 5.13.4).

Данная команда не имеет аргументов. Тем не менее она вызывает ответное сообщение с единственным аргументом. Если его значение «True» («Истина»), то функция инициирования триггера после совершения события для датчика событий включена. В противоположенном случае функция триггера отключена.

7.1.4.15 Команда «Read Transducer Channel initiate trigger configuration» («Считать конфигурацию канала преобразователя с инициализацией триггера»)

Наименование атрибута аргумента ответа: InitTrig data type Boolean.

Данная команда используется для считывания информации, позволяющей каналу преобразователя инициировать триггерную команду.

Данная команда не имеет аргументов. Тем не менее она генерирует ответное сообщение с аргументами, перечисленными в таблице 30. Значения этих аргументов описаны в 11.3.2.

7.1.4.16 Команда «Device clear» («Очистить устройство»)

Данная команда используется для очистки всех входных и выходных буферов адресованных каналов преобразователя. Более подробная информация об очистке устройства представлена на рисунке 7 для триггерных диаграмм состояний датчика и на рисунке 8 — для триггерных диаграмм состояний исполнительного устройства.

Данная команда не имеет аргументов и не вызывает ответа.

7.1.5 Команды для спящего режима ИМП

Нижеследующие команды должны выполняться, только если ИМП находится в спящем режиме. Список допустимых команд данного класса приведен в таблице 35.

Для всех команд данного класса требуется нулевое значение номера канала-получателя преобразователя. Если номер канала-получателя преобразователя в сообщении не равен нулю, команда

должна быть проигнорирована, и в регистре состояния-условия канала преобразователя (см. 5.13.4) должен быть установлен бит «Command rejected» («Отказ от выполнения команды»).

Таблица 35 — Команды для спящего режима

cmdFunctionId	Команда	Класс адреса		Ответ ожидается	Обязательная/ необязательная
		ИМП	Глобальный		
0	Зарезервировано	—	—	—	—
1	Выйти из спящего режима	Да	Нет	Да	Необязательная
2—127	Зарезервировано	—	—	—	—
128—255	Открыто для изготовителей	—	—	—	—

7.1.5.1 Команда «Wake-up» («Выйти из спящего режима»)

Данная команда вызывает переход адресованного ИМП в активный режим.

Данная команда не имеет аргументов. Ответное сообщение на данную команду всегда должно создаваться после завершения операции перехода. Ответное сообщение должно иметь тот же вид, что и ответное сообщение на команду «Read Status-Event Register» («Считать регистр состояния-события») (см. 7.1.1.8), адресованную ИМП.

Примечание — Если такая команда отправлена нескольким ИМП, то соответствующие устройства ИИЭР 1451.X должны иметь возможность обработки ответов от нескольких ИМП без прерывания работы системы.

7.1.6 Команды для активного режима ИМП

Перечисленные ниже команды могут выполняться, только если ИМП находится в активном режиме (см. таблицу 36). Если какая-либо из таких команд получена, когда ИМП находится не в активном режиме, то команда должна быть проигнорирована, и в регистре состояния-условия ИМП (см. 5.13) должен быть установлен бит «Command rejected» («Отказ от выполнения команды»).

Для всех команд данного класса требуется нулевое значение номера канала-получателя преобразователя. Если номер канала-получателя преобразователя в сообщении не равен нулю, то команда должна быть проигнорирована, и в регистре состояния-условия ИМП (см. 5.13.4) должен быть установлен бит «Command rejected» («Отказ от выполнения команды»).

Таблица 36 — Команды для активного режима ИМП

cmdFunctionId	Команда	Класс адреса		Ответ ожидается	Обязательная/ необязательная
		ИМП	Глобальный		
0	Зарезервировано	—	—	—	—
1	Считать версию ИМП	Да	Нет	Да	Обязательная
2	Перейти в спящий режим ИМП	Да	Нет	Нет	Необязательная
3	Сохранить рабочие настройки	Да	Нет	Нет	Обязательная
4	Восстановить рабочие настройки	Да	Нет	Нет	Обязательная
5	Считать версию ИИЭР 1451.0	Да	Нет	Да	Обязательная
6—127	Зарезервировано	—	—	—	—
128—255	Открыто для изготовителей	—	—	—	—

7.1.6.1 Команда «Read TIM version» («Считать версию ИМП»)

Наименование атрибута аргумента ответа: TIMVersion data type UInt16.

Данная команда используется для считывания номера версии ИМП.

Данная команда не имеет аргументов. Тем не менее она вызывает ответное сообщение с единственным аргументом. Значение данного аргумента представляет собой определенный изготовителем

номер версии ИМП. Содержание данного номера выбирается изготовителем, но при этом оно должно быть достаточным, чтобы пользователь мог определить изменения внутри устройства даже без изменения номера модели.

7.1.6.2 Команда «TIM sleep» («Перейти в спящий режим ИМП»)

Данная команда переводит адресованный ИМП в состояние с низким потреблением энергии, в котором ИМП отвечает только на команду «Wake-up» («Выйти из спящего режима»).

Данная команда не имеет аргументов и не вызывает ответа.

7.1.6.3 Команда «Store state» («Сохранить состояние»)

Наименование атрибута аргумента: StorState[N] 0≥N≥255 data type UInt8.

Данная команда используется для того, чтобы запустить процесс сохранения информации о режимах ИМП и каналов преобразователя и необходимое количество дополнительной информации, достаточное для точного восстановления настроек, существовавших на момент получения команды. Существуют два условия, запускающие процесс восстановления состояния:

- получение команды «Recall state» («Восстановить состояние»);
- восстановления после отключения питания.

Примечание — В сохраняемую информацию должны быть включены параметры обработки сигнала, рабочие режимы и т. д.

Данная команда имеет единственный 8-битовый (UInt8) аргумент, который должен назначить нужный режим в случае, если должно быть сохранено несколько режимов. Состояние запуска питания должно быть нулевым. Данная команда не вызывает ответа.

7.1.6.4 Команда «Recall state» («Восстановить состояние»)

Наименование атрибута аргумента: StorState[N] 0≥N≥255 data type UInt8.

Данная команда используется для восстановления настроек ИМП и всех каналов преобразователя в составе ИМП, действующих в момент получения команды «Store state» («Сохранить состояние») и соответствующих аргументу, определяющему номер состояния.

Данная команда имеет единственный 8-битовый (UInt8) аргумент. Данный аргумент должен быть использован для назначения номера состояния (режима), подлежащего восстановлению. Данная команда не требует ответа.

7.1.6.5 Команда «Read the IEEE 1451.0 version» («Считать версию ИИЭР 1451.0»)

Наименование атрибута аргумента ответа: StandardVersion data type UInt8.

Данная команда используется для считывания версии настоящего стандарта, поддерживаемой ИМП.

Данная команда не имеет аргументов. Ответное сообщение на данную команду должно иметь единственный байт, содержащий номер версии стандарта ИИЭР 1451.0 в поле данных, как показано в таблице 37.

Таблица 37 — Нумерация версий ИИЭР 1451.0

Стандартная версия	Стандартная версия ИИЭР 1451.0
0	Зарезервировано для прототипа или прочих нестандартных версий ИИЭР 1451.0
1	Соответствует исходной версии настоящего стандарта
2—127	Зарезервировано для будущих версий настоящего стандарта
128—255	Открыто для изготовителей

7.1.7 Команды для любого режима ИМП

Перечисленные ниже команды могут выполняться, когда ИМП находится в любом режиме. Список команд данного класса приведен в таблице 38.

Для всех команд данного класса требуется нулевое значение номера канала-получателя преобразователя. Если номер канала-получателя преобразователя в сообщении не равен нулю, то команда должна быть проигнорирована, и в регистре состояния-условия канала преобразователя (см. 5.13.4) должен быть установлен бит «Command rejected» («Отказ от выполнения команды»).

Таблица 38 — Команды для любого режима ИМП

cmdFunctionId	Команда	Класс адреса		Ответ ожидается	Обязательная/ необязательная
		ИМП	Глобальный		
0	Зарезервировано	—	—	—	—
1	Перезагрузка	Да	Нет	Нет	Необязательная
2—127	Зарезервировано	—	—	—	—
128—255	Открыто для изготовителей	—	—	—	—

7.1.7.1 Команда «Reset» («Перезагрузка»)

Данная команда используется для перезагрузки ИМП и всех каналов преобразователя, относящихся к данному ИМП.

После запуска команды «Reset» («Перезагрузка») ИМП и все относящиеся к нему каналы преобразователя должны немедленно начать процесс инициализации/начальной загрузки. Если ранее при помощи команды «Store state» («Сохранить состояние») (см. 7.1.6.3) был сохранен какой-либо режим или режимы, то при инициализации восстанавливается «нулевой» режим.

Данная команда не имеет аргументов и не вызывает ответа.

7.2 Команды, определенные изготовителем

Изготовитель может добавлять нестандартные команды, которые необходимо представить пользователю через командную ЭТДП, описанную в 5.5.2.5. За наличие программного обеспечения, необходимого для использования таких команд, отвечает пользователь.

8 Спецификация ЭТДП

В настоящем разделе описано содержание всех ЭТДП, определенных в настоящем стандарте.

8.1 Общий формат для ЭТДП

Все ЭТДП имеют общий формат, приведенный в таблице 39. Первое поле в любой ЭТДП — это длина ЭТДП. Она записана четырехбайтовым целым числом без знака. Следующий блок содержит информационную составляющую ЭТДП. В зависимости от ЭТДП информация может быть представлена в двоичном или текстовом виде. В последнем поле любой ЭТДП представлена контрольная сумма, которая должна использоваться для проверки целостности ЭТДП.

Таблица 39 — Основная структура любой ЭТДП

Поле	Описание	Тип	Число байтов
—	Длина ЭТДП	UInt32	4
От 1 до N	Блок данных	Различный	Переменное
—	Контрольная сумма	UInt16	2

8.1.1 Длина ЭТДП

Тип данных: целое число без знака, используемое для длины поля (UInt32, 4 байта).

Длина ЭТДП представляет собой общее число байтов в блоке данных ЭТДП плюс 2 байта контрольной суммы.

8.1.2 Блок данных

Тип данных: структура, определенная конкретной ЭТДП.

Данная структура содержит информацию, которая хранится в конкретной ЭТДП. Поля, содержащие данную структуру, различны для каждого типа ЭТДП. Во всех ЭТДП, разработанных изготовителем преобразователя, используется структура данных «Type/Length/Value (TLV)» («Тип/длина/значение»). В случае текстовых ЭТДП структура данных TLV используется для обозначения каталога для возможности доступа в различные разделы текстовой части ЭТДП, которая использует язык XML для информационного содержания.

При использовании структуры TLV, как показано в таблице 40, каждая запись сохраняется как «TLV кортеж». Поле «Тип» представляет собой тег в 1 байт, который идентифицирует TLV аналогично функциям тегов HTML или XML. Поле «Длина» определяет число байтов поля «Значения», а поле «Значение» представляет собой фактические данные. Каждая запись может состоять из одной или более структур TLV. Структура или тип данных поля значения определяются в описании ЭТДП в настоящем стандарте (см. таблицу 43 для примера).

Таблица 40 — Определение структуры тип/длина/значение (TLV)

Поле	Описание
Тип	Данный код идентифицирует поле в ЭТДП, которое содержится в поле «Значение». За исключением типов 2 и 3, одно и то же число в поле «Тип» будет иметь разное значение для различных ЭТДП
Длина	Число в данном поле представляет собой число байтов в поле «Значение». Число байтов в поле «Длина» контролируется записью в структуре «TEDS Identification TLV» («TLV для идентификации ЭТДП»)
Значение	Данное поле содержит информацию ЭТДП

8.1.3 Неиспользуемые коды поля «Тип»

В рамках определения каждой ЭТДП некоторые коды поля «Тип» не используются в таких ЭТДП. Данные коды типа перечисляются при определении каждой ЭТДП как «зарезервированные» или как «открытые для изготовителей».

8.1.3.1 Код поля «Тип» «Зарезервированные типы»

Коды поля «Тип», перечисленные как «зарезервированные», зарезервированы организацией Common Functionality и рабочей группой ЭТДП для будущих изменений к стандарту. Они не должны использоваться изготовителями или другими группами.

8.1.3.2 Код поля «Тип» «Открытые для изготовителей»

Коды поля «Тип», перечисленные как «открытые для изготовителей», могут быть использованы изготовителями для реализации функций, которые не определены в стандарте. Если изготовитель хочет реализовать в модуле преобразователя коды типов, которые не описаны в настоящем стандарте, и это устройство работает в системе, которая не распознает определенные поля типов изготовителя, то все описанные в настоящем стандарте функции должны работать нормально, но при этом дополнительные функции изготовителя поддерживаться не будут.

8.1.4 Совместимость со стандартом ИИЭР 1451.2—1997

ЭТДП по стандарту ИИЭР 1451.2—1997 не используют TLV-кортежи. Тем не менее первый байт, следующий за полем «Длина» в мета-ЭТДП, всегда содержит цифру два. Так как первый байт, следующий за полем «Длина» в любой ЭТДП, соответствующей ИИЭР 1451.0, всегда является кодом типа, то код типа «2» резервируется и не должен использоваться. Поскольку мета-ЭТДП является единственной ЭТДП в ИИЭР 1451.2—1997, которая содержит информацию о версии ЭТДП, то при использовании настоящего стандарта необходимо сначала считывать мета-ЭТДП, прежде чем пытаться прочитать какие-либо другие ЭТДП.

8.1.5 Совместимость со стандартом ИИЭР 1451.3—2003

ЭТДП по стандарту ИИЭР 1451.3—2003 не использует TLV-кортежи. Тем не менее первый байт, следующий за полем «Длина» в мета-ЭТДП, всегда содержит цифру один. Так как первый байт, следующий за полем «Длина» в любой ЭТДП, соответствующей ИИЭР 1451.0, всегда является кодом типа, то код типа «1» резервируется и не должен использоваться. Поскольку мета-ЭТДП является единственной ЭТДП в ИИЭР 1451.3—2003, которая содержит информацию о версии ЭТДП, то при использовании настоящего стандарта необходимо сначала считывать мета-ЭТДП, прежде чем пытаться прочитать какие-либо другие ЭТДП.

8.1.6 Контрольная сумма

Тип данных: 16-разрядное целое число без знака (UInt16, 2 байта).

Контрольной суммой должно являться дополнение до единицы суммы (по модулю 2^{16}) всех предыдущих байтов, включая начальное поле «Длина» ЭТДП и весь блок данных ЭТДП. При расчете контрольной суммы исключают само поле «Контрольная сумма».

$$\text{checksum} = 0\text{xFFFF} - \sum_{i=1}^{\text{TotalOctets}-2} \text{TEDSOctet}(i). \qquad (4)$$

Дополнение до единицы суммы N — это $(2^{16} - 1) - N$. Контрольная сумма может быть рассчитана как разница шестнадцатеричного значения 0xFFFF и суммы последовательности с первого байта до байта перед контрольной суммой, как показано в формуле (4). Значение дополнения до единицы для числа можно также вычислить путем обращения цифр данного числа.

Примечание — Вычисления контрольной суммы начинаются с добавления отдельных байтов при сохранении суммы в виде 16-разрядного числа. Если данное 16-разрядное число переполняется (выходит за пределы диапазона), то необходимо игнорировать переполнение и сохранять только нижние 16 битов. Далее следует взять логическое дополнение (дополнение до единицы) результирующего 16-разрядного числа.

8.2 Порядок байтов в числовых полях

Для числовых значений, требующих более 1 байта, первый байт, следующий за полем «Длина» кортежа, должен являться старшим значащим байтом. Последнее поле должно содержать младший значащий байт.

8.3 Поле «TEDSID» («Заголовок для идентификации ЭТДП»)

Тип поля: 3.

Имя поля: TEDSID.

Значение по умолчанию: «Не применяется». Данное поле обязательно во всех ЭТДП.

Идентификатор ЭТДП состоит из четырех полей, представленных в таблице 41, и является стандартным для всех ЭТДП. Данное поле всегда является первым в ЭТДП. Длина кортежа данного поля считается равной одному.

Содержание данного поля следующее:

- номер комплекса стандартов ИИЭР 1451 (0 для настоящего стандарта);
- класс ЭТДП;
- номер версии;
- длина кортежа.

Таблица 41 — Структура идентификатора ЭТДП

Поле	Содержание	Функция
Тип	03	Поле типа для идентификатора ЭТДП
Длина	04	Данное поле всегда устанавливается равным 04, указывая на то, что поле «Значение» содержит 4 байта
Семейство	00	В данном поле указывается стандарт комплекса стандартов ИИЭР 1451, который определяет данную ЭТДП
Класс	См. таблицу 17	В данном поле указывается ЭТДП, к которой осуществляется доступ. Значение представляет собой код доступа к ЭТДП, указанный в таблице 17
Версия	См. таблицу 42	В данном поле указывается версия ЭТДП. Значение представляет собой номер версии, определенный в настоящем стандарте. Нулевое значение в данном поле указывает на то, что ЭТДП не согласована ни с одним из выпущенных стандартов. В таблице 42 перечислены допустимые значения для данного поля
Длина кортежа	Число байтов	В данном поле указывается число байтов в поле «Длина» всех кортежей в ЭТДП, за исключением данного кортежа. Примечание — Для большинства ЭТДП число байтов в поле «Длина» кортежей — это один, что означает, что в поле значения содержится 255 или менее байтов. Однако в некоторых случаях может потребоваться более 8 битов для числа байтов в поле «Значение», поэтому данное поле определяет число байтов в поле «Длина» кортежа. Все кортежи внутри ЭТДП, кроме идентификатора ЭТДП, должны иметь одинаковое число байтов в поле «Длина»

Таблица 42 — Нумерация чисел версии ЭТДП

Версия ЭТДП	Версия стандарта ИИЭР 1451.0
0	Зарезервировано для прототипов или других нестандартных версий ЭТДП
1	Соответствует первоначальной версии настоящего стандарта
2—255	Зарезервировано для будущих версий настоящего стандарта

8.4 Мета-ЭТДП

Мета-ЭТДП является обязательной ЭТДП. Функция мета-ЭТДП заключается в том, что она должна сделать доступной через интерфейс всю информацию, необходимую для получения доступа к любому каналу преобразователя, а также информацию, общую для всех каналов преобразователей.

8.4.1 Доступ

Доступ к мета-ЭТДП осуществляется с помощью команд «Query TEDS» («Запросить ЭТДП»), «Read TEDS segment» («Считать сегмент ЭТДП»), «Write TEDS segment» («Записать сегмент ЭТДП») или «Update TEDS» («Обновить ЭТДП»). Аргумент команды должен определять код доступа к мета-ЭТДП, как определено в таблице 17.

Данная ЭТДП должна быть реализована как ЭТДП только для чтения, чтобы предотвратить внесение изменений в ее поля, так как данные изменения могут привести к непредсказуемому поведению. Если она реализована как ЭТДП только для чтения, то команды «Write TEDS segment» («Записать сегмент ЭТДП») и «Update TEDS» («Обновить ЭТДП») не должны применяться.

8.4.2 Блок данных

Основное содержание блока данных приведено в таблице 43. Подчиненные подпункты объясняют каждое поле данных в этом блоке данных. Поля «Длина» и «Контрольная сумма» технически не являются частью блока данных ЭТДП, но приведены в таблице 43 для более полного описания ЭТДП (см. 8.1).

Таблица 43 — Структура блока данных мета-ЭТДП

Тип поля	Название поля	Описание	Тип данных	Число байтов
—		Длина	UInt32	4
0—2	—	Зарезервировано	—	—
3	TEDSID	Заголовок для идентификации ЭТДП	UInt8	4
4	UUID	Глобальный уникальный идентификатор	UUID	10
5—9	—	Зарезервировано	—	—
Информация, относящаяся ко времени				
10	OholdOff	Рабочее время ожидания	Float32	4
11	SHoldOff	Время ожидания при медленном доступе	Float32	4
12	TestTime	Время самодиагностики	Float32	4
Число реализованных каналов преобразователя				
13	MaxChan	Число реализованных каналов преобразователя	UInt16	2
14	CGroup	Информационный субблок контрольной группы	—	—
Типы 20 и 21 определяют одну контрольную группу				
20	GrpType	Тип контрольной группы	UInt8	1
21	MemList	Список членов контрольной группы	Массив UInt16	NTc
15	VGroup	Информационный субблок векторной группы	—	—

Окончание таблицы 43

Тип поля	Название поля	Описание	Тип данных	Число байтов
Типы 20 и 21 определяют одну векторную группу				
20	GrpType	Тип векторной группы	UInt8	1
21	MemList	Список членов векторной группы	Массив UInt16	NTv
16	GeoLoc	Специализированная векторная группа для определения географического положения	—	—
Типы 24, 20 и 21 определяют один набор информации о географическом положении				
24	LocEnum	Нумерация, определяющая порядок предоставления информации о месте нахождения	Int8	1
20	GrpType	Тип векторной группы	Int8	1
21	MemList	Список членов векторной группы	Массив UInt16	NTv
17	Proxies	Субблок, определяющий прокси-сервер канала преобразователя	—	—
Типы 22,23 и 21 определяют один прокси-канал преобразователя				
22	ChanNum	Номер канала преобразователя прокси-канала преобразователя	UInt16	1
23	Organiz	Организация набора данных прокси-канала преобразователя	UInt8	1
21	MemList	Список членов прокси-канала преобразователя	Массив UInt16	NTp
18—19	—	Зарезервировано	—	—
25—127	—	Зарезервировано	—	—
128—255	—	Открыто для разработчиков	—	—
—		Контрольная сумма	UInt16	2

8.4.2.1 Поле «TEDSID» («Заголовок для идентификации ЭТДП»)

Заголовок идентификации ЭТДП должен соответствовать требованиям 8.3.

Данное поле является обязательным. Если данное поле опущено или содержит недопустимые значения, то СПП должен сообщить о фатальной ошибке ЭТДП.

8.4.2.2 Поле «UUID» («Глобальный уникальный идентификатор»)

Тип поля: 4.

Имя поля: UUID.

Тип данных: UUID, 10 байтов.

Данное поле должно присутствовать в мета-ЭТДП каждого ИМП и должно быть уникальным (как показано на рисунке 13). Если данное поле отсутствует, то СПП должен сообщить о фатальной ошибке ЭТДП.

Описание содержимого данного поля представлено в 4.12.

8.4.2.3 Наихудшие значения времени ожидания

В данной ЭТДП обеспечено два значения времени ожидания. Оба значения предназначены для обнаружения не дающих отклика ИМП. Для учета двух классов времени отклика введены два значения времени ожидания. Рабочее время ожидания используется для большинства операций и не достигает двух периодов. Второе значение времени ожидания (время ожидания при медленном доступе) используется для команд, которые, как ожидается, займут больше времени для выполнения, например, команды, которые обеспечивают запись в энергонезависимую память.

Примечание — Звездочкой обозначены элементы, которые могут существовать 0 и более раз в пределах ЭТДП.

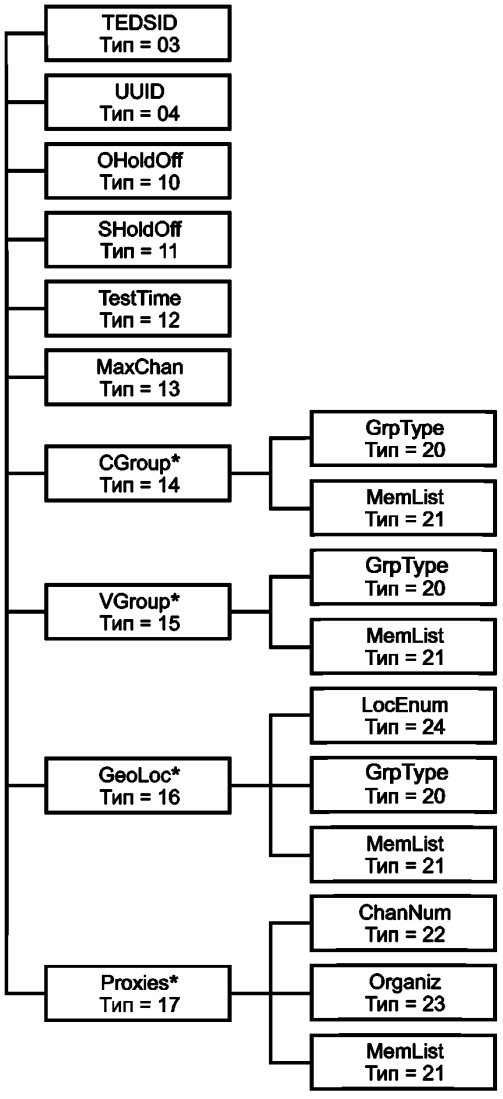


Рисунок 13 — Структура блока данных мета-ЭТДП

8.4.2.4 Поле «OHoldOff» («Рабочее время ожидания (тайм-аут)»)

Тип поля: 10.

Имя поля: OHoldOff.

Тип данных: действительное число одинарной точности (Float32, 4 байта).

Данное поле является обязательным. Если данное поле отсутствует, СПП должен сообщить о фатальной ошибке ЭТДП.

Поле «OHoldOff» рабочего времени ожидания содержит временной интервал в секундах, в течение которого отсутствие ответа после действия или вслед за поступлением команды может интерпретироваться как сбой в ходе операции. Это может быть время, более длительное, чем необходимо для начала отклика на наиболее медленно исполняемые команды, поддерживаемые устройством (например, наихудший случай отклика на команду), или время, в течение которого устройство должно быть готово принять следующую команду, когда не требуется ответа на текущую команду.

В данном поле представлена только часть времени, которая используется в модуле преобразователя. В СПП или главном процессоре, а также при передаче существуют дополнительные задержки, связанные с выполнением, которые должны быть добавлены к этому значению для получения значения временной задержки (тайм-аута).

8.4.2.5 Поле «SHoldOff» («Временная задержка (тайм-аут) при медленном доступе»)

Тип поля: 11.

Имя поля: SHoldOff.

Тип данных: действительное число одинарной точности (Float32, 4 байта).

Поле «SHoldOff» временной задержки при медленном доступе содержит временной интервал в секундах, в течение которого отсутствие ответа после действия или вслед за поступлением команды может интерпретироваться как сбой в ходе операции. Это может быть время более длительное, чем необходимо для начала отклика на наиболее медленно исполняемые команды, поддерживаемые устройством (например, наихудший случай отклика на команду), или время, в течение которого устройство должно быть готово принять следующую команду, когда не требуется ответа на текущую команду. Команды, которые, как ожидается, потребуют более длительного времени задержки, определены при описании отдельных команд (см. раздел 7).

Данное поле не является обязательным. Данное поле может отсутствовать, если единичный тайм-аут не приведет к чрезвычайно медленной работе СПП.

В данном поле представлена только часть времени, которая используется в модуле преобразователя. В СПП или главном процессоре, а также при передаче существуют дополнительные задержки, связанные с выполнением, которые должны быть добавлены к данному значению для получения значения временной задержки (тайм-аута).

8.4.2.6 Поле «TestTime» («Требование к времени самопроверки модуля преобразователя»)

Тип поля: 12.

Имя поля: TestTime.

Данное поле является обязательным. Если данное поле отсутствует, СПП должен сообщить о фатальной ошибке ЭТДП.

Тип данных: действительное число одинарной точности (Float32, 4 байта).

Данное поле содержит максимальное время в секундах, необходимое для выполнения самодиагностики. Если самодиагностика не реализована, то данное поле равно нулю.

8.4.2.7 Поле «MaxChan» («Число реализованных каналов преобразователя»)

Тип поля: 13.

Имя поля: MaxChan.

Тип данных: 16-разрядное целое число без знака (UInt16, 2 байта).

Данное поле является обязательным для всех ИМП. Если данное поле отсутствует, то СПП должен сообщить о фатальной ошибке ЭТДП.

Данное поле содержит число каналов преобразователя, реализованных в данном ИМП. Каналы преобразователя должны быть пронумерованы, начиная с единицы и последовательно увеличиваясь до данного числа. Не разрешается пропускать номера канала преобразователя.

8.4.2.8 Поле «CGroup» («Контрольные группы»)

Тип поля: 14.

Имя поля: CGroup.

Тип данных: 8-разрядное целое число без знака (Uint8, 1 байт).

Данное поле является обязательным для ИМП, содержащих контрольные группы. Если в модуле преобразователя нет контрольных групп, то данное поле опускается.

Контрольные группы определяют преобразователи, которые используются для управления работой других преобразователей, как показано в таблице 44. Определение контрольной группы представляет собой иерархическую структуру со следующими подполями (TLV-кортежами):

- тип группы;
- перечень членов группы.

В таблице 44 в графе «Нумерация» приведено значение, которое помещается в поле для типа группы, чтобы идентифицировать ту конкретную контрольную группу, которая определяется в данный момент. В графе «Функция» определена функция каждого канала преобразователя в группе. Перечень членов группы — это перечень номеров каналов преобразователей для преобразователей, которые выполняют каждую из функций. Цифры в графе «Порядок членов группы» определяют порядок, в котором должны быть перечислены номера каналов преобразователей. Перечень членов группы представляет собой упорядоченный массив, никакие элементы которого не могут быть опущены. Если канал преобразователя не требуется, то номер канала преобразователя для данной функции должен быть равен нулю.

Нумерация 2 может быть использована для маркировки каналов преобразователя встроенного исполнительного устройства, которые могут использоваться для установки фильтра верхних частот, фильтра нижних частот и коэффициента масштаба, связанного с датчиком любого типа.

Нумерация 3 используется для маркировки канала преобразователя встроенного исполнительного устройства, который может использоваться для установления периода выборки (частоты дискретизации) канала преобразователя.

Нумерация 8 используется в том случае, когда требования к одновременной выборке приводят к необходимости введения программируемой задержки, чтобы уравнять задержки всех каналов преобразователя внутри группы.

Нумерации 1 и 4 могут быть использованы для маркировки каналов преобразователя встроенного исполнительного устройства, которые используются для установки датчика событий. Они также маркируют канал преобразователя датчика, который может использоваться для считывания уровня сигнала в аналоговом датчике событий или текущей последовательности входного сигнала для цифрового датчика событий.

Таблица 44 — Нумерация типов контрольных групп

Нумерация	Порядок членов группы	Функция
0		Зарезервировано
1	1	Канал преобразователя аналогового датчика событий
	2	Канал преобразователя аналогового входного датчика, который измеряет значение входного сигнала для члена группы под номером 1. Первый член группы не обеспечивает измерения, а лишь проводит оценку состояния
1	3	Канал преобразователя для контроля и установки верхнего порогового значения встроенного исполнительного устройства
	4	Канал преобразователя для контроля и установки гистерезиса встроенного исполнительного устройства
2	1	Канал преобразователя датчика (любого типа)
	2	Канал преобразователя для контроля и установки фильтра высоких частот встроенного исполнительного устройства
	3	Канал преобразователя для контроля и установки фильтра низких частот встроенного исполнительного устройства
	4	Канал преобразователя для контроля и установки коэффициента масштаба встроенного исполнительного устройства
3	1	Канал преобразователя (любого типа)
	2	Канал преобразователя для контроля и установки интервала выборки встроенного исполнительного устройства
4	1	Канал преобразователя цифрового датчика событий
	2	Канал преобразователя цифрового входного датчика, который измеряет значение входного сигнала для члена группы под номером 1. Первый член группы не обеспечивает измерения, а лишь проводит оценку состояния
	3	Канал преобразователя для контроля и установки образцовой последовательности события встроенного исполнительного устройства
5	1	Канал преобразователя для контроля и установки временного интервала датчика
	2	Номер канала преобразователя, который вызывает фиксацию выходных данных с датчика временного интервала
6	1	Канал преобразователя датчика временного события («TimeInstance»)
	2	Номер канала преобразователя, который вызывает фиксацию выходных данных с датчика временного события («TimeInstance»)
7	1	Номер канала преобразователя датчика событий, используемого для запуска других преобразователей

Окончание таблицы 44

Нумерация	Порядок членов группы	Функция
7	2—N	Оставшиеся N записей предоставляют собой список каналов преобразователя, запускаемых при событии
8	1	Канал преобразователя (любого типа)
	2	Канал преобразователя для контроля и установки временной задержки встроенного исполнительного устройства
9	1	Канал преобразователя датчика расположения
	2	Номер канала преобразователя, который вызывает фиксацию выходных данных с датчика расположения
10—127		Зарезервировано для будущего расширения
128—255		Открыто для изготовителей

8.4.2.9 Поле «GrpType» («Тип группы»)

Тип поля: 20.

Имя поля: GrpType.

Тип данных: 8-разрядное целое число без знака (Uint8, 1 байт).

Данное поле является обязательным для ИМП, реализующих контрольные группы. Данное поле опускается, если в модуле преобразователя нет контрольных групп.

Тип группы определен в графе «Нумерация» таблицы 44 для контрольных групп или в соответствующей графе таблицы 45 для векторных групп.

8.4.2.10 Поле «MemList» («Список членов группы»)

Тип поля: 21.

Имя поля: MemList.

Тип данных: одномерный массив 16-разрядных целых чисел без знака (UInt16, от 2 до 510 байтов).

Данное поле является обязательным для ИМП, реализующих контрольные и векторные группы. В случае если в модуле преобразователя нет контрольных или векторных групп, данное поле опускается.

Тип группы определен в графе «Нумерация» таблицы 44 для контрольных групп или в соответствующей графе таблицы 45 для векторных групп.

Данное поле представляет собой перечень номеров каналов преобразователя, которые предназначены для составления контрольной группы. Они располагаются в порядке, указанном в таблицах 44 или 45.

8.4.2.11 Поле «VGroup» («Векторные группы»)

Тип поля: 15.

Имя поля: VGroup.

Тип данных: 8-разрядное целое число без знака (Uint8, 1 байт).

Данное поле является обязательным для ИМП, реализующих векторные группы. Если в модуле преобразователя нет векторных групп, то данное поле опускается.

Векторные группы определяют взаимосвязь между наборами данных в модуле датчика, как показано в таблице 45. По определению векторная группа представляет собой иерархическую структуру со следующими подполями:

- тип группы;
- список членов группы.

В таблице 45 в графе «Нумерация» приведено значение, которое помещается в поле «Тип группы», чтобы идентифицировать ту конкретную векторную группу, которая определяется в данный момент. В графе «Функция» определена функция каждого канала преобразователя в группе. Список членов группы — это перечень номеров каналов преобразователей для преобразователей, которые выполняют каждую из функций. Цифры в графе «Порядок членов группы» определяют порядок, в котором должны быть перечислены номера каналов преобразователей. Список членов группы представляет собой упорядоченный массив, ни один элемент которого не может быть опущен. Если канал преобразователя не требуется, то номер канала преобразователя для данной функции должен быть равен нулю.

8.4.2.12 Поле «GrpType» («Тип группы»)

Данное поле аналогично полю, описанному в 8.4.2.9.

Данное поле является обязательным для ИМП, реализующих векторные группы. Если в модуле преобразователя нет векторных групп, то данное поле опускается.

8.4.2.13 Поле «MemList» («Список членов группы»)

Структура данного поля аналогична структуре, описанной в 8.4.2.10.

Данное поле является обязательным для ИМП, реализующих векторные группы. Если в модуле преобразователя нет векторных групп, то данное поле опускается.

Данное поле представляет собой перечень номеров каналов преобразователя, которые составляют векторную группу. Они следуют в порядке, указанном в таблице 45.

Таблица 45 — Нумерация типов векторных групп

Нумерация	Порядок членов группы	Функция
0	—	Произвольное отношение
1	1 2 3	Компонента x пространственного вектора правосторонней прямоугольной системы координат Компонента y пространственного вектора правосторонней прямоугольной системы координат Компонента z пространственного вектора правосторонней прямоугольной системы координат
2	1 2 3	Компонента ρ пространственного вектора правосторонней цилиндрической системы координат Компонента φ пространственного вектора правосторонней цилиндрической системы координат Компонента z пространственного вектора правосторонней цилиндрической системы координат
3	1 2 3	Компонента r пространственного вектора правосторонней сферической системы координат Компонента θ пространственного вектора правосторонней сферической системы координат Компонента φ пространственного вектора правосторонней сферической системы координат
4	1 2 3	Широта географической системы координат Долгота географической системы координат Высота географической системы координат
5	1 2	Синфазная компонента двумерного вектора Квадратурная компонента двумерного вектора
6	1 2 3	Красная составляющая вектора цветового пространства Зеленая составляющая вектора цветового пространства Голубая составляющая вектора цветового пространства
7	1 2	Действительная часть комплексного числа Мнимая часть комплексного числа
8—127	—	Зарезервировано для будущего расширения
128—255	—	Открыто для изготовителей

8.4.2.14 Поле «GeoLoc» («Группа географического места нахождения»)

Тип поля: 16.

Имя поля: GeoLoc.

Данное поле обязательно для ИМП, реализующих информацию о динамическом месте нахождения, как указано в таблице 46. В случае если данное поле опущено, СПП должен считать, что данный ИМП не предоставляет информацию о месте нахождения.

Группа географического места нахождения представляет собой иерархическую структуру со следующими подполями (TLV-кортежами):

- нумерация места нахождения;

- тип группы;
- список членов группы.

Данное поле реализует специализированный тип векторной группы, который используется для обеспечения информации о динамическом географическом месте нахождения.

8.4.2.15 Поле «LocEnum» («Нумерация места нахождения»)

Тип поля: 24.

Имя поля: LocEnum.

Тип данных: 8-разрядное целое число без знака (Uint8, 1 байт).

Данное поле является необязательным. Если данное поле опущено, то СПП должен считать, что данный ИМП не предоставляет информацию о месте нахождения.

Значения для ключа типа канала преобразователя приведены в таблице 46.

Таблица 46 — Нумерация типов географического места нахождения

Величина	Значение
0	Данный ИМП не предоставляет информацию о месте нахождения
1	Информация о статическом географическом месте нахождения предоставляется при помощи ЭТДП географического места нахождения
2	Предоставляется информация о динамическом географическом месте нахождения
3	Предоставляется информация о динамическом географическом месте нахождения относительно места нахождения, определенного в ЭТДП географического места нахождения
4—255	Зарезервировано для будущего расширения

8.4.2.16 Поле «GrpType» («Тип группы»)

Данное поле аналогично полю, описанному в 8.4.2.12, за исключением того, что тип группы должен быть ограничен типами от 1 до 4.

Данное поле является обязательным, если нумерация места нахождения принимает значения 2 или 3. В случае если ИМП не предоставляет информацию о месте нахождения, данное поле опускается.

8.4.2.17 Поле «MemList» («Список членов группы»)

Структура данного поля аналогична структуре, описанной в 8.4.2.10.

Данное поле является обязательным, если нумерация места нахождения принимает значения 2 или 3. В случае если ИМП не предоставляет информацию о месте нахождения, данное поле опускается.

Данный перечень номеров каналов преобразователей составляет векторную группу. Номера располагаются в порядке, который указан в таблице 45 для значений нумерации с 1 по 4.

8.4.2.18 Поле «Proxies» («Прокси-серверы каналов преобразователей»)

Тип поля: 17.

Имя поля: Proxies.

Данное поле является обязательным для ИМП, реализующих прокси-серверы каналов преобразователей. Данное поле опускается в случае, если в модуле преобразователя нет прокси-серверов каналов преобразователей.

Примечание — Если данное поле опущено, а в ИМП имеется один или несколько прокси-серверов, то каналы преобразователей обнаруживаются без ЭТДП канала преобразователя, что является фатальной ошибкой.

Прокси-сервер каналов преобразователя является искусственной конструкцией, которая используется для объединения выходных данных нескольких датчиков или входных данных нескольких исполнительных устройств в единую структуру. Прокси-сервер каналов преобразователя имеет номер канала преобразователя и может быть запущен, считан или записан, но он не обладает другими характеристиками канала преобразователя. Прокси-серверы каналов преобразователя не имеют ЭТДП канала преобразователя, ЭТДП калибровки, ЭТДП частотной характеристики или ЭТДП передаточной функции. Тем не менее с прокси-сервером могут быть связаны текстовые ЭТДП, ЭТДП для специальных приложений конечного пользователя или ЭТДП с именем преобразователя, заданным пользователем.

TLV-кортеж прокси-сервера каналов преобразователя состоит из трех подполей:

- номер канала преобразователя прокси-сервера каналов преобразователя;
- организация набора данных прокси-сервера каналов преобразователя;
- список членов прокси-сервера каналов преобразователя.

8.4.2.19 Подполе «ChanNum» («Номер канала преобразователя прокси-сервера каналов преобразователя»)

Тип поля: 22.

Имя поля: ChanNum.

Тип данных: 16-разрядное целое число без знака (UInt16, 2 байта).

Данное поле является обязательным для ИМП, реализующих прокси-серверы каналов преобразователя. Данное поле опускается, если в модуле преобразователя нет прокси-серверов каналов преобразователей.

Данное поле содержит канал преобразователя, который будет использоваться при обращении к данному прокси-серверу. Он может быть использован для считывания данных из прокси-сервера датчика, для записи данных на прокси-сервер исполнительного устройства или для запуска всех преобразователей данного прокси-сервера.

8.4.2.20 Подполе «Organiz» («Организация набора данных прокси-сервера канала преобразователя»)

Тип поля: 23.

Имя поля: Organiz.

Тип данных: 8-разрядное целое число без знака (Uint8, 1 байт).

Данное поле является обязательным для ИМП, реализующих прокси-серверы каналов преобразователя. Данное поле опускается, если в модуле преобразователя нет прокси-серверов каналов преобразователей.

Как показано в таблице 47, существует два метода комбинирования наборов данных для/от нескольких преобразователей. В случае если наборы данных от различных преобразователей содержат разное число слов, должен использоваться метод «блоков». Комбинирование наборов данных методом «блоков» изображено графически в левой части рисунка 6. Комбинирование наборов данных методом «чередования» изображено в правой части рисунка 6. Первая запись в комбинированном наборе данных, «Отсчет от преобразователя X», присутствует только в методе «чередования» 2 и, как ожидается, является некоторым типом временного тега, который позволяет СПП определить, когда был получен первый отсчет в наборе данных для некоторых типов датчиков. Время получения остальных отсчетов, как правило, может быть определено на основе характеристик датчиков, если известно время получения первого отсчета. Запись «Образец данных от преобразователя X» опускается, если используется метод «чередования» 1. Необходимость в возможности установления временной метки существует и при методе «блоков», однако при этом не требуется никаких специальных характеристик для реализации таких меток.

Таблица 47 — Нумерация комбинированных наборов данных

Нумерация	Значение
0	Метод «блоков»
1	Метод «чередования» 1
2	Метод «чередования» 2: аналогичен методу «чередования» 1 за исключением того, что перечень чередующихся каналов преобразователя должен предваряться набором данных из одной выборки от другого канала преобразователя или прокси-сервера. Такой прием обычно используется для установления временной метки, указывающей время получения первого отсчета в комбинированном наборе данных
3—255	Зарезервировано

8.4.2.21 Подполе «Поле «MemList» («Список членов прокси-сервера каналов преобразователя»)

Структура данного поля аналогична структуре, описанной в 8.4.2.10.

Данный перечень номеров каналов преобразователей составляет прокси-сервер. Номера каналов располагаются в порядке появления их данных в прокси-сервере. Если используется метод «блоков», то номер канала для первого блока данных будет идти первым, номер канала для второго блока данных будет идти вторым и так далее вплоть до данных блока N. Если используется один из методов «чередования», то первой записью в перечне будет номер канала для X канала преобразователя, если это требуется, затем будет следовать номер канала, который на рисунке 6 изображен как преобразователь 1, и так далее вплоть до номера канала N.

8.5 ЭТДП канала преобразователя

Данная ЭТДП является обязательной. Функция ЭТДП канала преобразователя должна заключаться в том, чтобы сделать доступной в интерфейсе всю информацию о канале преобразователя, к которому в текущий момент идет обращение, для обеспечения его надлежащей работы.

8.5.1 Доступ

Доступ к ЭТДП канала преобразователя осуществляется с использованием команд «Query TEDS» («Запросить ЭТДП»), «Read TEDS segment» («Считать сегмент ЭТДП»), «Write TEDS segment» («Записать сегмент ЭТДП») или «Update TEDS» («Обновить ЭТДП»). Аргумент команды должен определять код доступа к ЭТДП канала преобразователя, как показано в таблице 17.

Данная ЭТДП может быть реализована как ЭТДП только для чтения для предотвращения изменений, которые вносятся в поле и могут привести к непредсказуемому поведению. В случае если ЭТДП реализована как ЭТДП только для чтения, то команды «Write TEDS segment» («Записать сегмент ЭТДП») или «Update TEDS» («Обновить ЭТДП») канала преобразователя не должны применяться.

8.5.2 Блок данных

В таблице 48 и на рисунке 14 приведена информация, которая должна содержаться в данной ЭТДП. В последующих подпунктах содержится объяснение каждого поля в структуре.

8.5.2.1 Поле «TEDSID» («Идентификатор ЭТДП»)

Идентификатор ЭТДП должен соответствовать структурам, определенным в 8.3.

Данное поле является обязательным. В случае если данное поле опускается или содержит недопустимые значения, СПП должен сообщить о фатальной ошибке ЭТДП.

8.5.2.2 Поле «CalKey» («Ключ калибровки»)

Тип поля: 10.

Имя поля: CalKey.

Тип данных: 8-разрядное целое число без знака (Uint8, 1 байт).

Данное поле является обязательным полем. Если данное поле опускается или содержит недопустимые значения, то СПП должен сообщить о фатальной ошибке ЭТДП.

Данное поле содержит нумерацию, которая показывает калибровочные возможности данного канала преобразователя. В таблице 49 приведен список нумерованных величин и их значений. В графе «Имя» определены идентификаторы, символизирующие нумерованные величины, и данные имена используются в остальной части настоящего стандарта.

8.5.2.3 Коррекции системы

В случае если коррекция осуществляется в СПП, главном процессоре или любом другом месте системы, то должны использоваться нумерации ключей калибровки CAL_SUPPLIED или CAL_CUSTOM.

8.5.2.4 Коррекции модуля преобразователя

При выполнении коррекции в ИМП с использованием одного из методов, указанных в 8.6.1.1, и информации, хранящейся в ЭТДП калибровки (см. 8.6.3), должны использоваться нумерации ключей калибровки TIM_CAL_SUPPLIED и TIM_CAL_SELF. В случае если метод коррекции не является ни одним из методов, описанных в 8.6.1.1, используется TIM_CAL_CUSTOM.

8.5.2.5 Поле «ChanType» («Ключ типа канала преобразователя»)

Тип поля: 11.

Имя поля: ChanType.

Тип данных: 8-разрядное целое число без знака (Uint8, 1 байт).

Данное поле является обязательным полем. Если данное поле опускается или содержит недопустимые значения, то СПП должен сообщить о фатальной ошибке ЭТДП.

Таблица 48 — Структура блока данных ЭТДП канала преобразователя

Поле	Имя поля	Описание	Тип	Число байтов
—		Длина ЭТДП	UInt32	4
0—2	—	Зарезервировано	—	
3	TEDSID	Идентификация ЭТДП	UInt8	4

Продолжение таблицы 48

Поле	Имя поля	Описание	Тип	Число байтов
4—9	—	Зарезервировано		
Информация, относящаяся к каналу преобразователя				
10	CalKey	Ключ калибровки	UInt8	1
11	ChanType	Ключ типа канала преобразователя	UInt8	1
12	PhyUnits	Физические единицы измерения	UNITS	11
50	UnitType	Нумерация интерпретации физических единиц измерения	UInt8	1
51	Radians	Показатель степени для радиан	UInt8	1
52	SterRad	Показатель степени для стерадиан	UInt8	1
53	Meters	Показатель степени для метров	UInt8	1
54	Kilogram	Показатель степени для килограммов	UInt8	1
55	Seconds	Показатель степени для секунд	UInt8	1
56	Amperes	Показатель степени для ампер	UInt8	1
57	Kelvins	Показатель степени для кельвинов	UInt8	1
58	Moles	Показатель степени для молей	UInt8	1
59	Candelas	Показатель степени для кандел	UInt8	1
60	UnitsExt	Код доступа к ЭТДП с расширенным набором единиц измерения	UInt8	1
13	LowLimit	Проектная рабочая нижняя граница диапазона	Float32	4
14	HiLimit	Проектная рабочая верхняя граница диапазона	Float32	4
15	OError	Погрешность при наихудших условиях	Float32	4
16	SelfTest	Ключ самодиагностики	UInt8	1
17	MRange	Возможность работы в нескольких диапазонах	UInt8	1
Информация, относящаяся к преобразователю данных				
18	Sample		—	—
40	DatModel	Модель данных	UInt8	1
41	ModLenth	Длина модели данных	UInt8	1
42	SigBits	Старшие биты модели	UInt16	2
19	DataSet			
43	Repeats	Максимальное повторение данных	UInt16	2
44	SOrigin	Начало отсчета серии	Float32	4
45	StepSize	Шаг дискретизации серии	Float32	4
46	SUnits	Единицы измерения серии	UNITS	11
47	PreTrigg	Максимальное число выборок с предварительным триггером	UInt16	2

Окончание таблицы 48

Поле	Имя поля	Описание	Тип	Число байтов
Информация, относящаяся ко времени				
20	UpdateT	Время обновления канала преобразователя (tu)	Float32	4
21	WSetupT	Время подготовки к записи канала преобразователя (tws)	Float32	4
22	RSetupT	Время подготовки к считыванию канала преобразователя (trs)	Float32	4
23	SPeriod	Период дискретизации канала преобразователя (tsp)	Float32	4
24	WarmUpT	Время готовности канала преобразователя	Float32	4
25	RDelayT	Время задержки считывания канала преобразователя	Float32	4
26	TestTime	Требование ко времени самодиагностики канала преобразователя	Float32	4
Информация о времени отсчета				
27	TimeSrc	Источник времени отсчета	UInt8	1
28	InPropDl	Входящая задержка распространения через устройства передачи данных	Float32	4
29	OutPropD	Исходящая задержка распространения через устройства передачи данных	Float32	4
30	TSError	Погрешность задержки триггер-отсчет	Float32	4
Атрибуты				
31	Sampling	Атрибут выборки данных	UInt8	1
48	SampMode	Возможность выбора режима выборки данных	UInt8	1
49	SDefault	Режим выборки данных по умолчанию	UInt8	1
32	DataXmit	Атрибут передачи данных	UInt8	1
33	Buffered	Буферизованный атрибут	UInt8	1
34	EndOfSet	Атрибут операции «Окончание набора данных»	UInt8	1
35	EdgeRpt	Атрибут «Сообщение о достижении порогового значения»	UInt8	1
36	ActHalt	Атрибут «Остановка исполнительного устройства»	UInt8	1
Чувствительность				
37	Directon	Направление чувствительности	Float32	4
38	DAngles	Углы направления	Два Float32	8
Дополнительные параметры				
39	ESOption	Дополнительные параметры датчика событий	UInt8	1
61—127	—	Зарезервировано	—	—
128—255	—	Открыто для изготовителей	—	—
—		Контрольная сумма	UInt16	2

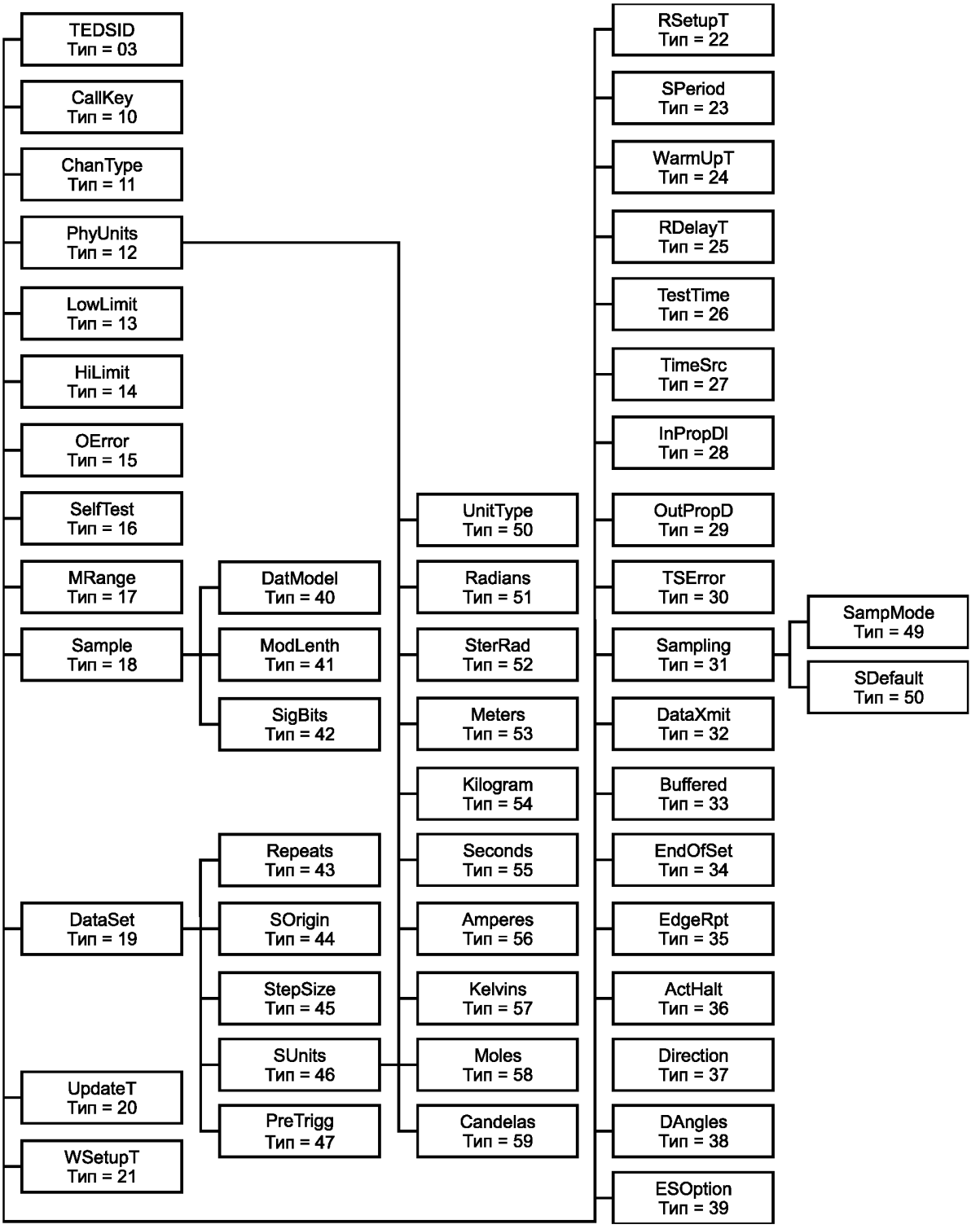


Рисунок 14 — ЭТДП канала преобразователя

Таблица 49 — Нумерация ключей калибровки

Величина	Возможность калибровки	Имя
0	Информация о калибровке не требуется или не предоставляется модулем преобразователя. Это означает, что с данным каналом преобразователя не связана ни одна ЭТДП калибровки. При осуществлении попытки доступа к ЭТДП калибровки длина ЭТДП калибровки должна быть равна нулю	CAL_NONE
1	Информация о калибровке указывается в ЭТДП калибровки, как описано в 8.6. Коррекция осуществляется за пределами ИМП	CAL_SUPPLIED
2	Зарезервировано	
3	Информация о калибровке предоставляется в форме, которая не описана в настоящем стандарте. Коррекция осуществляется за пределами ИМП	CAL_CUSTOM
4	Информация о калибровке указывается в ЭТДП калибровки, как описано в 8.6. Коррекция осуществляется в ИМП	TIM_CAL_ SUPPLIED
5	Информация о калибровке указывается в ЭТДП калибровки, как описано в 8.6, и коррекция осуществляется в ИМП. Информация о калибровке регулируется возможностью самокалибровки	TIM_CAL_SELF
6	Информация о калибровке предоставляется в форме, которая не описывается в настоящем стандарте. Это может быть ЭТДП, заданная изготовителем. Коррекция выполняется в ИМП. При осуществлении попытки доступа к ЭТДП калибровки длина ЭТДП калибровки должна быть равна нулю	TIM_CAL_CUSTOM
7—255	Зарезервировано	—

Значения типов ключей канала преобразователя приведены в таблице 50.

Таблица 50 — Нумерация типов канала преобразователя

Величина	Значение
0	Датчик
1	Исполнительное устройство
2	Датчик событий
3—255	Зарезервировано для будущего расширения

8.5.2.6 Поле «PhyUnits» («Физические единицы измерения»)

Тип поля: 12.

Имя поля: PhyUnits.

Данное поле является обязательным для всех ЭТДП канала преобразователя. Если данное поле опускается, то СПП должен сообщить о фатальной ошибке ЭТДП.

Поле «PhyUnits» («Физические единицы измерения») используется для определения единиц СИ для физической величины, которая измеряется или контролируется в данный момент.

Если показатель степени любого из полей, описанных в 8.5.2.8—8.5.2.16, равен нулю, то данное поле может быть опущено. В этих случаях значение по умолчанию должно быть 128.

Более подробная информация о том, как выбрать значение, которое будет записано в полях «PhyUnits» («Физические единицы измерения»), приведена в 4.11 или приложении J.

8.5.2.7 Поле «UnitType» («Нумерация интерпретации физических единиц измерения»)

Тип поля: 50.

Имя поля: UnitType.

Тип данных: 8-разрядное целое число без знака (Uint8, 1 байт).

Данное поле является обязательным для всех ЭТДП канала преобразователя. Если данное поле опускается, то СПП должен сообщить о фатальной ошибке ЭТДП.

Данное поле описано в 4.11.

8.5.2.8 Поле «Radians» («Показатель степени для радиан»)
Тип поля: 51.
Имя поля: Radians.
Тип данных: 8-разрядное целое число без знака (Uint8, 1 байт).
Данное поле описано в 4.11.
8.5.2.9 Поле «Steradians» («Показатель степени для стерадиан»)
Тип поля: 52.
Имя поля: Steradians.
Тип данных: 8-разрядное целое число без знака (Uint8, 1 байт).
Данное поле описано в 4.11.
8.5.2.10 Поле «Meters» («Показатель степени для метров»)
Тип поля: 53.
Имя поля: Meters.
Тип данных: 8-разрядное целое число без знака (Uint8, 1 байт).
Данное поле описано в 4.11.
8.5.2.11 Поле «Kilograms» («Показатель степени для килограммов»)
Тип поля: 54.
Имя поля: Kilograms.
Тип данных: 8-разрядное целое число без знака (Uint8, 1 байт).
Данное поле описано в 4.11.
8.5.2.12 Поле «Seconds» («Показатель степени для секунд»)
Тип поля: 55.
Имя поля: Seconds.
Тип данных: 8-разрядное целое число без знака (Uint8, 1 байт).
Данное поле описано в 4.11.
8.5.2.13 Поле «Amperes» («Показатель степени для ампер»)
Тип поля: 56.
Имя поля: Amperes.
Тип данных: 8-разрядное целое число без знака (Uint8, 1 байт).
Данное поле описано в 4.11.
8.5.2.14 Поле «Kelvins» («Показатель степени для кельвинов»)
Тип поля: 57.
Имя поля: Kelvins.
Тип данных: 8-разрядное целое число без знака (Uint8, 1 байт).
Данное поле описано в 4.11.
8.5.2.15 Поле «Moles» («Показатель степени для молей»)
Тип поля: 58.
Имя поля: Moles.
Тип данных: 8-разрядное целое число без знака (Uint8, 1 байт).
Данное поле описано в 4.11.
8.5.2.16 Поле «Candelas» («Показатель степени для кандел»)
Тип поля: 59.
Имя поля: Candelas.
Тип данных: 8-разрядное целое число без знака (Uint8, 1 байт).
Данное поле описано в 4.11.
8.5.2.17 Поле «UnitsExt» («Код доступа к ЭТДП с расширенным набором единиц измерения»)
Тип поля: 60.
Имя поля: UnitsExt.
Тип данных: 8-разрядное целое число без знака (Uint8, 1 байт).

Данное поле является необязательным для ЭТДП канала преобразователя. В случае если данное поле не представлено, СПП будет считать, что в данном канале преобразователя не существует ЭТДП с расширенным набором единиц измерения.

Данное поле предназначено для обеспечения возможности применения текстового расширения к физическим единицам измерения. Данное поле не должно использоваться как заменитель физических единиц измерения. Так как в ЭТДП канала преобразователя может содержаться более одного набора физических единиц, то данное поле должно обеспечивать код доступа ЭТДП к текстовой ЭТДП, обе-

спечивающей данное расширение. В случае если требуется единичная ЭТДП с расширенным набором единиц измерения, следует использовать код доступа ЭТДП, указанный в таблице 17.

8.5.2.18 Поле «LowLimit» («Проектная рабочая нижняя граница диапазона»)

Тип поля: 13.

Имя поля: LowLimit.

Тип данных: действительное число одинарной точности (Float32, 4 байта).

Данное поле является обязательным для всех ЭТДП канала преобразователя. В случае если данное поле опускается, СПП должен сообщить о фатальной ошибке ЭТДП.

Для датчиков данная величина должна представлять самое низкое допустимое значение для данных канала преобразователя, которое канал преобразователя должен обеспечивать согласно заданию на проектирование после применения коррекции. Данное значение должно быть выражено в единицах измерения, определенных в поле «PhyUnits» («Физические единицы измерения») ЭТДП канала преобразователя. В случае если скорректированные данные канала преобразователя находятся ниже данной границы, может иметь место несоответствие техническим характеристикам канала преобразователя, установленным изготовителем.

Примечание — Проектная рабочая нижняя граница диапазона всегда выражается в единицах системы СИ. При использовании ЭТДП калибровки выходные данные процесса коррекции могут быть преобразованы в единицы системы СИ путем применения постоянных преобразования единиц СИ, указанных в ЭТДП калибровки (см. 8.6.1.6).

Для исполнительных устройств данная величина должна представлять самое низкое допустимое значение для данных канала преобразователя, которое канал преобразователя должен принимать согласно заданию на проектирование до применения коррекции. Данное значение должно быть выражено в единицах измерения, определенных в поле «PhyUnits» («Физические единицы измерения») ЭТДП канала преобразователя. Запись скорректированных данных канала преобразователя ниже этой границы может привести к выходу за пределы технических характеристик модуля преобразователя, установленных изготовителем.

Примечание — Для каналов преобразователей, которые используют несколько входных сигналов для формирования одного выходного сигнала, данная граница, которая выражается на основе выходного сигнала, будет в большинстве случаев являться номинальным (паспортным), а не точным значением.

В случаях когда коррекция не применяется, а модель данных каналов преобразователя не является действительным числом одинарной точности, до проведения сравнения необходимо выполнить преобразование данных канала преобразователя в действительное число одинарной точности (или преобразование предельного значения к модели данных канала преобразователя).

Примечание — Например, такое преобразование необходимо, когда ключом калибровки является CAL_NONE, а модель данных является N-байтовым целым числом. Следует отметить, что данное преобразование может ограничить практический диапазон или точность преобразованных данных канала преобразователя.

Если данный параметр не применяется, то ему должно быть присвоено значение NaN (см. 4.5.1).

Примечание — В качестве примера применения, в котором границы диапазона не применяются, можно рассмотреть блок переключателей, смоделированных как N-байтовые данные. В этом случае оба поля границ диапазона должны быть установлены на значение NaN. Тем не менее это не означает, что границы диапазона не применяются к N-байтовым данным. Например, 12-разрядное целое число без выраженных единиц измерения, как, например, необработанный сигнал на выходе аналогово-цифрового преобразователя, будет также смоделировано как N-байтовые данные. В данном случае границы диапазона применимы.

Если поле максимального повторения данных (см. 8.5.2.28) данного канала преобразователя не равно нулю, то значение данного поля должно применяться для всех случаев повторения.

8.5.2.19 Проектная рабочая верхняя граница диапазона

Тип поля: 14.

Имя поля: HiLimit.

Тип данных: действительное число одинарной точности (Float32, 4 байта).

Данное поле является обязательным для всех ЭТДП канала преобразователя. В случае если данное поле опускается, СПП должен сообщить о фатальной ошибке ЭТДП.

Для датчиков данная величина должна представлять самое высокое допустимое значение для данных канала преобразователя, которое канал преобразователя должен обеспечивать согласно заданию

на проектирование после применения коррекции. Оно должно быть выражено в единицах измерения, определенных в поле физических единиц измерения ЭТДП канала преобразователя. В случае если скорректированные данные канала преобразователя находятся выше данной границы, может иметь место несоответствие техническим характеристикам канала преобразователя, установленным изготовителем.

Примечание — Проектная рабочая верхняя граница диапазона всегда выражается в единицах системы СИ. При использовании ЭТДП калибровки выходные данные процесса коррекции могут быть преобразованы в единицы системы СИ путем применения постоянных преобразования единиц СИ, указанных в ЭТДП калибровки (см. 8.6.1.6).

Для исполнительных устройств данная величина должна представлять самое высокое допустимое значение для данных канала преобразователя, которое канал преобразователя должен принимать согласно заданию на проектирование до применения коррекции. Оно должно быть выражено в единицах измерения, определенных в поле физических единиц измерения ЭТДП канала преобразователя. Запись скорректированных данных канала преобразователя выше этой границы может привести к выходу за пределы технических характеристик модуля преобразователя, установленных изготовителем.

Примечание — Для каналов преобразователей, которые используют несколько входных сигналов для формирования одного выходного сигнала, данная граница, которая выражается на основе выходного сигнала, будет в большинстве случаев являться номинальным (паспортным), а не точным значением.

В случаях, когда коррекция не применяется, а модель данных каналов преобразователя не является действительным числом одинарной точности, до проведения сравнения необходимо выполнить преобразование данных канала преобразователя в действительное число одинарной точности (или преобразование предельного значения к модели данных канала преобразователя).

В случае если данный параметр не применяется, ему должно быть присвоено значение NaN (см. 4.5.1).

Если поле «Максимальное повторение данных» (см. 8.5.2.28) данного канала преобразователя не равно нулю, то значение данного поля должно применяться для всех случаев повторения.

8.5.2.20 Поле «OError» («Погрешность при наихудших условиях»)

Тип поля: 15.

Имя поля: OError.

Тип данных: действительное число одинарной точности (Float32, 4 байта).

Данное поле является обязательным для всех ЭТДП канала преобразователя. В случае если данное поле опускается, СПП должен сообщить о нефатальной ошибке ЭТДП.

«Суммарная стандартная погрешность» описана в [B10] (подраздел 2.2 приложения С). Значение данного поля должно быть выражено в единицах, определенных в поле «PhyUnits» («Физические единицы измерения») ЭТДП канала преобразователя.

Данное поле используется для описания погрешности значения выходного сигнала данного канала преобразователя при наихудшей комбинации изменений в окружающей среде и любых других факторов, таких как напряжение блока питания, которые могут изменить выходной сигнал канала преобразователя.

8.5.2.21 Поле «SelfTest» («Ключ самодиагностики»)

Тип поля: 16.

Имя поля: SelfTest.

Тип данных: 8-разрядное целое число без знака (Uint8, 1 байт).

Данное поле является обязательным для всех ЭТДП канала преобразователя. В случае если данное поле опускается, СПП должен сообщить о нефатальной ошибке ЭТДП.

Данное поле определяет возможности самодиагностики канала преобразователя, как показано в таблице 51.

Таблица 51 — Нумерация ключей самодиагностики

Величина	Значение
0	Функция самодиагностики не требуется или не обеспечивается
1	Функция самодиагностики обеспечивается
2—255	Зарезервировано для будущего расширения

8.5.2.22 Поле «MRange» («Возможность работы в нескольких диапазонах»)

Тип поля: 17.

Имя поля: MRange.

Тип данных: 8-разрядное целое число без знака (Uint8, 1 байт).

Данное поле является необязательным для ЭТДП канала преобразователя. Если оно не реализовано, то СПП получает информацию об отсутствии возможности работы данного канала преобразователя в нескольких диапазонах.

Содержание данного поля определяет, может ли какой-либо преобразователь в ИМП работать в разных диапазонах. Если значение данного поля «True» («Истина»), то канал преобразователя имеет возможность работы более чем в одном диапазоне. В противном случае возможность работы в нескольких диапазонах отсутствует. В случае если возможность работы в нескольких диапазонах существует, то требуется наличие командной ЭТДП. Данный тип ЭТДП требуется для определения команд, необходимых для выбора рабочего диапазона.

8.5.2.23 Поле «Sample» («Определение отсчета (выборки)»)

Тип поля: 18.

Имя поля: Sample.

Данное поле является обязательным. Если данное поле опускается, то СПП должен сообщить о фатальной ошибке ЭТДП.

Поле «Sample» («Определение отсчета (выборки)») состоит из трех полей и требуется для всех ЭТДП канала преобразователя.

Подполя, составляющие данный тип, следующие:

- модель данных;
- длина модели данных;
- типовые значащие биты.

8.5.2.24 Поле «DatModel» («Модель данных»)

Тип поля: 40.

Имя поля: DatModel.

Тип данных: 8-разрядное целое число без знака (Uint8, 1 байт).

Данное поле является обязательным. Если данное поле опускается, то СПП должен сообщить о фатальной ошибке ЭТДП.

Данное поле описывает модель данных, которые используются при выдаче команд «Read Transducer Channel data-set segment» («Считать сегмент набора данных канала преобразователя») (см. 7.1.3.1) или «Write Transducer Channel data-set segment» («Записать сегмент набора данных канала преобразователя») (см. 7.1.3.2) для данного канала преобразователя. Нумерация значения представлена в таблице 52.

Существуют два отличия между формами целых чисел (нумерация 0 и 5) и формами дробных чисел (нумерация 3 и 6).

Десятичная запятая, которая отделяет целую и дробную части битов, находится справа от младшего значащего бита для целого N-байтового числа или длинного целого. Она находится справа от старшего значащего бита для N-байтового дробного числа или длинного дробного числа. Выравнивание значащих битов отличается, как указано в поле типовых значащих битов (см. 8.5.2.26).

Таблица 52 — Нумерация моделей данных

Значение	Базовый тип данных	Модель	Ограничение на длину модели данных
0	UInt8	N-байтовое целое число (без знака)	0≤N≤8
1	Float32	Действительное число одинарной точности	N=4
2	double	Действительное число двойной точности	N=8
3	UInt8	N-байтовое дробное число (без знака)	0≤N≤8
4	UInt8	Битовая последовательность	0≤N≤8
5	UInt8	Длинное целое число (без знака)	9≤N≤255

Окончание таблицы 52

Значение	Базовый тип данных	Модель	Ограничение на длину модели данных
6	UInt8	Длинное дробное число (без знака)	9≤N≤255
7	TimeInstance	Время дня	N=8
8—255	—	Зарезервировано для будущего расширения	—

Как ожидается, нумерация 5 и 6 будет использоваться редко. Предполагается, что СПП не будет обрабатывать данные с такой нумерацией, а может передать их необработанными в ответ на запрос этих данных.

N-байтовое дробное число может использоваться для того, чтобы держать коэффициенты полинома (см. 8.6.3.22) в рамках представляемых границ и чтобы избежать переполнения данных при их преобразовании в физические единицы измерения.

Модель данных битовой последовательности используется для совокупностей битов, которые не имеют числового значения. Примером такого типа является положение каждого переключателя в блоке переключателей. Значимость бита определяется пользователем.

Логические данные (такие как {0,1} или {ложь, истина}) должны быть представлены в виде битовой последовательности длиной 1 байт. Способ представления значений «истина» и «ложь» приведен в 4.8.

Модель данных времени дня поддерживает типы данных «TimeInstance» («Момент времени»), определенные в 4.9.2.

8.5.2.25 Поле «ModLenth» («Длина модели данных»)

Тип поля: 41.

Имя поля: ModLenth.

Тип данных: 8-разрядное целое число без знака (Uint8, 1 байт).

Данное поле является обязательным. Это поле может быть опущено в случае следующих типов данных: действительное число одинарной точности, действительное число двойной точности и TimeInstance, поскольку такие типы данных имеют фиксированную длину. Во всех остальных случаях, если данное поле опускается, СПП должен сообщить о фатальной ошибке ЭТДП.

Данное поле содержит число байтов, указанное в поле «DatModel» («Модель данных»). Ограничения на длину модели приведены в таблице 52. Нулевое значение в поле «ModLenth» («Длина модели данных») указывает на существование канала преобразователя, который не производит или не использует данные.

8.5.2.26 Поле «SigBits» («Типовые значащие биты»)

Тип поля: 42.

Имя поля: SigBits.

Тип данных: 16-разрядное целое число без знака (UInt16, 2 байта).

Данное поле является обязательным. Если данное поле опускается, то СПП должен сообщить о фатальной ошибке ЭТДП.

Когда модель данных является N-байтовым целым числом (нумерация 0 или 5) или N-байтовым дробным числом (нумерация 3 или 6), значение данного поля представляет собой число битов, которые являются значимыми.

Например, если данные канала преобразователя поступают из 12-битового аналого-цифрового преобразователя, то:

- нумерация модели данных = 0 (N-байтовое целое число);
- длина модели данных = 2 (число байтов, необходимых для хранения 12 битов);
- типовые значащие биты = 12.

В случае если модель данных является N-байтовым целым числом, N-байтовым дробным числом, длинным целым числом или длинным дробным числом, поле «SigBits» («Типовые значащие биты») не должно превышать восьмикратную длину модели данных.

В случае если модель данных является N-байтовым целым или длинным целым числом, биты данных должны быть выровнены по правому краю в потоке байтов. Поле «SigBits» («Типовые значащие биты») определяет число битов, которые являются значащими. Нулевое значение не допускается.

В случае если модель данных является N-байтовым дробным или длинным дробным числом, биты данных должны быть выровнены по левому краю в потоке байтов. Поле «SigBits» («Типовые значащие биты») определяет число битов, которые являются значащими.

В случае если модель данных является действительным числом одинарной точности или действительным числом двойной точности (нумерация 1 или 2), значение данного поля представляет число битов в преобразователе сигнала каналов преобразователя.

В случае если модель данных является действительным числом одинарной точности, действительным числом двойной точности, битовой последовательностью или временем дня, то поле «SigBits» («Типовые значащие биты») постоянно и может быть проигнорировано.

8.5.2.27 Поле «DataSet» («Определение наборов данных»)

Тип поля: 19.

Имя поля: DataSet.

Поле «DataSet» («Определение наборов данных») является обязательным для всех ЭТДП канала преобразователя.

Данное поле является обязательным[1]. Если данное поле опускается, то поле «Maximum data repetitions» («Максимальное повторение данных») и поле «Maximum pre-trigger samples» («Максимальное число выборок с предварительным триггером») должны быть приняты равными нулю.

Подполя, составляющие данный тип, следующие:

- максимальное повторение данных;
- начало отсчета серии;
- шаг дискретизации серии;
- единицы измерения серии;
- максимальное число выборок с предварительным триггером.

8.5.2.28 Поле «Repeats» («Максимальное повторение данных»)

Тип поля: 43.

Имя поля: Repeats.

Тип данных: 16-разрядное[2] целое число без знака (UInt16, 2 байта).

Данное поле является необязательным. Если данное поле опускается, то максимальное число повторений данных должно быть равно нулю.

Данное поле содержит число (L) повторений значения канала преобразователя, которое может быть произведено или быть получено одиночным триггером после исходного отсчета. Каждое повторение представляет собой дополнительное значение измерения или срабатывания, производимое или потребляемое каналом преобразователя при каждом триггерном событии. Данное значение должно быть отделено от исходного значения, связанного с триггером, вдоль некоторой оси (например, времени) на величину, определенную в ЭТДП канала преобразователя в поле «Шаг дискретизации серии» (см. 8.5.2.30) и поле «Единицы измерения серии» (см. 8.5.2.31) соответственно. В случае если L равно нулю, значения начала отсчета серии, шага дискретизации серии и единиц измерения серии могут быть проигнорированы. Цель данной структуры — позволить каналам преобразователя произвести временной ряд или масс-спектр или создать форму сигнала с применением одиночного триггера.

В процессе работы число значений канала преобразователя, которые могут быть произведены или получены одиночным триггером, может быть установлено на любое значение ниже значения данного поля путем использования команды «Set Transducer Channel data repetition count» («Установить число повторений данных канала преобразователя»), как это определено в 7.1.2.1, в случае если данная команда применима.

При считывании или записи данных с числом повторения данных больше нуля порядок передачи должен осуществляться следующим образом: первым передается нулевой (старший) отсчет (выборка данных), вторым передается первое повторение (следующий старший отсчет) и так далее.

8.5.2.29 Поле «SOrigin» («Начало отсчета серии»)

Тип поля: 44.

Имя поля: SOrigin.

Тип данных: действительное число одинарной точности (Float32, 4 байта).

Данное поле может быть опущено, если поле «Maximum data repetitions» («Максимальное повторение данных») (см. 8.5.2.28) равно нулю. В случае если поле «Maximum data repetitions» («Максимальное повторение данных») отлично от нуля, а данное поле опускается, СПП должен присвоить началу отсчета серии нулевое значение.

---

[1] В оригинале ISO/IEC/IEEE 21450:2010 допущена ошибка. Ошибочно приведено «необязательным».

[2] В оригинале ISO/IEC/IEEE 21450:2010 допущена ошибка. Ошибочно приведено «8-разрядное».

В случае когда число повторений данных больше нуля, начало отсчета серии представляет собой значение независимой переменной, связанной с первой опорной точкой в наборе данных. Значения начала отсчета серии выражаются в единицах, определенных в поле «Единицы измерения серии» в ЭТДП канала преобразователя (см. 8.5.2.31). Если единицами измерения серии являются секунды, то содержанием данного поля является число секунд задержки перед началом получения или применения набора данных.

Например, если единицы измерения серии представляют время, то данное значение является временем задержки между получением триггерного сигнала и получением или применением первого отсчета (выборки).

8.5.2.30 Поле «StepSize» («Шаг дискретизации серии»)

Тип поля: 45.

Имя поля: StepSize.

Тип данных: действительное число одинарной точности (Float32, 4 байта).

Данное поле может быть опущено, если поле «Maximum data repetitions» («Максимальное повторение данных») (см. 8.5.2.28) равно нулю. Если поле «Maximum data repetitions» («Максимальное повторение данных») отлично от нуля, а данное поле опускается, то СПП должен сообщить о фатальной ошибке ЭТДП.

В случае когда число повторений данных больше нуля, шаг дискретизации серии представляет собой минимальный промежуток между значениями независимой переменной, связанной с последовательными членами набора данных. Шаг дискретизации серии выражается в единицах, определенных в поле «Единицы измерения серии» в ЭТДП канала преобразователя (см. 8.5.2.31).

Для каналов преобразователей, реализующих возможность программирования шага дискретизации серии, данное поле должно задавать минимальный интервал, который поддерживает ЭТДП.

Примечание — Для каналов преобразователей, реализующих возможность программировать промежутки между выборками данных, рекомендуется использовать встроенное исполнительное устройство для этой функции. Это позволяет задавать промежуток в физических единицах измерения и назначать ЭТДП калибровки порядок преобразования данных единиц в битовую комбинацию, необходимую для программирования оборудования.

8.5.2.31 Поле «SUnits» («Единицы измерения серии»)

Тип поля: 46.

Имя поля: SUnits.

Тип данных: физические единицы (UNITS, 10 байтов).

Данное поле может быть опущено, если поле «Maximum data repetitions» («Максимальное повторение данных») (см. 8.5.2.28) равно нулю. Если поле «Maximum data repetitions» («Максимальное повторение данных») отлично от нуля, а это поле опускается, то СПП должен сообщить о фатальной ошибке ЭТДП.

Данное поле содержит физические единицы, связанные с полем «Начало отсчета серии» (см. 8.5.2.29) и полем «Шаг дискретизации серии» (см. 8.5.2.30) в ЭТДП канала преобразователя.

8.5.2.32 Поле «PreTrigg» («Максимальное число выборок с предварительным триггером»)

Тип поля: 47.

Имя поля: PreTrigg.

Тип данных: 16-разрядное целое число без знака (UInt16, 2 байта).

Данное поле может быть опущено, если поле «Maximum data repetitions» («Максимальное повторение данных») (см. 8.5.2.28) равно нулю или если канал преобразователя не поддерживает режимы автономной выборки данных с предварительно заданным счетчиком триггера (см. 5.10.1.3). Если данное поле опускается, а атрибут режима выборки показывает, что данный канал преобразователя может работать в режимах автономной выборки с предварительно заданным счетчиком триггера, то должно быть выдано сообщение о нефатальной ошибке ЭТДП и принято нулевое максимальное число выборок с предварительным триггером.

Данное поле содержит число (L) повторений значения канала преобразователя, которые могут быть выбраны и сохранены до триггера при работе в режимах автономной выборки с предварительным триггером. Каждое повторение представляет собой дополнительное значение измерения, производимое каналом преобразователя, которое должно быть отделено от исходного значения вдоль некоторой оси (например, времени) на величину, определенную в ЭТДП канала преобразователя в поле «Шаг дискретизации серии» (см. 8.5.2.30) и в поле «Единицы измерения серии» (см. 8.5.2.31) соответственно. В случае когда L равно нулю, датчик может не поддерживать работу в режиме автономной выборки данных с предварительно заданным счетчиком триггера.

В процессе работы число значений канала преобразователя, которые могут быть получены и сохранены до получения триггерного сигнала, может быть установлено на любое значение ниже значения данного поля путем использования команды «Set Transducer Channel pre-trigger count» («Установить предварительное значение счетчика триггера»), как определено в 7.1.2.2, в случае если данная команда применима.

При считывании или записи данных с числом повторения данных больше нуля порядок передачи должен осуществляться следующим образом: первым передается нулевой (старший) отсчет (выборка данных), вторым передается первое повторение (следующий старший отсчет) и так далее.

8.5.2.33 Поле «UpdateT» («Время обновления канала преобразователя»)

Тип поля: 20.

Имя поля: UpdateT.

Тип данных: действительное число одинарной точности (Float32, 4 байта).

Данное поле является обязательным для всех ЭТДП канала преобразователя. Для каналов преобразователя, работающих в автономном режиме выборки данных без предварительно заданного счетчика триггера (см. 5.10.1.2) или в автономном режиме выборки данных с предварительно заданным счетчиком триггера (см. 5.10.1.3), данное число должно быть равно интервалу выборки. В случае если интервал выборки является программируемым и канал преобразователя работает в любом из автономных режимов выборки данных, данная величина не имеет значения и должна быть установлена на ноль. Если данное поле опущено, то должно быть выдано сообщение о фатальной ошибке ЭТДП.

Данное поле содержит выраженное в секундах максимальное время (t_u) между событием запуска триггера и фиксированием первой выборки в наборе данных для данного канала преобразователя. Данный промежуток времени должен быть рассчитан с учетом того, что число повторений данных равняется значению, указанному в поле «Maximum data repetitions» («Максимальное повторение данных») (см. 8.5.2.28). Данный параметр позволяет при необходимости процессорам СПП определить значения времени ожидания.

Для датчиков, работающих в автономном режиме выборки данных, данный параметр применяется только в том случае, если преобразователь находится в рабочем режиме.

8.5.2.34 Поле «WSetupT» («Время подготовки к записи канала преобразователя»)

Тип поля: 21.

Имя поля: WSetupT.

Тип данных: действительное число одинарной точности (Float32, 4 байта).

Данное поле является обязательным для всех каналов преобразователя исполнительных устройств. В случае если данное поле опускается для исполнительного устройства, должно быть выдано сообщение о фатальной ошибке ЭТДП. Данное поле может быть опущено для датчиков или датчиков событий.

Данное поле содержит выраженное в секундах минимальное время (t_{ws}) между окончанием цикла записи данных и применением триггера. Для большинства каналов преобразователей данная характеристика связана со временем подготовки электронной части устройства. Для более сложных каналов преобразователя, особенно если они используют микропроцессор, может потребоваться дополнительное время, определяемое данной постоянной.

8.5.2.35 Поле «RSetupT» («Время подготовки к считыванию канала преобразователя»)

Тип поля: 22.

Имя поля: RSetupT.

Тип данных: действительное число одинарной точности (Float32, 4 байта).

Данное поле является обязательным для всех каналов преобразователя датчиков. В случае если данное поле опущено для датчика, то должно быть выдано сообщение о фатальной ошибке ЭТДП. Данное поле может быть опущено для исполнительных устройств или датчиков событий.

Данное поле содержит выраженное в секундах максимальное время (t_{rs}) между временем получения модулем датчика триггерного сигнала и временем, когда данные становятся доступными для считывания. Для большинства устройств данная характеристика связана с временем подготовки электроники канала преобразователя. Для более сложных каналов преобразователя, особенно если они используют микропроцессор, может потребоваться дополнительное время, определяемое данной постоянной.

В случае если значение времени подготовки к считыванию канала преобразователя равно нулю, канал преобразователя может быть считан в любое время после получения триггерного сигнала.

8.5.2.36 Поле «SPeriod» («Период дискретизации канала преобразователя»)

Тип поля: 23.

Имя поля: SPeriod.

Тип данных: действительное число одинарной точности (Float32, 4 байта).

Данное поле является обязательным для всех каналов преобразователя. В случае если данное поле опущено, должно быть выдано сообщение о фатальной ошибке ЭТДП. Для многих встроенных каналов преобразователя период дискретизации не имеет значения и может быть установлен на ноль, чтобы указать, что он является несущественным.

Период дискретизации канала преобразователя (t_{sp}) — это выраженный в секундах минимальный период дискретизации для канала преобразователя, свободного от считывания и записи (поскольку отсутствует требование считывания или записи по каждому триггеру).

Для датчика и исполнительного устройства канала преобразователя это время обычно ограничивается временем аналого-цифрового и цифро-аналогового преобразований, скоростью процессора модуля преобразователя и т. д. В более сложных каналах преобразователя оно может отражать время канала преобразователя или время получения выборки данных так, как, например, датчик уровня pH, который по каждому сигналу триггера делает забор новой выборки, используя насос. Если применяется считывание или запись, то фактические частоты дискретизации будут ограничены временем подготовки и временем передачи данных, зависящих от типа канала преобразователя.

Для каналов преобразователей с включенным атрибутом автономного режима выборки данных данный параметр должен представлять собой время выборки данных, связанное с технической реализацией модуля преобразователя.

Примечание — Для каналов преобразователей, реализующих возможность программирования интервала между выборками, рекомендуется использовать встроенное исполнительное устройство для данной функции. Это позволяет указать интервал с использованием физических единиц измерения, а также применять ЭТДП калибровки для определения того, как преобразовать данные в битовую последовательность, необходимую для программирования оборудования.

Для датчиков событий данный параметр должен представлять собой минимальное время разрешения события (минимальное время между событиями, которое можно определить).

Период дискретизации канала преобразования должен быть выражен в секундах.

8.5.2.37 Поле «WarmUpT» («Время готовности канала преобразователя»)

Тип поля: 24.

Имя поля: WarmUpT.

Тип данных: действительное число одинарной точности (Float32, 4 байта).

Данное поле является обязательным. В случае если данное поле опущено, СПП должен сообщить о нефатальной ошибке ЭТДП.

Данное поле содержит выраженный в секундах период времени, за который канал преобразователя стабилизирует свои характеристики в рамках заранее определенных допусков, указанных в поле «Погрешность при наихудших условиях» (см. 8.5.2.20[1)]), после включения питания канала преобразователя.

8.5.2.38 Поле «RDelayT» («Время задержки считывания канала преобразователя»)

Тип поля: 25.

Имя поля: RDelayT.

Тип данных: действительное число одинарной точности (Float32, 4 байта).

Данное поле является обязательным для всех датчиков каналов преобразователя. Если данное поле опущено для одного из датчиков, то должно быть выдано сообщение о фатальной ошибке ЭТДП. Данное поле может быть опущено для исполнительных устройств и датчиков событий.

Данное поле содержит выраженное в секундах максимальное время задержки (t_{ch}) между получением команды «Read Transducer Channel data-set segment» («Считать сегмент набора данных канала преобразователя») (см. 7.1.3.1) и началом передачи цикла данных.

8.5.2.39 Поле «TestTime» («Требование ко времени самодиагностики канала преобразователя»)

Тип поля: 26.

Имя поля: TestTime.

Тип данных: действительное число одинарной точности (Float32, 4 байта).

[1)] В оригинале ISO/IEC/IEEE 21450:2010 допущена ошибка. Ошибочно приведена ссылка на 8.5.2.7.

Данное поле является обязательным для всех каналов преобразователя, реализующих возможность самодиагностики, которая определяется ключом самодиагностики (см. 8.5.2.21). Данное поле может быть опущено в случае, если ключ самодиагностики сигнализирует, что возможность самодиагностики не реализована. В случае если ключ самодиагностики указывает, что возможность самодиагностики существует, а данное поле опущено, СПП должен сообщить о нефатальной ошибке ЭТДП. СПП может получить информацию, является ли время самодиагностики обязательным или нет, посредством доступа к возможности самодиагностики.

Данное поле содержит выраженное в секундах максимальное время, необходимое для выполнения самодиагностики.

8.5.2.40 Поле «TimeSrc» («Источник времени выборки данных»)

Тип поля: 27.

Имя поля: TimeSrc.

Тип данных: 8-разрядное целое число без знака (Uint8, 1 байт).

Данное поле не является обязательным и может быть опущено. В случае если данное поле опущено, должна быть рассмотрена опция «NoHelp».

Возможные значения для данного поля приведены в таблице 53.

Значение «Incoming» («Входящие») требует внесения в поле ЭТДП канала преобразователя значения входящей задержки распространения через логику передачи данных (см. 8.5.2.41). Значение «Outgoing» («Исходящие») требует внесения в поле ЭТДП канала преобразователя значения исходящей задержки распространения через логику передачи данных (см. 8.5.2.42).

Работа со значениями «MInterval» и «SInterval» требует наличия встроенного датчика для предоставления такой же информации, что и информация поля ЭТДП для «Incoming» («Входящие)». Для значения «ToDSense» модуль преобразователя предоставляет информацию о времени дня, когда была получена выборка. При работе со значениями «MInterval» и «SInterval» или «ToDSense» требуется назначение контрольной группы в мета-ЭТДП для идентификации датчика временного интервала или датчика временного события («TimeInstance»), который должен использоваться с данным каналом преобразователя.

Таблица 53 — Нумерация источников времени выборки данных

Нумерация	Имя	Функция
0	NoHelp	В модуле преобразователя недоступны никакие средства для поддержки определения времени фиксации выборки данных
1	Incoming	Интервал времени между получением сигнала триггера и фиксацией выборки данных не измеряется. По умолчанию используется значение задержки между получением сигнала триггера и фиксацией выборки данных, которое содержится в поле «Incoming propagation delay through the data transport logic field» («Входящая задержка распространения через логику передачи данных») ЭТДП канала преобразователя (см. 8.5.2.41)
2	Outgoing	Интервал времени между фиксацией выборки данных и передачей выборки данных не измеряется. По умолчанию используется значение задержки между фиксацией выборки данных и передачей выборки данных, которое содержится в поле «Outgoing propagation delay through the data transport logic field» («Исходящая задержка распространения через логику передачи данных») ЭТДП канала преобразователя (см. 8.5.2.42). Значение требуется только в том случае, если работа осуществляется в режиме выборки данных «Immediate operation» («Немедленное выполнение») (см. 5.10.1.7)
3	MInterval	Интервал времени между получением сигнала триггера и фиксацией выборки данных измеряется с помощью датчика временного интервала. ЭТДП калибровки для датчика временного интервала обеспечивает метод преобразования выходного сигнала датчика временного интервала в значение времени
4	SInterval	Интервал времени между получением сигнала триггера и фиксацией выборки данных измеряется с помощью датчика временного интервала с часами, полученными от сигнала синхронизации. ЭТДП калибровки для датчика временного интервала обеспечивает метод преобразования выходного сигнала встроенного датчика в значение времени. Примечание — Данная возможность существует не у всех представителей комплекса стандартов ИИЭР 1451

Окончание таблицы 53

Нумерация	Имя	Функция
5	ToDSense	Датчик временного события («TimeInstance») предоставляет время дня, когда была обработана выборка. Характеристики данного датчика определены в ЭТДП датчика временного события
6—127		Зарезервировано
128—255		Доступны для использования изготовителями. Примечание — Использование данных нумераций может поставить под угрозу совместимость каналов преобразователей. Для их использования в системе требуется наличие специального программного обеспечения

8.5.2.41 Поле «InPropDI» («Входящая задержка распространения через логику передачи данных»)

Тип поля: 28.

Имя поля: InPropDI.

Тип данных: действительное число одинарной точности (Float32, 4 байта).

Данное поле является обязательным, если поле источника времени выборки данных (см. 8.5.2.40) содержит значение «Incoming» («Входящие»); в противном случае поле может быть опущено. В случае если поле источника времени выборки данных содержит значение «Incoming» («Входящие»), а данное поле отсутствует, должно быть выдано сообщение о нефатальной ошибке ЭТДП, и полю источника времени выборки данных должно быть присвоено значение «NoHelp».

Данное поле содержит действительное число одинарной точности, определяющее выраженное в секундах время задержки между получением сигнала триггера на физическом уровне стека протоколов связи и сбором или применением выборки данных. Данное число может быть использовано для определения времени обработки выборки ИМП на основании времени запуска сигнала триггера.

8.5.2.42 Поле «OutPropD» («Исходящая задержка распространения через логику передачи данных»)

Тип поля: 29.

Имя поля: OutPropD.

Тип данных: действительное число одинарной точности (Float32, 4 байта).

Данное поле является обязательным, если поле источника времени выборки данных (см. 8.5.2.40) содержит значение «Outgoing» («Исходящие»); в противном случае поле может быть опущено. В случае если поле источника времени выборки данных содержит значение «Outgoing» («Исходящие»), а данное поле отсутствует, должно быть выдано сообщение о нефатальной ошибке ЭТДП, и полю источника времени выборки данных должно быть присвоено значение «NoHelp».

Данное поле содержит действительное число одинарной точности, определяющее выраженное в секундах время задержки между сигналом фиксации последней выборки (см. 5.9.2) и сигналом от логики передачи данных о транспортировке сообщения с данными. Данное число может быть использовано для определения времени, когда выборка была обработана модулем датчика, на основании времени получения сообщения с данными. Данная операция требуется только в том случае, если работа выполняется в режиме выборки данных «Immediate operation» («Немедленное выполнение») (см. 5.10.1.7).

8.5.2.43 Поле «TSError» («Погрешность задержки триггер-отсчет»)

Тип поля: 30.

Имя поля: TSError.

Тип данных: действительное число одинарной точности (Float32, 4 байта).

Данное поле является необязательным для всех каналов преобразователя. Если данное поле опущено, то СПП принимает значение данного поля за ноль, что указывает на то, что информация не предоставляется.

Данное поле содержит действительное число одинарной точности, определяющее выраженную в секундах погрешность задержки по умолчанию между сигналом триггера и получением или отработкой отсчета (выборки данных).

Выражение погрешности должно соответствовать «суммарной стандартной погрешности», определенной в NIST TechnicalNote1297 [B10].

8.5.2.44 Поле «Sampling» («Атрибут выборки»)

Тип поля: 31.

Имя поля: Sampling.

Данное поле является обязательным для всех каналов преобразователя. Если данное поле опущено, то СПП должен сообщить о фатальной ошибке ЭТДП.

Подполя, составляющие данный тип, следующие:

- возможность установки режимов выборки;
- режим выборки по умолчанию.

8.5.2.45 Поле «SampMode» («Атрибут возможности установки режимов выборки»)

Тип поля: 48.

Имя поля: SampMode.

Тип данных: 8-разрядное целое число без знака (Uint8, 1 байт).

Данное поле является обязательным для всех каналов преобразователя. Если данное поле опущено, то СПП должен сообщить о фатальной ошибке ЭТДП.

Данный атрибут используется для указания режимов выборки данных (см. 5.10.1), которые поддерживаются данным каналом преобразователя. Как показано в таблице 54, каждому допустимому режиму выборки в данном поле присваивается бит. Если бит, присвоенный заданному режиму выборки, установлен на единицу, то канал преобразователя может работать в этом режиме. В случае установки более чем одного бита режим выборки данных является программируемым.

Таблица 54 — Атрибут возможности установки режимов выборки

Бит	Определение
0 (младший значащий бит)	Установлен на единицу, если канал преобразователя может работать в режиме «Trigger-initiated» («Выборка данных по срабатыванию триггера») (см. 5.10.1.1)
1	Установлен на единицу, если канал преобразователя может работать в режиме «Free-running without pre-trigger» («Автономная выборка данных без предварительно заданного счетчика триггера») (см. 5.10.1.2)
2	Установлен на единицу, если канал преобразователя может работать в режиме «Free-running with pre-trigger» («Автономная выборка данных с предварительно заданным счетчиком триггера») (см. 5.10.1.3)
3	Установлен на единицу, если канал преобразователя может работать в режиме «Continuous Sampling» («Непрерывная выборка») (см. 5.10.1.6)
4	Установлен на единицу, если канал преобразователя может работать в режиме выборки данных «Immediate operation» («Немедленное выполнение») (см. 5.10.1.7)
5	Зарезервировано
6	Зарезервировано
7 (старший значащий бит)	Зарезервировано

8.5.2.46 Поле «SDefault» («Режим выборки данных по умолчанию»)

Тип поля: 49.

Имя поля: SDefault.

Тип данных: 8-разрядное целое число без знака (Uint8, 1 байт).

Данное поле является обязательным для всех каналов преобразователя, способных работать более чем в одном режиме выборки (см. 8.5.2.45). Если допускается только один режим выборки данных, то режим выборки по умолчанию должен являться единственным доступным режимом выборки. В случае если допускается несколько режимов выборки, а данное поле опущено или установлено несколько битов, СПП должен сообщить о фатальной ошибке ЭТДП.

Данный атрибут используется для указания режима выборки (см. 5.10.1), используемого по умолчанию для данного канала преобразователя. Как показано в таблице 55, каждому допустимому режиму выборки присвоен бит данного поля. Если бит, присвоенный заданному режиму выборки, установлен на единицу, то данный бит идентифицирует режим выборки по умолчанию для данного канала преобразователя. В данном поле может быть установлен только один бит.

Таблица 55 — Атрибут режима выборки по умолчанию

Бит	Определение
0 (младший значащий бит)	Установлен на единицу, если режим «Trigger-initiated» («Выборка данных по срабатыванию триггера») (см. 5.10.1.1) является режимом выборки по умолчанию для данного канала преобразователя
1	Установлен на единицу, если режим «Free-running without pre-trigger» («Автономная выборка данных без предварительно заданного счетчика триггера») (см. 5.10.1.2) является режимом выборки по умолчанию для данного канала преобразователя
2	Установлен на единицу, если режим «Free-running with pre-trigger» («Автономная выборка данных с предварительно заданным счетчиком триггера») (см. 5.10.1.3) является режимом выборки по умолчанию для данного канала преобразователя
3	Установлен на единицу, если режим «Continuous sampling» («Непрерывная выборка») (см. 5.10.1.6) является режимом выборки по умолчанию для данного канала преобразователя
4	Установлен на единицу, если режим выборки данных «Immediate operation» («Немедленное выполнение») (см. 5.10.1.7) является режимом выборки по умолчанию для данного канала преобразователя
5	Зарезервировано
6	Зарезервировано
7 (старший значащий бит)	Зарезервировано

8.5.2.47 Поле «Buffered» («Атрибут передачи данных через буфер»)

Тип поля: 33[1)].

Имя поля: Buffered.

Тип данных: 8-разрядное целое число без знака (Uint8, 1 байт).

Данное поле является необязательным. Если данное поле опущено, то СПП должен считать, что у канала преобразователя имеется только один доступный буфер.

Данный атрибут описывает режимы передачи данных через буфер (см. 5.10.3), доступные для данного канала преобразователя. В таблице 56 приведен перечень допустимых значений для данного атрибута.

Таблица 56 — Атрибут передачи данных через буфер

Значение	Описание
0	Канал преобразователя имеет только один доступный буфер
1	Канал преобразователя имеет несколько доступных буферов и может работать только в режиме «Buffered» («Передача данных через буфер»)
2	Канал преобразователя имеет несколько доступных буферов и может работать как в режиме «Buffered» («Передача данных через буфер»), так и в режиме «Unbuffered» («Передача данных без использования буфера»). По умолчанию используется режим «Unbuffered»
3	Канал преобразователя имеет несколько доступных буферов и может работать как в режиме «Buffered» («Передача данных через буфер»), так и в режиме «Unbuffered» («Передача данных без использования буфера»). По умолчанию используется режим «Buffered»
4—255	Зарезервировано

[1)] В оригинале ISO/IEC/IEEE 21450:2010 допущена ошибка. Ошибочно приведено значение «32». В таблице 48 для данного поля приведено значение «33».

8.5.2.48 Поле «EndOfSet» («Атрибут операции «набор данных закончен»)

Тип поля: 34[1)].

Имя поля: EndOfSet.

Тип данных: 8-разрядное целое число без знака (Uint8, 1 байт).

Данное поле является обязательным для исполнительного устройства канала преобразователя. Данное поле может быть опущено для датчиков и датчиков событий. В случае если данное поле опущено для исполнительного устройства со значением поля «Maximum data repetitions» («Максимальное повторение данных») (см. 8.5.2.28) больше нуля, должно быть выдано сообщение о фатальной ошибке ЭТДП. В случае если данное поле опущено для исполнительного устройства со значением поля «Maximum data repetitions» («Максимальное повторение данных»), равным нулю, то должно быть выдано сообщение о нефатальный ошибке ЭТДП и включен режим «Hold» («Удержать»).

Данный атрибут описывает режимы работы «End-of-data-set» («Набор данных закончен») (см. 5.10.4), поддерживаемые данным исполнительным устройством. В таблице 57 приведен перечень допустимых значений для данного атрибута.

Таблица 57 — Атрибут операции «набор данных закончен»

Значение	Описание
0	Не применяется
1	Данный канал преобразователя удерживает последнее значение из набора данных до тех пор, пока не будет получен другой сигнал триггера, как описано в 5.10.4.1
2	Данный канал преобразователя зацикливает данные последнего набора до тех пор, пока не будет получен другой сигнал триггера, как описано в 5.10.4.2
3	Данный канал преобразователя может работать либо в режиме «Hold» («Удержать»), либо в режиме «Recirculate» («Зациклить»). Два данных режима являются взаимоисключающими. По умолчанию используется режим «Hold» («Удержать»)
4	Данный канал преобразователя может работать либо в режиме «Hold» («Удержать»), либо в режиме «Recirculate» («Зациклить»). Два данных режима являются взаимоисключающими. По умолчанию используется режим «Recirculate» («Зациклить»)
5—255	Зарезервировано

8.5.2.49 Поле «DataXmit» («Атрибут передачи данных»)

Тип поля: 32[2)].

Имя поля: DataXmit.

Тип данных: 8-разрядное целое число без знака (Uint8, 1 байт).

Данное поле является необязательным атрибутом для датчиков или датчиков событий каналов преобразователя. Данное поле может быть опущено для исполнительных устройств. Если данное поле опущено для датчика или датчика событий, то СПП должен считать, что данный канал преобразователя имеет возможность работы только в режиме «Commanded» («Передача данных только по команде») (см. 5.10.2.1).

Данный атрибут описывает режимы передачи данных, которые поддерживает данный канал преобразователя (см. 5.10.2). В таблице 58 приведен перечень допустимых значений для данного атрибута.

Таблица 58 — Атрибут передачи данных

Значение	Описание
0	Зарезервировано
1	Данный канал преобразователя имеет возможность работы только в режимах «Commanded» («Передача данных только по команде») (см. 5.10.2.1)

[1)] В оригинале ISO/IEC/IEEE 21450:2010 допущена ошибка. Ошибочно приведено значение «33». В таблице 48 для данного поля приведено значение «34».

[2)] В оригинале ISO/IEC/IEEE 21450:2010 допущена ошибка. Ошибочно приведено значение «33». В таблице 48 для данного поля приведено значение «32».

Окончание таблицы 58

Значение	Описание
2	Данный канал преобразователя имеет возможность работы в режиме «Streaming» («Потоковый режим передачи данных») при заполненном буфере (см. 5.10.2.2) или в режиме «Commanded» («Передача данных только по команде»)
3	Данный канал преобразователя имеет возможность работы в режиме «Streaming» («Потоковый режим передачи данных») с фиксированным интервалом (см. 5.10.2.3) или в режиме «Commanded» («Передача данных только по команде»)
4	Данный канал преобразователя имеет возможность работы в режиме «Commanded» («Передача данных только по команде»), в режиме «Streaming» («Потоковый режим передачи данных») при заполненном буфере или в режиме «Streaming» («Потоковый режим передачи данных») с фиксированным интервалом
5—255	Зарезервировано

8.5.2.50 Поле «EdgeRpt» («Атрибут «Сообщение о достижении порогового значения»)

Тип поля: 35.

Имя поля: EdgeRpt.

Тип данных: 8-разрядное целое число без знака (Uint8, 1 байт).

Данное поле является обязательным атрибутом для датчика событий каналов преобразователя. Данное поле может быть опущено для датчиков или исполнительных устройств. Если данное поле опущено или для датчика событий отображается зарезервированное значение, то СПП должен сообщить о фатальной ошибке ЭТДП.

Данный атрибут описывает рабочие режимы «Edge-to-report» («Сообщение о достижении порогового значения») (см. 5.10.6), поддерживаемые данным датчиком событий. В таблице 59 приведен перечень допустимых значений для данного атрибута.

Таблица 59 — Опции рабочего режима «Edge-to-report» («Сообщение о достижении порогового значения»)

Значение	Описание
0	Не применяется
1	Данный датчик событий сообщает только о переходе верхнего порогового значения
2	Данный датчик событий сообщает только о переходе нижнего порогового значения
3	Данный датчик событий сообщает о переходах как верхнего, так и нижнего пороговых значений
4	Зарезервировано
5	Данный датчик событий имеет возможность сообщения об обоих типах переходов, но по умолчанию он сообщает о переходе верхнего порогового значения
6	Данный датчик событий имеет возможность сообщения об обоих типах переходов, но по умолчанию он сообщает о переходе нижнего порогового значения
7	Данный датчик событий имеет возможность сообщения об обоих типах переходов, но по умолчанию он сообщает о переходах обоих пороговых значений
8—255	Зарезервировано

8.5.2.51 Поле «ActHalt» («Атрибут остановки исполнительного устройства»)

Тип поля: 36.

Имя поля: ActHalt.

Тип данных: 8-разрядное целое число без знака (Uint8, 1 байт).

Данное поле является обязательным атрибутом для исполнительного устройства каналов преобразователя. Данное поле может быть опущено для датчиков или датчиков событий. В случае если данное поле опущено или для исполнительного устройства отображается зарезервированное значение, СПП должен сообщить о фатальной ошибке ЭТДП.

Данный атрибут описывает режимы «Actuator-halt» («Остановка исполнительного устройства») (см. 5.10.7), поддерживаемые данным исполнительным устройством. Данный режим определяет поведение исполнительного устройства при получении команды «Transducer Channel Idle» («Перейти в режим ожидания канала преобразователя») (см. 7.1.4.2). В таблице 60 приведен перечень допустимых значений для данного атрибута.

Таблица 60 — Операции режима «Actuator-halt» («Остановка исполнительного устройства»)

Значение	Описание
0	Не применяется
1	Остановить немедленно
2	Остановить по окончании набора данных
3	Перейти в заданное состояние
4—255	Зарезервировано

8.5.2.52 Поле «Directon» («Направление чувствительности»)

Тип поля: 37.

Имя поля: Directon.

Тип данных: 8-разрядное целое число без знака (Uint8, 1 байт).

Данное поле является необязательным.

Для датчиков, измеряющих физические явления с трехмерными пространственными свойствами, такие как ускорение, скорость или перемещение, данное поле определяет направление относительно прямоугольной системы координат, указанной изготовителем датчика. Датчик должен иметь на выходе положительный сигнал в случае, если физическое явление применяется к датчику в заданном направлении.

Направление определяется с помощью «правила правой руки» в «правосторонней» системе координат: пальцы правой руки указывают направление + X, ладонь — направление + Y, большой палец — направление + Z.

Для одноосевых датчиков используемая ось должна совпадать с осью Z.

Изготовитель датчика должен определить по крайней мере две оси из трех путем маркировки на корпусе датчика.

Направление физических явлений, измеряемых каналом преобразователя, должно быть определено с помощью нумерации, приведенной в таблице 61.

Таблица 61 — Нумерация направлений чувствительности

Значение	Описание
0	Не применяется
1	+ X
2	– X
3	+ Y
4	– Y
5	+ Z
6	– Z
7—255	Зарезервировано для будущего расширения

8.5.2.53 Поле «DAngles» («Направляющие углы»)

Тип поля: 38.

Имя поля: DAngles.

Тип данных: два действительных числа одинарной точности (Float32, 4 байта).

Данное поле является необязательным.

Два действительных числа одинарной точности определяют два угла, измеряемые от опорной плоскости и опорного направления, обозначенных изготовителем на канале преобразователя. Первое число представляет собой значение координаты ρ, выраженное в радианах, в правосторонней цилиндрической пространственной системе координат. Второе число представляет собой значение координаты ϕ, выраженное в радианах, в правосторонней цилиндрической пространственной системе координат.

8.5.2.54 Поле «ESOption» («Дополнительные параметры датчиков событий»)

Тип поля: 39.

Имя поля: ESOption.

Значение по умолчанию: 0.

Тип данных: 8-разрядное целое число без знака (Uint8, 1 байт).

Данное поле применяется только для датчика событий каналов преобразователя. Данное поле может быть опущено для датчиков и исполнительных устройств. Если данное поле опущено для датчика событий, то СПП должен сообщить о фатальной ошибке ЭТДП.

У датчика событий есть дополнительные параметры: изменяемая битовая последовательность, верхнее пороговое значение и/или гистерезис. Датчик событий также имеет возможность обнаружения несоответствий в настройках этих параметров, как описано в 5.6.2.3. Представленная ниже нумерация определяет способность модуля преобразователя выявлять эти несоответствия и сообщать о них.

Дополнительные параметры приведены в таблице 62.

Таблица 62 — Дополнительные параметры датчика событий

Значение	Описание
0	Не применяется
1	Последовательность/пороговое значение/гистерезис не изменяемы
2	Параметры изменяемы, обнаружены несоответствия
3	Параметры изменяемы, несоответствия не обнаружены
4—255	Зарезервировано

8.5.2.55

Примечание — Содержание подпункта 8.5.2.55 исключено, поскольку оно полностью повторяет содержание подпункта 8.5.2.17 оригинала ISO/IEC/IEEE 21450:2010.

8.6 ЭТДП калибровки

Данное поле является необязательной ЭТДП. Функция ЭТДП калибровки заключается в предоставлении всей информации, используемой коррекционным программным обеспечением совместно с каналом преобразователя, к которому идет обращение.

8.6.1 Процесс коррекции

Коррекция — это процесс применения определенной математической функции к данным канала преобразователя, полученным от одного или более каналов преобразователя, и/или данным, полученным от других элементов программного обеспечения. В данном подразделе приведено описание моделирования процесса коррекции для обеспечения понимания особенностей разработки и использования записей ЭТДП калибровки для осуществления коррекции.

Коррекция предназначена для согласования двух различных чисел, связанных с каналом преобразователя:

- числа со стороны СПП. Данное число представляет собой значение канала преобразователя, представленное в поле физических единиц измерения ЭТДП канала преобразователя (см. 8.5.2.6) и преобразованное с использованием постоянных преобразования единиц измерения СИ (см. 8.6.3.4), содержащихся в калибровочной ЭТДП. Данное число используется для представления данных каналов преобразователей в системе пользователя;
- числа со стороны преобразователя. Данное число считывается из оборудования канала преобразователя или записывается в него (обычно аналого-цифровой или цифро-аналоговый преобразователь).

Цель коррекции зависит от типа канала преобразователя, которому она адресована. Тем не менее применение коррекции одинаково и не зависит от типа канала преобразователя:

- для датчиков коррекция принимает в качестве входного сигнала данные со стороны канала преобразователя, которому она адресована, и, возможно, данные от других каналов преобразователя. Выходной коррекционный сигнал представляет собой число со стороны СПП;
- для исполнительных устройств коррекция принимает в качестве входного сигнала число со стороны СПП для канала преобразователя, которому она адресована, описывающее следующее состояние исполнительного устройства, и, возможно, данные от других каналов преобразователя. Выходной коррекционный сигнал представляет собой число со стороны преобразователя.

Данные числа, либо со стороны СПП, либо со стороны преобразователя, могут быть использованы в качестве входного сигнала для функции коррекции другого канала преобразователя, как выбрано в поле «Correction input Transducer Channel number» («Номер входного канала преобразователя коррекции») (см. 8.6.3.15).

8.6.1.1 Метод

Для датчика «коррекция» является процессом, при котором постоянные, определяемые в процессе калибровки, применяются к выходному сигналу датчика, чтобы преобразовать существующее число в требуемую для системы форму. Для исполнительного устройства «коррекция» применяется к числу, предоставляемому системой, чтобы преобразовать его в форму, требуемую исполнительным устройством.

Процесс калибровки представляет собой процесс задания уравнения, отвечающего за сегмент диапазона преобразователя и описывающего передаточную функцию преобразователя для данного сегмента. Для любой заданной калибровки определяются один или более сегментов. До настоящего времени самым распространенным числом сегментов является значение «один», что означает, что большинство преобразователей могут быть описаны одним уравнением. Наиболее часто используется уравнение (5):

$$y = mx + b = b + mx. \quad (5)$$

В ЭТДП калибровки определены два метода. Первый метод основан на том, что большинство калибровок используют линейное преобразование для одного сегмента, поэтому данный особый случай определяется отдельно. Второй метод является общим методом, который будет работать практически с любыми функциями коррекции. Данный общий метод может быть использован с несколькими сегментами и уравнениями второго и более порядков. Метод также может учитывать несколько входных сигналов, что может быть использовано для компенсации выходного сигнала с одного преобразователя в отношении изменений, вызванных факторами, не связанными с измеряемой величиной.

8.6.1.2 Линейный метод

Данный метод является частным случаем общего метода, для которого функция коррекции имеет вид, представленный в уравнении (5), и когда требуется только один сегмент.

8.6.1.3 Общий метод

Метод, описанный в настоящем стандарте, является попыткой разработать единый метод, который может быть использован для формирования любой функции коррекции, в том числе линейным методом. Все часто используемые функции коррекции состоят из двух различных этапов. Первым этапом является сегментация. В рамках данного этапа калибровочную кривую делят на сегменты для обеспечения желаемой точности с учетом достижения приемлемой степени полиномиального уравнения. Несмотря на то что чаще всего используется выделение одного сегмента, принятие решение об использовании одного сегмента также является сегментацией. Предельный случай представляет собой поиск по таблице, в которой для каждой возможной битовой последовательности в пределах заданного диапазона определен свой сегмент. Второй этап функции коррекции заключается в определении полиномиальной кривой, соответствующей отклику преобразователя в пределах каждого сегмента. Стоит отметить, что линейное уравнение, то есть многочлен первой степени, используется наиболее часто. В случае поиска по таблице используется многочлен (полином) нулевой степени.

Второй момент, который следует учитывать, заключается в том, что большинство функций коррекции используют только одну входную переменную. Однако если для данной функции должно быть определено общее решение, необходимо рассмотреть случай, когда для одного выходного сигнала требуется несколько входных сигналов. Обычно функцию с несколькими входными сигналами используют для компенсации температурных воздействий на выходной сигнал датчика или устройства согласования сигналов или для коррекции поперечной чувствительности в тензометрических установках.

Уравнение, приведенное к правильной математической форме, имеет вид, представленный формулой (6), и определяется как полином (многопараметрический полином). Члены уравнения «*H*» являются смещениями, которые вычитаются из входного сигнала для сведения к минимуму текущего числа, которое возводится в степень, чтобы снизить вероятность операций умножения, приводящих к числовому переполнению. Данное значение смещения может выходить за пределы сегмента, для которого оно определено. Следует отметить, что уравнение полинома в заданном сегменте может быть также выражено формулой (7), где член «*X* – *H*» формулы (6) заменен на «*x*».

$$\sum_{i=0}^{D_1}\sum_{j=0}^{D_2}\dots\sum_{p=0}^{D_n} C_{i,j,\dots,p}\left[X_1-H_1\right]^i\left[X_2-H_2\right]^j\dots\left[X_n-H_n\right]^p. \quad (6)$$

D_k является степенью входного члена X_k. Это означает, что D_k — высшая степень, в которую возводится $[X_k - H_k]$ в любом члене полинома. Следует отметить, что степень каждого входного члена может быть различной.

$$y=\left(A_0+A_1x_1+A_2x_1^2\dots+A_ix_1^i\right)\left(B_0+B_1x_2+B_2x_2^2\dots+B_jx_2^j\right)(\dots)\left(N_0+N_1x_n+N_2x_n^2\dots+N_px_n^p\right). \quad (7)$$

В случае когда имеется два входных сигнала, полиномы могут быть перемножены для получения уравнения (8). Принимая во внимание, что произведение постоянных также является постоянной, уравнение можно переписать в виде формулы (9):

$$y=A_0B_0+A_1B_0x_1+A_2B_0x_1^2+A_3B_0x_1^3+\dots A_iB_0x1_i+A_0B_1x_2+A_1B_1x_1x_2+A_2B_1x_1^2x_2+\dots A_iB^jx_1^ix_2^j, \quad (8)$$

$$C_0+C_1x_x+C_2x_1^2+C_3x_1^3+\dots C_nx_1^i+C_{n+1}x_2+C_{n+2}x_1x_2+C_{n+3}x_1^2x_2+\dots C_mx_1^ix_2^j. \quad (9)$$

Добавление входных сигналов не изменяет вид уравнения, а лишь приводит к увеличению числа членов.

Формулы (7)—(9) являются примерами данной функции, представленной в двух простейших и хорошо известных формах. Формула (11)[1)] представляет собой форму данного уравнения, которая используется для коррекции по таблице. Уравнение (10) представляет собой форму, которую примет уравнение, если функция коррекции является линейной функцией. Вторая форма уравнения (11) является более распространенной формой написания данного уравнения:

$$y = C_0, \quad (10)$$

$$y = (C_0 + C_1x)=b + mx. \quad (11)$$

Сегментация одного или нескольких каналов преобразователя делит входной домен на ячейки с ортогональными границами. Например, для двумерного полинома ячейки являются прямоугольниками. Каждая ячейка имеет свой собственный набор коэффициентов $C_{i,j,\dots p}$.

Для процесса коррекции с двумя входными сигналами, показанного на рисунке 15, в случае если каждый входной сигнал имеет степень 1, полином для каждой ячейки описывается следующим образом:

$$C_{0,0}+C_{0,1}\cdot\left(X_2-H_2\right)+C_{1,0}\cdot\left(X_1-H_1\right)+C_{1,1}\cdot\left(X_1-H_1\right)\cdot\left(X_2-H_2\right). \quad (12)$$

Как отмечено в формуле (12), для каждого сегмента X_1 существуют различные значения H_1. То есть H_1 одинаково для сегментов 1, 2 и 3, но отличается для сегментов 1 и 4. Аналогичным образом H_2 изменяется от сегмента 1 к сегменту 2 или от сегмента 4 к сегменту 5. Коэффициенты $C_{0,0}$ — $C_{1,1}$ различны для каждого сегмента. Коррекционное программное обеспечение должно определить, в каком из сегментов X_1 или X_2 происходит измерение, и должно выбрать, соответственно, коэффициенты и смещения.

[1)] В оригинале ISO/IEC/IEEE 21450:2010 допущена ошибка. Номер формулы пропущен. В тексте настоящего стандарта приведено возможное значение для номера формулы. Значение требует уточнения.

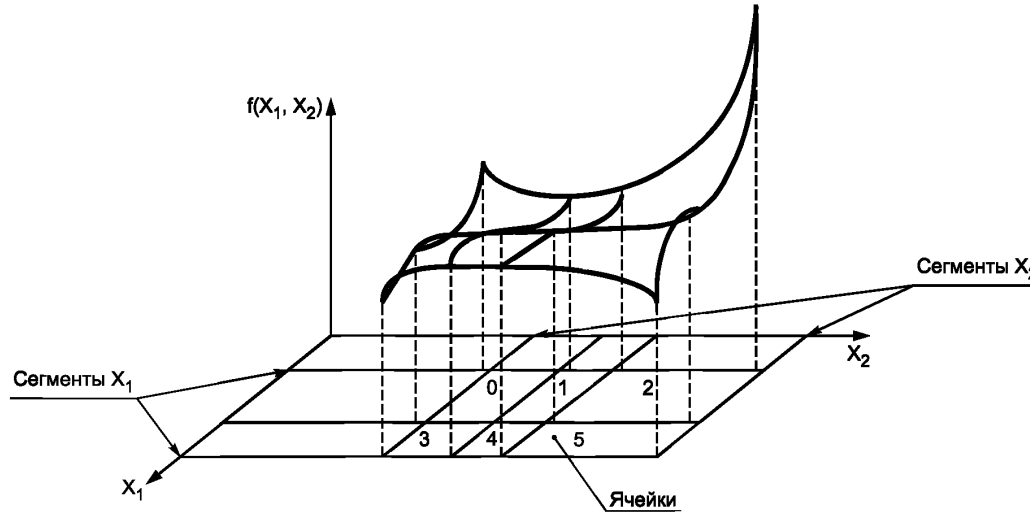


Рисунок 15 — Двумерная функция, разбитая на ячейки 2x3

8.6.1.4 Применение процесса коррекции

В случае если поле «Calibration key» («Ключ калибровки») (см. 8.5.2.2) в ЭТДП канала преобразователя имеет значение «CAL_SUPPLIED», алгоритм коррекции осуществляется в СПП, главном процессоре или в другом месте системы. Рекомендуется выполнение алгоритма в СПП.

В случае если поле «Calibration key» («Ключ калибровки») имеет значение «TIM_CAL_SUPPLIED» или «TIM_CAL_SELF», алгоритм коррекции должен быть выполнен в ИМП.

Предполагается, но не является обязательным, что коррекционное программное обеспечение будет использовать для своих вычислений числовой формат с плавающей запятой. В связи с этим требуется используемое коррекционным программным обеспечением преобразование в числовой формат и из числового формата и применение всевозможных моделей данных каналов преобразователя. В случае если коррекция осуществляется в ИМП, требуется преобразование в/из моделей данных, используемых только в ИМП. Преобразование записей ЭТДП калибровки также может быть необходимо для использования в коррекционном программном обеспечении. Описание метода преобразования выходит за рамки настоящего стандарта.

Применение процесса коррекции в СПП, главном процессоре или другом месте системы должно осуществляться в соответствии со следующими правилами:

- коррекция на датчике должна быть запущена после считывания с ИМП новых данных со стороны преобразователя;
- коррекция на исполнительном устройстве должна быть запущена после того, как новые данные получены со стороны СПП, и перед тем, как скорректированные данные со стороны преобразователя записаны в ИМП;
- корректирующий программный механизм должен использовать значения, которые в настоящий момент доступны для любых других обязательных данных каналов преобразователя;
- применение процесса коррекции не должно инициировать запуск сигналов триггера или процессы считывания на любом канале преобразователя;
- применение процесса коррекции не должно инициировать процессы записи на канале преобразователя, которому она не адресована;
- число со стороны преобразователя имеет тип данных, указанный в поле «Data model» («Модель данных») (см. 8.5.2.24), поле «Data model length» («Длина модели данных») (см. 8.5.2.25) и поле «Model significant bits» («Старшие значащие биты модели») (см. 8.5.2.26) в ЭТДП канала преобразователя.

Применение процесса коррекции в ИМП должно осуществляться в соответствии со следующими правилами:

- коррекция на датчике может быть запущена после получения новой выборки или после получения каналом преобразователя сигнала триггера. В случае если коррекция запускается после полу-

чения сигнала триггера, время подготовки к считыванию канала преобразователя (см. 8.5.2.35) должно включать в себя время, необходимое для коррекции;

- коррекция на исполнительном устройстве может быть запущена после того, как в канал преобразователя записаны новые данные, и перед запуском канала преобразователя с помощью сигнала триггера. В случае если коррекция запускается с помощью сигнала триггера, время подготовки к записи канала преобразователя (см. 8.5.2.34) должно включать в себя время, необходимое для коррекции;
- корректирующий программный механизм может использовать значения, которые в настоящий момент доступны в ИМП для любых других обязательных данных каналов преобразователя. В качестве дополнительной функции каждый входной сигнал может быть обработан, как только он был получен;
- коррекция не должна приводить к запуску любого канала преобразователя, участвующего в коррекции;
- число со стороны СПП имеет тип данных, указанный в поле «Data model» («Модель данных») (см. 8.5.2.24), поле «Data model length» («Длина модели данных») (см. 8.5.2.25) и поле «Model significant bits» («Старшие значащие биты модели») (см. 8.5.2.26) в ЭТДП канала преобразователя.

Независимо от того, где применяется процесс коррекции, он должен осуществляться в соответствии со следующими правилами:

- применение процесса коррекции к одному каналу преобразователя не должно изменять данные со стороны СПП или данные со стороны другого канала преобразователя, даже если другой канал преобразователя является входным сигналом для процесса коррекции данного канала преобразователя;
- если число повторений данных для адресуемого канала преобразователя больше нуля (например, векторные данные), то число повторений данных для любых других каналов преобразователя, используемых в коррекции, должно быть либо равно нулю (скалярному), либо равно числу повторения данных адресуемого канала преобразователя. Коррекция должна применяться с использованием векторных элементов последовательно, когда каждый входной векторный сигнал должен создавать выходной векторный сигнал. Для коррекции каждого векторного элемента скалярные данные используются без изменений;
- в процессе коррекции при необходимости могут использоваться или создаваться цифровые данные типа физических единиц измерения (например, если данные являются просто «счетчиками»).

Следует принимать во внимание порядок и доступность входных сигналов для процесса коррекции. В случае необходимости процесс коррекции получает и использует данные, которые являются доступными. Если один из параметров уже обновлен, а другой параметр еще не был обновлен, это может привести к изменениям в результатах. Это особенно важно при использовании скорректированных данных. Входные сигналы для процесса коррекции, которые представляют собой скорректированные значения, полученные от других каналов преобразователя, должны быть преобразованы до начала коррекции в интересующем канале преобразователя.

8.6.1.5 Преобразование между единицами СИ и единицами измерения выходного сигнала процесса коррекции

Постоянные преобразования единиц СИ (см. 8.6.3.4) могут быть использованы для преобразования чисел в полях ЭТДП между различными наборами единиц измерения.

8.6.1.6 Преобразование единиц измерения выходного сигнала процесса коррекции в единицы СИ

Преобразование в единицы СИ из единиц измерения выходного сигнала процесса коррекции осуществляется с помощью формулы (13):

$$SI = Slope(Value) + Intercept, \quad (13)$$

где SI — число, выраженное в единицах СИ;

Slope (наклон) — задается в поле «SI units conversion slope» («Наклон преобразования единиц измерения СИ») (см. 8.6.3.5) данной ЭТДП;

Intercept (пересечение) — задается в поле «SI units conversion intercept» («Пересечение преобразования единиц измерения СИ») (см. 8.6.3.6) данной ЭТДП;

Value (значение) — число в единицах выходного сигнала процесса коррекции.

8.6.1.7 Преобразование единиц СИ в единицы измерения выходного сигнала процесса коррекции

Преобразование единиц СИ в единицы измерения выходного сигнала процесса коррекции осуществляется с помощью формулы (14):

$$Value = (SI - Intercept)/Slope, \quad (14)$$

где SI — число, выраженное в единицах СИ;

Slope (наклон) — задается в поле «SI units conversion slope» («Наклон преобразования единиц измерения СИ») (см. 8.6.3.5) данной ЭТДП;

Intercept (пересечение) — задается в поле «SI units conversion intercept» («Пересечение преобразования единиц измерения СИ») (см. 8.6.3.6) данной ЭТДП;

Value (значение) — число в единицах выходного сигнала процесса коррекции.

8.6.2 Доступ

Доступ к ЭТДП калибровки осуществляется с помощью команд «Query TEDS» («Запросить ЭТДП»), «Read TEDS segment» («Считать сегмент ЭТДП»), «Write TEDS segment» («Записать сегмент ЭТДП»), «Update TEDS» («Обновить ЭТДП»). Аргумент команды должен определять код доступа к ЭТДП калибровки, как показано в таблице 17.

Данная ЭТДП может быть реализована как ЭТДП только для чтения или как ЭТДП для чтения/записи по выбору изготовителя. Тем не менее если ЭТДП реализована как ЭТДП только для чтения, то такие команды канала преобразователя, как «Write TEDS segment» («Записать сегмент ЭТДП») и «Update TEDS» («Обновить ЭТДП»), не должны применяться.

8.6.3 Блок данных

В таблице 63 и на рисунке 16 показана структура данных, которая должна использоваться для ЭТДП калибровки. В последующих подпунктах рассмотрено каждое из полей данных для данной структуры.

8.6.3.1 Поле «TEDSID» («Идентификатор ЭТДП»)

Идентификатор ЭТДП должен соответствовать структуре, определенной в 8.3.

Данное поле является обязательным. В случае если данное поле опущено или содержит недопустимые значения, СПП должен выдать сообщение о фатальной ошибке ЭТДП.

8.6.3.2 Поле «LstCalDt» («Время и дата последней калибровки»)

Тип поля: 10.

Имя поля: LstCalDt.

Тип данных: значение времени (TimeInstance, 8 байтов).

Данное поле является необязательным.

Данное поле содержит время и дату проведения последней калибровки канала преобразователя. Оно выражено в формате TimeInstance, описанном в 4.9.2.

8.6.3.3 Поле «CalInrvl» («Калибровочный интервал»)

Тип поля: 11.

Имя поля: CalInrvl.

Тип данных: продолжительность времени (TimeDuration, 8 байтов).

Данное поле является необязательным.

Калибровочный интервал представляет собой выраженный в секундах интервал времени, в течение которого канал преобразователя может работать без калибровки и предоставлять данные в пределах рабочей погрешности, описанной в 8.6.3.9.

8.6.3.4 Поле «SIConvrt» («Постоянные преобразования единиц СИ»)

Тип поля: 12.

Имя поля: SIConvrt.

Данное поле является обязательным. Если данное поле опущено, то СПП должен сообщить о нефатальной ошибке ЭТДП. В случае если единица измерения откалибрована в единицах СИ, то значение «SI units conversion slope» («Наклон преобразования единиц СИ») должно быть установлено на «один», а значение «SI units conversion intercept» («Пересечение преобразования единиц СИ») — на «ноль».

Данное поле состоит из двух подполей:

- «SI units conversion slope» («Наклон преобразования единиц СИ»);
- «SI units conversion intercept» («Пересечение преобразования единиц СИ»).

Примечание — Не во всех таблицах постоянных преобразования приведено одно и то же число с плавающей точкой для наклона и пересечения для данного преобразования. Некоторые таблицы обеспечивают большую точность по сравнению с другими, в остальных таблицах значения округляются по-разному. При сравнении значений, указанных в данной ЭТДП, с числами из таблицы коэффициентов преобразования для определения физических единиц на выходе процесса коррекции с использованием постоянных из данной ЭТДП такие различия должны быть приняты во внимание.

8.6.3.5 Поле «SISlope» («SI units conversion slope»,«Наклон преобразования единиц СИ»)

Тип поля: 30.

Имя поля: SISlope.

Тип данных: действительное число одинарной точности (Float32, 4 байта).

Данное поле является обязательным. Если данное поле отсутствует, то СПП должен сообщить о нефатальной ошибке ЭТДП.

Данное поле содержит число, которое при умножении на выходной сигнал процесса коррекции и добавлении к числу значения «SI units conversion intercept» («Пересечение преобразования единиц СИ») дает число, которое представляет собой физическое значение в единицах СИ.

Если выходной сигнал процесса коррекции выражен в единицах СИ, то значение данного поля должно быть равно одному.

8.6.3.6 Поле «Intrcpt» («SI units conversion intercept», «Пересечение преобразования единиц СИ»)

Тип поля: 31.

Имя поля: Intrcpt.

Тип данных: действительное число одинарной точности (Float32, 4 байта).

Данное поле является обязательным. Если данное поле отсутствует, то СПП должен сообщить о нефатальной ошибке ЭТДП.

Данное поле содержит число, которое при добавлении к значению «SI units conversion slope» («Наклон преобразования единиц СИ»), умноженному на выходной сигнал процесса коррекции, дает число, которое представляет собой физическое значение в единицах СИ.

Если выходной сигнал процесса коррекции выражен в единицах СИ, то значение данного поля должно быть равно нулю.

Таблица 63 — Структура блока данных ЭТДП калибровки

Тип поля	Имя поля	Описание	Тип	Обязательное (О)/ необязательное (Н/о) поле	Длина данных (байты)
—		Длина ЭТДП	UInt32		4
0—2	—	Зарезервировано	—	—	—
3	TEDSID	Идентификатор ЭТДП	UInt8	О	4
4—9	—				
Информация, относящаяся к дате калибровки					
10	LstCalDt	Время и дата последней калибровки	TimeInstance	Н/о	8
11	CalInrvl	Калибровочный интервал	TimeDuration	Н/о	
Информация о единицах измерения					
12	SIConvrt	Постоянные преобразования единиц СИ	—	Н/о	—
30	SISlope	Наклон преобразования единиц СИ	Float32	Н/о	4
31	Intrcpt	Пересечение преобразования единиц СИ	Float32	Н/о	4
Рабочие пределы и погрешность					
13	LowLimit	Рабочая нижняя граница диапазона	Float32	Н/о	4
14	HiLimit	Рабочая верхняя граница диапазона	Float32	Н/о	4
15	OError	Рабочая погрешность	Float32	Н/о	4
Математическое преобразование, которое должно быть выполнено над данными до или после коррекции					
16	OConvert	Действие после преобразования	UInt8	Н/о	1

Окончание таблицы 63

Тип поля	Имя поля	Описание	Тип	Обязательное (О)/ необязательное (Н/о) поле	Длина данных (байты)
17	IConvert	Действие до преобразования	UInt8	Н/о	1
TLV-кортеж 20 используется для случая линейного метода преобразования					
20	LinOnly	Данное поле используется в том случае, когда требуется только линейное односекционное преобразование	—	Н/о	—
TLV-кортеж 21 содержит описание одного канала преобразователя, который участвует в коррекции, определенной в данной ЭТДП					
21	XdcrBlk	Описание канала преобразователя	—	Н/о	—
40	Element	Номер элемента	UInt16	Н/о	1
41	ChanNum	Входной канал преобразователя коррекции	UInt16	Н/о	1
42	ChanKey	Ключ входного канала преобразователя коррекции	UInt8	Н/о	1
43	Degree	Порядок (степень) канала преобразователя	UInt8	Н/о	1
44	STable	Таблица граничных значений сегмента	—	—	—
45	OTable	Таблица значений смещения сегмента	Float32Array	Н/о	Примечание 1
46	LoBndry	Массив пределов нижней границы	Float32Array	Н/о	Примечание 1
47	HiBndry	Предел верхней границы	Float32	Н/о	4
TLV-кортеж 22 содержит набор коэффициентов для одного сегмента коррекции, указанного в данной ЭТДП					
22	CoefBlk	Коэффициент полинома	—	Н/о	—
50	CellNum	Номер ячейки сегмента, к которой применяется данный набор коэффициентов	UInt8	Н/о	1
51	CoefSet	Набор коэффициентов, применяемый к данной ячейке	Float32Array	Н/о	Примечание 2
18—19	—	Зарезервировано	—	—	—
23—29	—	Зарезервировано	—	—	—
48—49	—	Зарезервировано	—	—	—
52—127	—	Зарезервировано	—	—	—
128—255	—	Открыто для изготовителей	—	—	—
—		Контрольная сумма	UInt16	—	2

Примечание 1 — Число сегментов, умноженное на 4.

Примечание 2 — ((Степень элемента 1) + 1) × ((Степень элемента 2) +1) (...) × ((Степень элемента n) + 1) × 4.

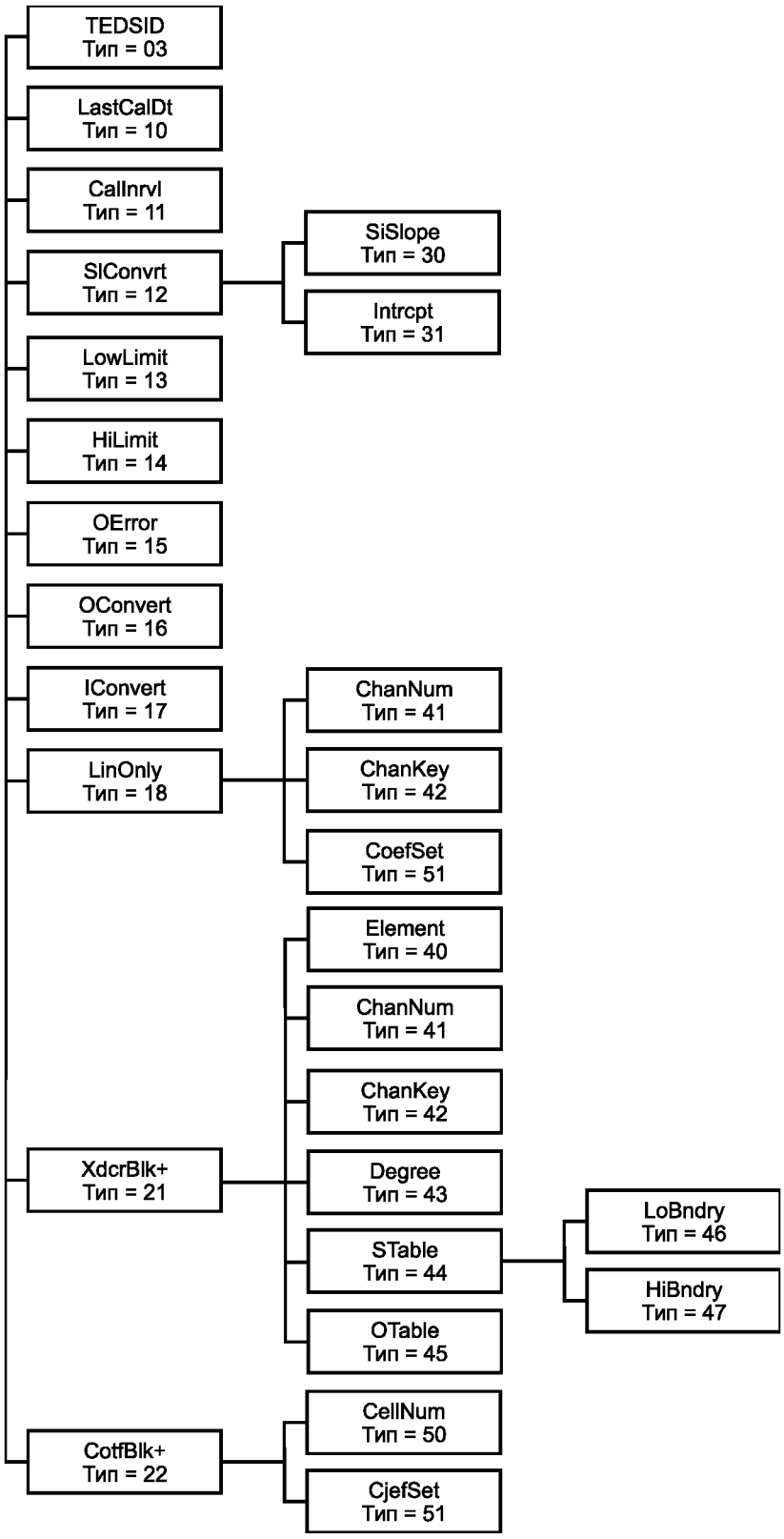


Рисунок 16 — Структура ЭТДП калибровки

8.6.3.7 Поле «LowLimit» («Рабочая нижняя граница диапазона»)
Тип поля: 13.
Имя поля: LowLimit.

Тип данных: действительное число одинарной точности (Float32, 4 байта).

Данное поле является необязательным. Если данное поле не предоставляется, а значения «SI units conversion slope» («Наклон преобразования единиц СИ») (см. 8.6.3.5) и «SI units conversion intercept» («Пересечение преобразования единиц СИ») (см. 8.6.3.6) равны одному и нулю соответственно, то СПП должен использовать проектное значение нижней рабочей границы диапазона, указанное в ЭТДП канала преобразователя (см. 8.5.2.7). В противном случае СПП должен сообщить о фатальной ошибке ЭТДП.

Для датчиков данное поле должно определять самое низкое допустимое значение данных канала преобразователя после применения коррекции, выраженное в единицах измерения, соответствующих данной ЭТДП калибровки. Если скорректированные данные канала преобразователя находятся ниже данной границы, то рабочая погрешность, рассмотренная в 8.6.3.9, может быть превышена.

Для исполнительных устройств данное поле должно определять самое низкое допустимое значение данных канала преобразователя до применения коррекции, выраженное в единицах измерения, соответствующих данной ЭТДП калибровки. Запись скорректированных данных канала преобразователя ниже данной границы может привести к выходу основных характеристик преобразователя за пределы, установленные изготовителем в спецификациях ИМП.

Примечание — Для каналов преобразователей, которые используют несколько входных сигналов для формирования одного выходного сигнала, данная граница, которая выражается на основе выходного сигнала, будет в большинстве случаев являться номинальным (паспортным), а не точным значением.

Если данный параметр не применяется, то ему должно быть присвоено значение NaN (см. 4.5.1).

Примечание — В качестве примера, когда границы диапазона не применяются, можно привести блок переключателей, моделируемый как N-байтовые данные. В данном случае в обоих полях границ диапазона должны быть установлены значения NaN. Это не означает, что границы диапазона не применяются к N-байтовым данным. Например, 12-разрядное целое число без единиц измерения, такое как необработанный выходной сигнал с аналого-цифрового преобразователя, также будет смоделирован как N-байтовые данные. В этом случае границы диапазона применимы.

8.6.3.8 Поле «HiLimit» («Рабочая верхняя граница диапазона»)

Тип поля: 14.

Имя поля: HiLimit.

Тип данных: действительное число одинарной точности (Float32, 4 байта).

Данное поле является необязательным. Если данное поле не предоставляется и значения «SI units conversion slope» («Наклон преобразования единиц СИ») (см. 8.6.3.5) и «SI units conversion intercept» («Пересечение преобразования единиц СИ») (см. 8.6.3.6) равны одному и нулю соответственно, то СПП должен использовать проектное значение верхней границы диапазона, указанное в ЭТДП канала преобразователя (см. 8.5.2.19). В противном случае СПП должен сообщить о фатальной ошибке ЭТДП.

Для датчиков данное поле должно определять самое высокое допустимое значение данных канала преобразователя после применения коррекции, выраженное в единицах измерения, соответствующих данной ЭТДП калибровки. Если скорректированные данные канала преобразователя превышают данную границу, то рабочая погрешность, рассмотренная в 8.6.3.9, может быть превышена.

Для исполнительных устройств данное поле должно определять самое высокое допустимое значение данных канала преобразователя до применения коррекции, выраженное в единицах измерения, соответствующих данной ЭТДП калибровки. Запись скорректированных данных канала преобразователя выше данной границы может привести к выходу основных характеристик преобразователя за пределы, установленные изготовителем в спецификациях ИМП.

Примечание — Для каналов преобразователей, которые используют несколько входных сигналов для формирования одного выходного сигнала, данная граница, которая выражается на основе выходного сигнала, будет в большинстве случаев являться номинальным (паспортным), а не точным значением.

Если данный параметр не применяется, то ему должно быть присвоено значение NaN (см. 4.5.1).

8.6.3.9 Поле «OError» («Рабочая погрешность»)

Тип поля: 15.

Имя поля: OError.

Тип данных: действительное число одинарной точности (Float32, 4 байта).

Данное поле является необязательным. Если данное поле отсутствует, то СПП должен использовать погрешность при наихудших условиях, указанную в ЭТДП канала преобразователя.

Данное поле использует «суммарную стандартную погрешность», определенную в NISTTechnicalNote1297 [B10]. Значение данного поля должно быть выражено в единицах измерения, соответствующих данной ЭТДП калибровки.

Если данный параметр не применяется, то ему должно быть присвоено значение NaN (см. 4.5.1).

8.6.3.10 Поле «OConvert» («Действие после преобразования»)

Тип поля: 16.

Имя поля: OConvert.

Тип данных: 8-разрядное целое число без знака (Uint8, 1 байта).

Данное поле является необязательным. Если данное поле опущено, то СПП должен считать, что действия после преобразования не требуется.

Поле «OConvert» («Действие после преобразования») содержит нумерацию, определяющую математическую операцию, которая должна быть выполнена для выходного сигнала процесса коррекции для получения значения в единицах измерения, заданных в ЭТДП канала преобразователя. В таблице 64 перечислены допустимые значения для данного поля.

8.6.3.11 Поле «IConvert» («Действие до преобразования»)

Тип поля: 17.

Имя поля: IConvert.

Тип данных: 8-разрядное целое число без знака (Uint8, 1 байт).

Данное поле является необязательным. Если данное поле опущено, то СПП должен считать, что действия до преобразования не требуется.

Поле «IConvert» («Действие до преобразования») содержит нумерацию, определяющую математическую операцию, которая должна быть выполнена для входного сигнала процесса коррекции для получения значения в единицах измерения, заданных в ЭТДП канала преобразователя. В таблице 64 перечислены допустимые значения для данного поля.

Таблица 64 — Действие до или после коррекции

Величина	Значение
0	Действие до или после коррекции не требуется
1	Инверсия: применяется «1/x»
2	Логарифм по основанию 10: применяется «log10(x)»
3	Экспонента с основанием 10: применяется «10x»
4	Натуральный логарифм: применяется «ln(x)»
5	Экспонента с основанием e: применяется «ex»
6—255	Зарезервировано

8.6.3.12 Поле «LinOnly» («Только линейное преобразование»)

Тип поля: 20.

Имя поля: LinOnly.

Данное поле является обязательным, если используется только метод линейного преобразования. Должны быть предусмотрены данное поле или поля 21 и 22. Если поля 20 или 21 и 22 не представлены, то СПП должен выдать сообщение о фатальной ошибке ЭТДП.

Данное поле описывает все постоянные, необходимые для одного канала преобразователя, когда преобразование содержит единственную секцию и является линейным. Если используется данное поле, то поля 21 и 22 и все связанные подполя должны быть опущены. Данное поле состоит из следующих подполей:

- поле «Correction input Transducer Channel number» («Число входного канала преобразователя коррекции») (см. 8.6.3.15). Данное поле может быть опущено, если число входного канала преобразователя коррекции совпадает с числом канала преобразователя, к которому применяется данная калибровочная ЭТДП;

- поле «Correction input Transducer Channel key» («Ключ входного канала преобразователя коррекции») (см. 8.6.3.16). Если данное поле опущено, то должно быть сделано допущение, что данные для датчика будут получены со стороны преобразователя процесса коррекции (рисунок 17). Для исполнительного устройства должно быть сделано допущение, что входной сигнал для коррекции приходит со стороны СПП процесса коррекции;

- поле «Set of coefficients» («Набор коэффициентов») (см. 8.6.3.24). Набор коэффициентов должен быть ограничен значениями «SI units conversion slope» («Наклон преобразования единиц СИ») и «SI units conversion intercept» («Пересечение преобразования единиц СИ»).

В случае если включены поля типа 21 и 22, то поле 20 должно быть опущено.

8.6.3.13 Поле «XdcrBlk» («Описание канала преобразователя»)

Тип поля: 21.

Имя поля: XdcrBlk.

Данное поле является обязательным, если используется общий метод преобразования. Данное поле должно быть представлено вместе с полями 20 или 22. Если поле 20 или поля 21 и 22 не представлены, то СПП должен выдать сообщение о фатальной ошибке ЭТДП.

Если включен тип поля 20, то поля 21 и 22 должны быть опущены.

Данное поле описывает все постоянные, за исключением калибровочных коэффициентов, необходимые для одного канала преобразователя, который является частью процесса коррекции. Данное поле состоит из следующих подполей:

- поле «Element number» («Номер элементов»);
- поле «Correction input Transducer Channel number» («Номер входного канала преобразователя коррекции»);
- поле «Correction input Transducer Channel key» («Ключ входного канала преобразователя коррекции»);
- поле «Transducer Channel degree» («Порядок (степень) канала преобразователя»);
- поле «Segment boundary values table» («Таблица граничных значений сегмента»);
- поле «Segment offset values table» («Таблица значений смещений сегмента»).

Если несколько каналов преобразователя обеспечивают входной сигнал процесса коррекции для канала преобразователя, к которому применяется данная ЭТДП калибровки, то каждый из этих входных каналов преобразователя должен иметь описание в данной ЭТДП калибровки.

8.6.3.14 Поле «Element» («Номер элементов»)

Тип поля: 40.

Имя поля: Element.

Тип данных: 16-разрядное целое число без знака (UInt16, 2 байта).

В случае если включен тип поля 21, то поле «Element» («Номер элементов») является обязательным. В случае если поле 21 включено, а данное поле опущено, то СПП должен выдать сообщение о фатальной ошибке ЭТДП.

Номер элементов используется для определения номера ячеек. Номер элементов определяет порядок нумерации ячеек, как описано в 8.6.3.23.

8.6.3.15 Поле «ChanNum» («Номер входного канала преобразователя коррекции»)

Тип поля: 41.

Имя поля: ChanNum.

Тип данных: 16-разрядное целое число без знака (UInt16, 2 байта).

В случае если включен тип поля 21, то данное поле является обязательным. Если включено поле 21, а данное поле опущено, то СПП должен выдать сообщение о фатальной ошибке ЭТДП.

Данное поле содержит номер канала преобразователя для данного входного сигнала процесса коррекции.

8.6.3.16 Поле «ChanKey» («Ключ входного канала преобразователя коррекции»)

Тип поля: 42.

Имя поля: ChanKey.

Тип данных: 8-разрядное целое число без знака (Uint8, 1 байт).

В случае если включен тип поля 21, то данное поле является обязательным. Если включено поле 21, а данное поле опущено, то СПП должен выдать сообщение о фатальной ошибке ЭТДП.

Данный ключ служит для определения источника значения данных, связанных с данным каналом преобразователя. Возможные значения ключа и их описания приведены в таблице 65.

Таблица 65 — Ключ входного канала преобразователя коррекции

Значение	Описание
0	Число входного сигнала коррекции должно быть получено со стороны преобразователя процесса коррекции
1	Число входного сигнала коррекции должно быть получено со стороны СПП процесса коррекции
2—255	Недействительные

В зависимости от того, доступны ли одновременно скорректированные и нескорректированные данные, для канала преобразователя возможны следующие варианты:

- если доступны только нескорректированные данные, то значение ключа должно быть равно нулю;
- если доступны только скорректированные данные, то значение ключа должно быть равно одному;
- если доступны обе формы данных, то пользователь может выбрать любую форму для входного сигнала процесса коррекции. Данный выбор зависит от факторов, связанных с калибровочными кривыми, и его описание выходит за рамки настоящего стандарта.

На рисунке 17 приведены значения ключей. Процесс коррекции для канала преобразователя 3 использует данные из канала преобразователя 1 в качестве одного из входных сигналов. Для процесса коррекции канала преобразователя 3 данные канала преобразователя 1 могут быть получены с любой стороны канала преобразователя 1 в зависимости от ключа, указанного в описании канала преобразователя 1 в ЭТДП калибровки канала преобразователя 3.

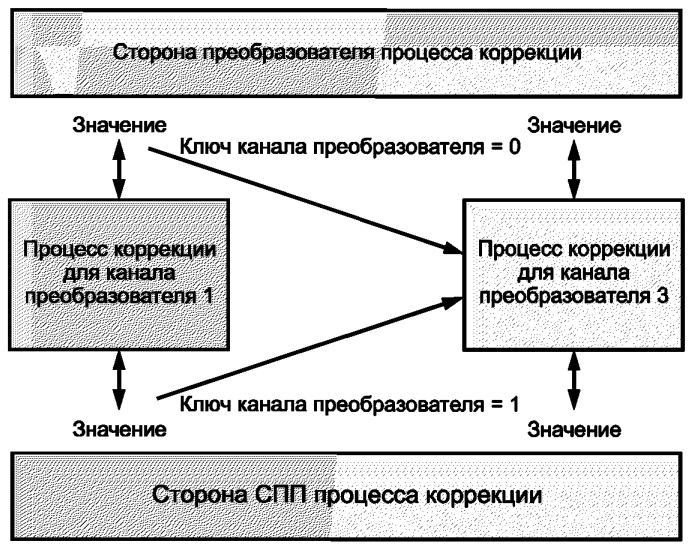


Рисунок 17 — Значение ключа входного канала преобразователя коррекции

8.6.3.17 Поле «Degree» («Порядок (степень) канала преобразователя»)

Тип поля: 43.

Имя поля: Degree.

Тип данных: 8-разрядное целое число без знака (Uint8, 1 байт).

В случае если включен тип поля 21, то данное поле является обязательным. Если поле 21 включено, а данное поле опущено, то СПП должен выдать сообщение о фатальной ошибке ЭТДП.

Порядок канала преобразователя должен совпадать с порядком соответствующего входного сигнала коррекции. Порядок — самая высокая степень, в которую возводится каждый член полинома,

описывающего входной сигнал коррекции, для которого применяется данное описание канала преобразователя.

8.6.3.18 Поле «STable» («Таблица граничных значений сегмента»)

Тип поля: 44.

Имя поля: STable.

В случае если включен тип поля 21, то данное поле является обязательным. Если поле 21 включено, а данное поле опущено, то СПП должен выдать сообщение о фатальной ошибке ЭТДП.

Данное поле состоит из следующих подгрупп:

- массив пределов нижней границы;
- предел верхней границы.

8.6.3.19 Поле «LoBndry» («Массив пределов нижней границы»)

Тип поля: 46.

Имя поля: LoBndry.

Тип данных: массив действительных чисел одинарной точности (Float32Array).

В случае если включен тип поля 21, то данное поле является обязательным. Если поле 21 включено, а данное поле опущено, то СПП должен выдать сообщение о фатальной ошибке ЭТДП.

Массив пределов нижней границы является одномерным массивом (таблицей), содержащим значения нижней границы для каждого сегмента входного сигнала, описываемого с помощью описания данного канала преобразователя.

Элементы должны определять границы сегментов области (домена) в порядке возрастания номеров. Следует отметить, что область (домен) может быть основана как на данных со стороны преобразователя, так и на данных со стороны СПП в соответствии со значением ключа выходного канала преобразователя коррекции (см. 8.6.3.16). Тем не менее табличные значения указываются в формате с плавающей точкой, поэтому при использовании данных со стороны преобразователя требуется процесс преобразования данных.

Первый элемент должен иметь значение, меньшее или равное минимально возможному значению входного сигнала. Последний элемент должен иметь значение, равное минимальному значению входного сигнала для последнего сегмента. Значение входного сигнала (X), которое должно быть определено как принадлежащее заданному сегменту, должно удовлетворять следующему уравнению:

$$B \le X < B_{(S+1)},$$

где B — предел нижней границы сегмента;

$B_{(S+1)}$ — предел нижней границы для следующего более высокого сегмента.

Данные значения хранятся в ЭТДП как действительные числа одинарной точности, несмотря на то что они могут представлять данные, которые находятся в другом числовом представлении.

8.6.3.20 Поле «HiBndry» («Предел верхней границы»)

Тип поля: 47.

Имя поля: HiBndry.

Тип данных: действительное число одинарной точности (Float32).

В случае если включен тип поля 21, то данное поле является обязательным. Если поле 21 включено, а данное поле опущено, то СПП должен выдать сообщение о фатальной ошибке ЭТДП.

Предел верхней границы является числом одинарной точности, которое больше максимального значения для самого высокого сегмента в диапазоне.

8.6.3.21 Поле «OTable» («Таблица значений смещений сегмента»)

Тип поля: 45.

Имя поля: OTable.

Тип данных: массив действительных чисел одинарной точности (Float32Array)[1)].

В случае если включен тип поля 21, то данное поле является обязательным. Если поле 21 включено, а данное поле опущено, то СПП должен выдать сообщение о фатальной ошибке ЭТДП.

Таблица значений смещений сегмента представляет собой одномерный массив (таблицу), содержащий одно значение смещения для каждого сегмента входного сигнала, описываемого при помощи описания данного канала преобразователя.

[1)] В оригинале ISO/IEC/IEEE 21450:2010 допущена ошибка. Ошибочно приведено «Float32».

Следует отметить, что O_s (так же как и H_k) могут быть основаны на данных входного сигнала как со стороны преобразователя, так и со стороны СПП в соответствии со значением ключа входного канала преобразователя коррекции (см. 8.6.3.16).

Данные значения хранятся в ЭТДП как действительные числа одинарной точности, несмотря на то что они могут представлять данные, которые находятся в другом числовом представлении.

8.6.3.22 Поле «CoefBlk» («Блок коэффициентов полинома»)

Тип поля: 22.

Имя поля: CoefBlk.

Данное поле является обязательным, если используется общий метод преобразования. Данное поле должно быть представлено вместе с полем 21[1)] или 20. Если поле 20 или поля 21 и 22 не представлены, то СПП должен выдать сообщение о фатальной ошибке ЭТДП.

Если тип поля 20 включен, то поля 21 и 22 должны быть опущены.

Данное поле состоит из двух подблоков:

- номер ячейки;
- набор коэффициентов.

8.6.3.23 Поле «CellNum» («Номер ячейки»)

Тип поля: 50.

Имя поля: CellNum.

Тип данных: 16-разрядное целое число без знака (UInt16, 2 байта).

В случае если включен тип поля 22, данное поле является обязательным. Если поле 22 включено, а данное поле опущено, то СПП должен выдать сообщение о фатальной ошибке ЭТДП.

Ячейки нумеруются от 0 до m. Ячейка 0 является ячейкой с самыми низкими номерами сегментов всех входных каналов преобразователей. Нумерация продолжается для следующего более высокого номера сегмента домена X_n (домен сегментов для элемента с наибольшим номером). По достижении последнего сегмента домена X_k следующая ячейка закрывает самый низший сегмент домена X_k снова и следующий, более высокий сегмент домена X_{k-1}. Рассмотрим в качестве примера коррекцию двух каналов преобразователей с двумя сегментами (A1, A2) на первом входном сигнале коррекции и тремя сегментами (B1, B2, B3) на втором входном сигнале коррекции. В терминах, используемых в данной ЭТДП, входной сигнал A будет определен как элемент 0, а входной сигнал B как элемент 1. Ячейки должны быть пронумерованы в следующем порядке сегментов: (A1, B1) = 0, (A1, B2) = 1, (A1, B3) = 2, (A2, B1) = 3, (A2, B2) = 4 и (A2, B3) = 5.

8.6.3.24 Поле «CoefSet» («Набор коэффициентов»)

Тип поля: 51.

Имя поля: CoefSet.

Тип данных: массив действительных чисел одинарной точности (Float32Array)[2)].

Данное поле является обязательным. В случае если используется тип поля 20, то данное поле должно содержать только два значения: «Intercept» («Пересечение»), b или C_0, и следующее за ним «Slope» («Наклон»), m или C_1. Данное поле является обязательным. Если данное поле опущено, то СПП должен выдать сообщение о фатальной ошибке ЭТДП.

Набор коэффициентов представляет собой одномерный массив (таблицу), содержащий коэффициент для каждого члена в уравнении. Набор коэффициентов полинома должен соответствовать ячейке, определяемой по номеру ячейки внутри блока коэффициентов.

Примечание — Каждый элемент в массиве идентифицируется с помощью имени (заглавной буквы) с подстрочным индексом. Если мы именуем элементы как C и присваиваем им подстрочные индексы, то первый индекс представляет порядок (степень) канала преобразователя с элементом под номером 0 и может изменяться от 0 до значения, заданного для данного элемента в поле «Transducer Channel Degree» («Порядок (степень) канала преобразователя»). Второй индекс предназначен для канала преобразователя с элементом под номером 1. Это продолжается до последнего индекса, идентифицирующего запись для элемента с наибольшим номером.

Каждая запись в таблице является коэффициентом $C_{i,j,\ldots p}$, который используется в полиноме:

$$\sum_{i=0}^{D(1)}\sum_{j=0}^{D(2)}\cdots\sum_{p=0}^{D(n)} C_{i,j,\ldots,p}\left[X_1-H_1\right]^i\left[X_2-H_2\right]^j\ldots\left[X_n-H_n\right]^p. \qquad (15)$$

[1)] В оригинале ISO/IEC/IEEE 21450:2010 допущена ошибка. Ошибочно приведено «22».

[2)] В оригинале ISO/IEC/IEEE 21450:2010 допущена ошибка. Ошибочно приведено «Float32».

Коэффициенты должны храниться в таблице, начиная с $C_{0,0, \ldots, 0}$ и увеличивая самый правый подстрочный индекс. Когда любой индекс достигает своего предела, необходимо снова присвоить ему нулевое значение и увеличить значение подстрочного индекса, находящегося слева.

$C_{0,0, \ldots 0,0}$;	$C_{0,0, \ldots 01}$ [1];	$C_{0,0, \ldots 0,Dn}$
$C_{0,0, \ldots, 1,0}$;	$C_{0,0, \ldots 1,1}$;	$C_{0,0, \ldots D(n-1),Dn}$
.....		
$C_{D1,D2, \ldots D(n-1),0}$;	$C_{D1,D2, \ldots D(n-1),1}$;	$C_{D1,D2, \ldots D(n-1),Dn}$

Все блоки коэффициентов ЭТДП калибровки должны иметь одинаковую длину. Коэффициенты, которые не требуются для данного сегмента, но которые требуются для другого сегмента, должны быть установлены на ноль.

8.7 ЭТДП частотной характеристики

Данное поле является необязательной ЭТДП. Функция ЭТДП частотной характеристики заключается в предоставлении пользователю информации, касающейся частотной характеристики канала преобразователя.

ЭТДП частотной характеристики обеспечивает характеристику частотного и фазового отклика канала преобразователя. К некоторым факторам, влияющим на частотную характеристику, можно отнести: датчик, обработку (согласование) аналогового сигнала, сглаживающий фильтр и цифровую обработку сигналов. ЭТДП предоставляет сквозной отклик от аналогового датчика к цифровому выходному сигналу. Данная характеристика приводится с предположением, что выходной сигнал ИМП считывается в диапазоне частот, которые охватывают описанную частотную характеристику.

ЭТДП частотной характеристики нормируется на опорной частоте. То есть неявно предполагается, что данные преобразователя привязаны к данной частоте.

8.7.1 Доступ

Доступ к ЭТДП частотной характеристики осуществляется с помощью команд «Query TEDS» («Запросить ЭТДП»), «Read TEDS segment» («Считать сегмент ЭТДП»), «Write TEDS segment» («Записать сегмент ЭТДП») или команды «Update TEDS» («Обновить ЭТДП»). Аргумент команды должен определять код доступа к ЭТДП частотной характеристики согласно таблице 17.

Данная ЭТДП может быть реализована как ЭТДП только для чтения или как ЭТДП для чтения/записи по выбору изготовителя. Тем не менее если данная ЭТДП реализована как ЭТДП только для чтения, то команды «Write TEDS segment» («Записать сегмент ЭТДП») канала преобразователя и «Update TEDS» («Обновить ЭТДП») канала преобразователя не должны применяться.

8.7.2 Блок данных

В таблице 66 и на рисунке 18 показана структура данных, которая должна использоваться для ЭТДП частотной характеристики. В последующих подпунктах рассмотрено каждое поле данных структуры.

Таблица 66 — Структура блока данных ЭТДП частотной характеристики

Тип поля	Имя поля	Описание	Тип	Длина данных в байтах
—		Длина ЭТДП	UInt32	4
0—2	—	Зарезервировано	—	—
3	TEDSID	Идентификатор ЭТДП	UInt8	4
4—9	—	Зарезервировано	—	—
10	RefFreq	Опорная частота	Float32	4
11	RefAmp	Тестовая амплитуда	Float32	4
12	RefPhase	Фаза на опорной частоте	Float32	4

[1] В оригинале ISO/IEC/IEEE 21450:2010 допущена ошибка. Ошибочно приведено «$C_{0,0}$».

Окончание таблицы 66

Тип поля	Имя поля	Описание	Тип	Длина данных в байтах
Следующие поля содержат структуру, которая определяет одну точку данных. Структура повторяется n раз, по одному разу для каждой точки данных				
13	Points	Точки в таблице	—	—
14—127	—	Зарезервировано	—	—
128—255	—	Открыто для изготовителей	—	—
—	—	Контрольная сумма	UInt16	2

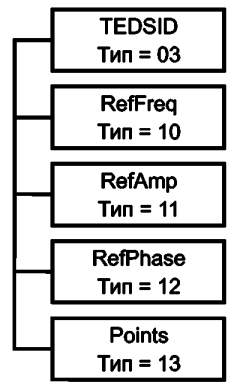


Рисунок 18 — Структура ЭТДП частотной характеристики

8.7.2.1 Поле «TEDSID» («Идентификатор ЭТДП»)

Идентификатор ЭТДП должен соответствовать структуре, определенной в 8.3.

Данное поле является обязательным. Если данное поле опущено или содержит недопустимые значения, то СПП должен выдать сообщение о фатальной ошибке ЭТДП.

8.7.2.2 Поле «RefFreq» («Опорная частота»)

Тип поля: 10.

Имя поля: RefFreq.

Тип данных: действительное число одинарной точности (Float32, 4 байта).

Данное поле является обязательным. Если данное поле отсутствует, то СПП должен выдать сообщение о фатальной ошибке ЭТДП.

В данном поле указывается опорная частота, при которой значение амплитуды определяется равным единице. Единица измерения данного поля — герц.

8.7.2.3 Поле «RefAmp» («Тестовая амплитуда»)

Тип поля: 11.

Имя поля: RefAmp.

Тип данных: действительное число одинарной точности (Float32, 4 байта).

Данное поле является обязательным. Если данное поле отсутствует, то СПП должен выдать сообщение о фатальной ошибке ЭТДП.

В данном поле указывается амплитуда входного сигнала, которая была использована для получения информации об отклике. Единицы измерения данного поля должны совпадать с единицами измерения, заданными в поле «Physical units» («Физические единицы измерения») ЭТДП канала преобразователя (см. 4.11 или 8.5.2.6).

8.7.2.4 Поле «RefPhase» («Фаза на опорной частоте»)

Поле Тип: 12.

Имя поля: RefPhase.

Тип данных: действительное число одинарной точности (Float32, 4 байта).

Данное поле является обязательным. Если данное поле опущено, то СПП должен выдать сообщение о фатальной ошибке ЭТДП.

В данном поле указывается сдвиг фазы выходного сигнала канала преобразователя на опорной частоте в поле 2 (см. 8.7.2.2). Единицей измерения для данного параметра должен быть радиан.

8.7.2.5 Поле «Points» («Точки в таблице»)

Тип поля: 13.

Имя поля: Points.

Тип данных: действительное число тройной точности (Float32, 4 байта) значения (12 байтов).

Данное поле является обязательным. Если данное поле отсутствует, то СПП должен выдать сообщение о фатальной ошибке ЭТДП.

Данное поле определяет точку данных в таблице откликов. Каждая точка данных состоит из трех подполей:

- частота;
- амплитудная характеристика;
- фазовая характеристика.

Данное поле может повторяться для описания такого числа точек, которое изготовитель считает адекватным для определения частотной характеристики канала преобразователя.

В таблице 67 приведена структура TLV-кортежа для данного поля.

Таблица 67 — Структура каждой точки

Поле	Определение
Field type (Тип поля)	Всегда равно 13 для данного поля
Length (Длина)	Всегда равно 12, так как в поле значения содержатся три числа с плавающей точкой, что требует 12 байтов
Frequency (Частота)	В данном поле указана частота, для которой применима информация об амплитуде и фазе в следующих двух полях. В качестве единицы измерения данного поля должен применяться герц
Amplitude (Амплитуда)	В данном поле указывается амплитуда выходного сигнала канала преобразователя относительно амплитуды на опорной частоте, указанной в поле 2. Амплитудная характеристика определяется отношением амплитуды на текущей частоте «Frequency» к амплитуде на опорной частоте «RefFreq»
Phase (Фаза)	В данном поле указывается сдвиг фазы выходного сигнала канала преобразователя на текущей частоте «Frequency». В качестве единицы измерения для данного параметра должен быть использован радиан

8.8 ЭТДП передаточной функции

Данное поле является необязательной ЭТДП. Она содержит ряд постоянных, которые могут быть использованы для описания передаточной функции преобразователя. К некоторым факторам, влияющим на передаточную функцию, можно отнести датчик, обработку (преобразование) аналогового сигнала, сглаживающий фильтр и цифровую обработку сигнала. Передаточная функция обеспечивает сквозной отклик от аналогового датчика к цифровому выходному сигналу. Она предназначена для того, чтобы СПП или другой элемент системы был способен компенсировать частотную характеристику преобразователя.

Передаточная функция ЭТДП нормируется на опорной частоте. То есть неявно предполагается, что данные преобразователя привязаны к данной частоте.

8.8.1 Процесс компенсации

В случае если известна передаточная функция, описывающая зависимость между входным и выходным сигналами, может быть использована обратная функция для линеаризации или компенсации функции частотной характеристики системы в целом.

Компенсация частотной характеристики — это применение определенной математической функции к данным канала преобразователя. В данном разделе описан процесс моделирования процесса компенсации, что способствует общему пониманию этапов разработки и использования записей в ЭТДП передаточной функции для компенсации.

Цель компенсации зависит от типа канала преобразователя. Применение компенсации, однако, не зависит от типа канала преобразователя:

- для датчиков процесс компенсации использует в качестве входного сигнала данные со стороны преобразователя или данные до компенсации. На выходе компенсация предоставляет число со стороны СПП или число после компенсации;

- для исполнительных устройств процесс компенсации использует в качестве входного сигнала число со стороны СПП, то есть намеченное следующее состояние исполнительного устройства. На выходе коррекция предоставляет число со стороны преобразователя, скомпенсированное для фазовой и частотной характеристики контура и исполнительного устройства.

8.8.1.1 Метод

Функция компенсации определяется как обратная передаточная функция. Если передаточная функция задается для канала преобразователя как $H(f)$, то выходной сигнал канала преобразователя следует умножить на $1/H(f)$ для получения скомпенсированного выходного сигнала. Эта операция предполагает отсутствие нулей в передаточной функции в пределах частотного диапазона применения.

$H(f)$ представляется как факторизованное произведение определенного числа элементов, нормированных на опорной частоте, указанной в поле «Reference frequency» («Опорная частота») данной ЭТДП (см. 8.8.3.2). Уравнение (16) обеспечивает математическое представление данной функции.

$$H(f) = \frac{H_1(f)}{|H_1(f_{ref})|} \frac{H_2(f)}{|H_2(f_{ref})|} * .... * \frac{H_N(f)}{|H_N(f_{ref})|}. \quad (16)$$

На опорной частоте функция $H(f)$ ограничена значением $|H(f_{ref})| = 1$ с сохранением фазы. Форма каждого элемента $H_i(f)$ должна соответствовать описанию, представленному в 8.8.3.1—8.8.3.21.

С другой стороны, $H(f)$ описывается как рациональная функция, обычно называемая z-преобразованием (дискретным преобразованием Лапласа), что представлено в уравнении (17).

$$H(z) = \frac{A_0 + A_1 z^{-1} + A_2 z^{-2} + ... A_n^{z-n}}{1 + B_0 + B_1^{z-1} + B_2 z^{-2} .... B_m z^{-m}}, \quad (17)$$

где $z^{-1}(\omega) = e^{-j\omega T}$;

$\omega = 2\pi f$;

$j = \sqrt{-1}$;

и T — это временная постоянная, которая представляет задержку или время между получением выборок.

8.8.1.2 Применение

При обработке потока данных, поступающих от канала преобразователя (например, с помощью быстрого преобразования Фурье в частотную область), результирующий спектр может быть разделен на $H(f)$ для получения скорректированного спектра. Обратное преобразование может перенести результат обратно во временную область. На основе $H(f)$ могут быть получены другие функции, такие как импульсный отклик. Импульсный отклик, в свою очередь, может быть использован непосредственно к временным данным, потому что операция свертки обеспечивает необходимую коррекцию.

Для осуществления компенсации также может использоваться цифровой фильтр. Это прямо следует из z-преобразования. Факторизованная форма (форма с выраженными сомножителями) также может быть использована для расчета необходимых коэффициентов.

8.8.2 Доступ

Доступ к ЭТДП передаточной функции осуществляется с помощью команд «Query TEDS» («Запросить ЭТДП»), «Read TEDS segment» («Считать сегмент ЭТДП»), «Write TEDS segment» («Записать сегмент ЭТДП») или команды «Update TEDS» («Обновить ЭТДП»). Аргумент команды должен определять код доступа к ЭТДП передаточной функции согласно таблице 17.

Данная ЭТДП может быть реализована как ЭТДП только для чтения или как ЭТДП для чтения/записи по выбору изготовителя. Тем не менее если данная ЭТДП реализована как ЭТДП только для чтения, то команды «Write TEDS segment» («Записать сегмент ЭТДП») канала преобразователя и «Update TEDS» («Обновить ЭТДП») канала преобразователя не должны применяться.

8.8.3 Блок данных

В таблице 68 и на рисунке 19 показана структура блока данных для данной ЭТДП. В 8.8.3.1—8.8.3.21 приведено описание каждого поля данных этой структуры. Поле «Reference frequency» («Опорная частота») является обязательным. Остальные поля в данной ЭТДП являются обязательными или необязательными в зависимости от выбранного метода определения передаточной функции.

8.8.3.1 Поле «TEDSID» («Идентификатор ЭТДП»)

Описание заголовка ЭТДП приведено в 8.3.

Данное поле является обязательным. Если данное поле опущено или содержит недопустимые значения, то СПП должен выдать сообщение о фатальной ошибке ЭТДП.

8.8.3.2 Поле «RefFreq» («Опорная частота»)

Тип поля: 10.

Имя поля: RefFreq.

Тип данных: действительное число одинарной точности (Float32, 4 байта).

Данное поле является обязательным. Если данное поле опущено, то СПП должен выдать сообщение о фатальной ошибке ЭТДП.

В данном поле указывается опорная частота, для которой амплитуда определяется равной единице. Единица измерения данного поля — герц.

Таблица 68 — Структура блока данных ЭТДП передаточной функции

Тип поля	Имя поля	Описание	Тип	Число байтов
—		Длина ЭТДП	UInt32	4
0—2	—	Зарезервировано	—	—
3	TEDSID	Идентификатор ЭТДП	UInt8	4
4—9	—	Зарезервировано	—	—
Информация, относящаяся к передаточной функции				
10	RefFreq	Опорная частота	Float32	4
11	OneZero	Единственный нуль TF_SZ	Float32	4
12	OnePole	Единственный полюс TF_SP	Float32	4
13	ZeroPole	Единственный нуль функции с зависимым полюсом	—	—
14	PoleZero	Единственный полюс функции с зависимым нулем	—	—
15	ComplexZ	Комплексный нуль функции	—	—
16	ComplexP	Комплексный полюс функции	—	—
17	OneZZPol	Единственный ноль при нуле и единственном полюсе	—	—
18	CRSlope	Постоянный относительный наклон	Float32	4
19	SampleT	Время выборки/задержки	Float32	4
20	DependP	Единственный полюс функции на нуле TF_SPm (x)	Float32	4
21	DependZ	Единственный ноль функции на полюсе TF_SZm (x)	Float32	4
22	ComplexZF	Частота комплексного нуля	Float32	4
23	ComplexZQ	Фактор качества комплексного нуля	Float32	4
24	ComplexPF	Частота комплексного полюса	Float32	4
25	ComplexPQ	Фактор качества комплексного полюса	Float32	4
26—29	—	Зарезервировано	—	—
30	NCoeff	Коэффициенты числителя (A0, A1,... An)	Массив Float32	n×4
31	DCoeff	Коэффициенты знаменателя (B0, B1,... Bm)	Массив Float32	m×4
32—127	—	Зарезервировано	—	—

Окончание таблицы 68

Тип поля	Имя поля	Описание	Тип	Число байтов
128[1)]—255	—	Открыто для использования изготовителями	—	—
		Контрольная сумма		
Примечание — Число коэффициентов n или m для полей 30 или 31 может быть определено путем деления длины кортежа на 4.				

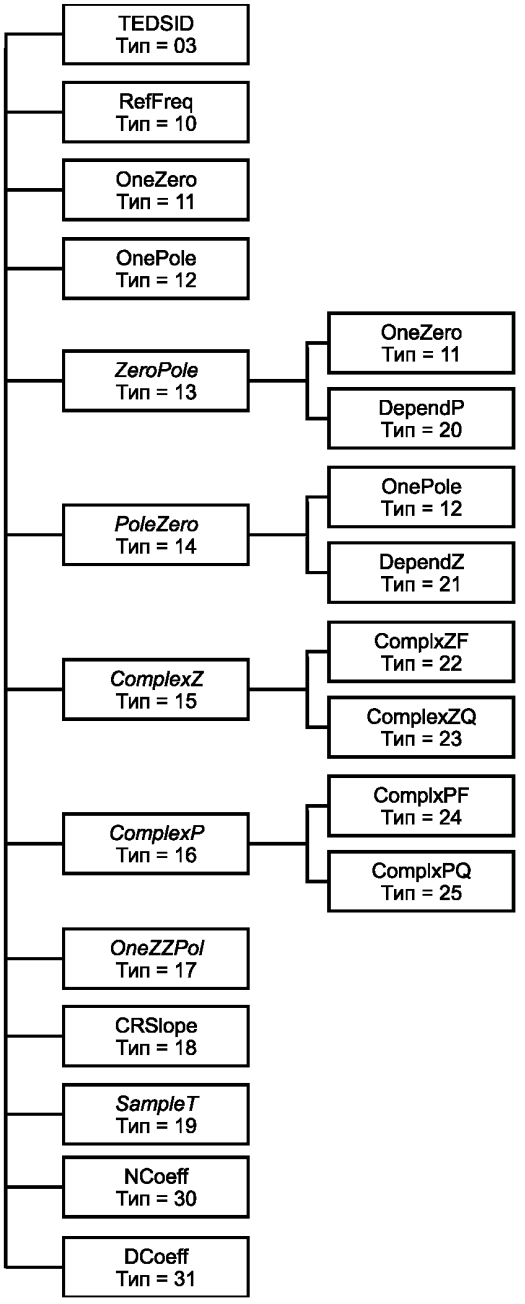


Рисунок 19 — Структура ЭТДП передаточной функции

[1)] В оригинале ISO/IEC/IEEE 21450:2010 допущена ошибка. Ошибочно приведено «12».

8.8.3.3 Поле «OneZero» («Единственный нуль функции»)

Тип поля: 11.

Имя поля: OneZero.

Тип данных: действительное число одинарной точности (Float32, 4 байта).

Данное поле является необязательным. Если данное поле опущено, то передаточная функция не содержит ни одного нуля.

Значение данного поля используется как F_{SZ} в выражении для передаточной функции, представленной в виде уравнения (18), которое непосредственно описывает фильтр высоких частот первого порядка с точкой минус 3 дБ для F_{SZ}.

$$H(f,F_{SZ}) = \left(1 + \frac{j \cdot f}{F_{SZ}}\right). \quad (18)$$

8.8.3.4 Поле «OnePole» («Единственный полюс функции»)

Тип поля: 11.

Имя поля: OnePole.

Тип данных: действительное число одинарной точности (Float32, 4 байта).

Данное поле является необязательным. Если данное поле отсутствует, то передаточная функция не содержит ни одного полюса.

Значение данного поля используется как F_{sp} в выражении для передаточной функции, представленной в виде уравнения (19), которое непосредственно описывает фильтр нижних частот первого порядка с точкой минус 3 дБ для F_{sp}.

$$H(f,F_{sp}) = \frac{1}{\left(1 + \frac{j \cdot f}{F_{sp}}\right)}. \quad (19)$$

8.8.3.5 Поле «ZeroPole» («Единственный нуль в паре нуль/полюс функции»)

Тип поля: 13.

Имя поля: ZeroPole.

Данное поле является необязательным. Если данное поле отсутствует, то передаточная функция не содержит пару нуль/полюс.

Данное поле состоит из двух подполей. Одно из них такое же, как поле «OneZero» («Единственный нуль функции»), описанное в 8.8.3.3, а другое — как поле «DependP» («Единственный полюс функции, зависящий от нуля»), описанное в 8.8.3.7.

8.8.3.6 Поле «Single zero» («Единственный нуль функции»)

Данное поле аналогично полю OneZero» («Единственный нуль функции») (см. 8.8.3.3).

Данное поле является обязательным, если присутствует поле 13. Если данное поле опущено, а поле 13 присутствует, то СПП должен выдать сообщение о фатальной ошибке ЭТДП.

8.8.3.7 Поле «DependP» («Единственный полюс функции, зависящий от нуля»)

Тип поля: 20.

Имя поля: DependP.

Тип данных: действительное число одинарной точности (Float32, 4 байта).

Данное поле является обязательным, если присутствует поле 13. Если данное поле опущено, а поле 13 присутствует, то СПП должен выдать сообщение о фатальной ошибке ЭТДП.

Каждое значение данного поля используется как «*x*» в передаточной функции, представленной уравнением (20).

$$H(f,x,v) = \frac{1}{\left(1 + \frac{j \cdot f}{x \cdot v}\right)}, \quad (20)$$

где v является значением «F_{SZ}» связанного с ним элемента поля «ZeroPole» («Единственный нуль в паре нуль/полюс функции») (см. 8.8.3.5).

8.8.3.8 Поле «PoleZero» («Единственный полюс функции с зависимым нулем»)

Тип поля: 14.

Имя поля: PoleZero.

Данное поле является необязательным. Если данное поле опущено, то передаточная функция не содержит пару полюс/ноль.

Данное поле состоит из двух подполей. Одно из них такое же, как поле «Single pole» («Единственный полюс функции»), описанное в 8.8.3.9[1)], а другое — как поле «DependZ» («Единственный нуль, зависящий от полюса»), описанное в 8.8.3.10.

8.8.3.9 Поле «Single pole» («Единственный полюс функции»)

Данное поле является таким же, как и поле «OnePole» («Единственный полюс функции»), описанное в 8.8.3.4[1)].

Данное поле является обязательным, если присутствует поле 14. Если данное поле опущено, а поле 14 присутствует, то СПП должен выдать сообщение о фатальной ошибке ЭТДП.

8.8.3.10 Поле «DependZ» («Единственный нуль функции, зависящий от полюса»)

Тип поля: 21.

Имя поля: DependZ.

Тип данных: действительное число одинарной точности (Float32, 4 байта).

Данное поле является обязательным, если присутствует поле 14. Если данное поле опущено, а поле 14 присутствует, то СПП должен выдать сообщение о фатальной ошибке ЭТДП.

Значение данного поля используется в качестве «*x*» в передаточной функции, представленной уравнением (21).

$$H(f,x,v)=\left(1+\frac{j\cdot f}{x\cdot v}\right), \tag{21}$$

где v является значением «F_{sp}» связанного с ним элемента поля «PoleZero» («Единственный полюс функции с зависимым нулем»).

8.8.3.11 Поле «ComplexZ» («Комплексный нуль функции»)

Тип поля: 15.

Имя поля: ComplexZ.

Данное поле является необязательным. Если данное поле опущено, то передаточная функция не содержит комплексного нуля.

Данное поле имеет два подполя: поле «ComplexZF» («Частота комплексного нуля») и поле «ComplexZQ» («Фактор качества комплексного нуля»).

8.8.3.12 Поле «ComplexZF» («Частота комплексного нуля»)

Тип поля: 22.

Имя поля: ComplexZF.

Тип данных: массив одинарной точности реал (Float32, 4 байта каждый).

Данное поле является обязательным, если присутствует поле 15. Если данное поле опущено, а поле 15 присутствует, то СПП должен выдать сообщение о фатальной ошибке ЭТДП.

Значение данного поля может быть использовано в качестве F_{zres} передаточной функции, представленной уравнением (22).

$$H\left(f,F_{\text{zres}},Q_{\text{z}}\right)=\left(1+\frac{j\cdot f}{Q_{\text{z}}\cdot F_{\text{zres}}}+\left(\frac{j\cdot f}{F_{\text{zres}}}\right)^{2}\right), \tag{22}$$

где параметр Q_z должен быть приведен в связанном элементе поля «ComplexZQ» («Фактор качества комплексного нуля»).

Единица измерения данного поля — герц.

8.8.3.13 Поле «ComplexZQ» («Фактор качества комплексного нуля»)

Тип поля: 23.

Имя поля: ComplexZQ.

Тип данных: действительное число одинарной точности (Float32, 4 байта).

Данное поле является обязательным, если присутствует поле 15. Если данное поле опущено, а поле 15 присутствует, то СПП должен выдать сообщение о фатальной ошибке ЭТДП.

[1)] В оригинале ISO/IEC/IEEE 21450:2010 допущена ошибка. Ошибочно приведена ссылка на 8.8.3.3.

Данное поле содержит параметр Q_z для передаточной функции комплексного нуля в соответствующем элементе поля «ComplexZ» («Комплексный нуль функции»).

Данная величина является безразмерной.

8.8.3.14 Поле «ComplexP» («Комплексный полюс функции»)

Тип поля: 16.

Имя поля: ComplexP[1)].

Если данное поле опущено, то передаточная функция не содержит комплексный полюс.

Данное поле имеет два подполя: поле «ComplexPF» («Частота комплексного полюса») и поле «ComplexPQ» («Фактор качества комплексного полюса»).

8.8.3.15 Поле «ComplexPF» («Частота комплексного полюса»)

Тип поля: 24.

Имя поля: ComplexPF.

Тип данных: действительное число одинарной точности (Float32, 4 байта).

Данное поле является обязательным, если присутствует поле 16. Если данное поле опущено, а поле 16 присутствует, то СПП должен выдать сообщение о фатальной ошибке ЭТДП.

Значение данного поля может быть использовано в качестве F_{zres} в передаточной функции, представленной уравнением (23).

$$H\left(f,F_{\text{zres}},Q_{\text{p}}\right)=\left(1+\frac{j\cdot f}{Q_{\text{p}}\cdot F_{\text{zres}}}+\left(\frac{j\cdot f}{F_{\text{zres}}}\right)^{2}\right), \tag{23}$$

где параметр Q_p должен быть приведен в соответствующем элементе поля «ComplexPQ» («Фактор качества комплексного полюса») (см. 8.8.3.16).

Единица измерения данного поля — герц.

8.8.3.16 Поле «ComplexPQ» («Фактор качества комплексного полюса»)

Тип поля: 25.

Имя поля: ComplexPQ.

Тип данных: действительное число одинарной точности (Float32, 4 байта).

Данное поле является обязательным, если присутствует поле 16. Если данное поле отсутствует, а поле 16 присутствует, то СПП должен выдать сообщение о фатальной ошибке ЭТДП.

Данное поле содержит параметр Q_p для передаточной функции комплексного полюса. Обычно оно используется для описания поведения системы с одной степенью свободы, такой как пружинная система масс в акселерометре или мембране с демпфированием воздуха в микрофоне. Параметрами являются резонансная частота и Q (или фактор качества) кривой отклика.

Данная величина является безразмерной.

8.8.3.17 Поле «OneZZPol» («Единственный нуль при нуле и единственном полюсе функции»)

Тип поля: 17.

Имя поля: OneZZPol.

Тип данных: действительное число одинарной точности (Float32, 4 байта).

Данное поле является необязательным. Если данное поле отсутствует, то передаточная функция не имеет единственного нуля при нуле и единственном полюсе функции.

Значение данного поля может быть использовано в качестве F_{hp} в передаточной функции, представленной уравнением (24).

$$H\left(f,F_{\text{hp}}\right)=\frac{\frac{j\cdot f}{F_{\text{hp}}}}{\left(1+\frac{j\cdot f}{F_{\text{hp}}}\right)}. \tag{24}$$

Уравнение (24) непосредственно описывает фильтр высоких частот первого порядка с точкой минус 3 дБ на F_{hp}. В качестве единицы измерения для данного поля должен быть использован герц.

8.8.3.18 Поле «CRSlope» («Постоянный относительный наклон»)

Тип поля: 18.

Имя поля: CRSlope.

Тип данных: действительное число одинарной точности (Float32, 4 байта).

[1)] В оригинале ISO/IEC/IEEE 21450:2010 допущена ошибка. Ошибочно приведено «ComplexZ».

Данное поле является необязательным. Если данное поле отсутствует, то передаточная функция не содержит постоянного относительного наклона. Значение данного поля используется в качестве *a* в передаточной функции, представленной уравнением (25).

$$H(f,a,F_{ref}) = \left(\frac{j \cdot f}{F_{ref}}\right)^{\frac{a}{\ln(10)}}, \quad (25)$$

где *a* — относительное изменение, соответствующее декаде частот, то есть частотам, отличающимся на один порядок. Значение F_{ref} может быть найдено в поле «Reference frequency» («Опорная частота»). Обычные значения *a* для большинства марок пьезоэлектрической керамики (титанат-цирконат свинца) находятся в диапазоне падения чувствительности порядка от минус 0,02 % до 2 % для декады частот (для частот, отличающихся на один порядок). Это связано с одновременной частотно-независимой задержкой фазы, которая, как было обнаружено, равняется минус 0,78°.

8.8.3.19 Поле «SampleT» («Время выборки/задержка»)

Тип поля: 19.

Имя поля: SampleT.

Тип данных: действительное число одинарной точности (Float32, 4 байта).

Данное поле является необязательным. Если данное поле опущено, то передаточная функция не содержит цифровой фильтр.

Значение данного поля представляет собой время между выборками или время задержки, которое используется в цифровом фильтре. В качестве единицы измерения данного поля должна быть использована секунда.

8.8.3.20 Поле «NCoeff» («Коэффициенты числителя»)

Тип поля: 30.

Имя поля: NCoeff.

Тип данных: массив действительных чисел одинарной точности (Float32, 4 байта каждый).

Данное поле является необязательным. Если данное поле опущено, то передаточная функция не содержит z-преобразования (дискретного преобразования Лапласа). Если данное поле опущено, а поле 31 присутствует, то СПП должен выдать сообщение о фатальной ошибке ЭТДП.

При использовании альтернативной формы передаточной функции (z-преобразования) данное поле содержит перечень коэффициентов, необходимых в числителе уравнения. Более подробная информация о z-преобразовании приведена в 8.8.1.1.

Примечание — Число коэффициентов данного поля может быть получено путем деления длины кортежа на четыре.

8.8.3.21 Поле «DCoeff» («Коэффициенты знаменателя»)

Тип поля: 31.

Имя поля: DCoeff.

Тип данных: массив действительных чисел одинарной точности (Float32, 4 байта каждый).

Данное поле является необязательным. Если данное поле опущено, то передаточная функция не содержит z-преобразования. Если данное поле опущено, а поле 30 присутствует, то СПП должен выдать сообщение о фатальной ошибке ЭТДП.

При использовании альтернативной формы передаточной функции (z-преобразования) данное поле содержит перечень коэффициентов, необходимых в знаменателе уравнения. Более подробная информация о z-преобразовании приведена в 8.8.1.1.

Примечание — Число коэффициентов данного поля может быть получено путем деления длины кортежа на четыре.

8.9 Текстовая ЭТДП

Данное поле относится к классу необязательных ЭТДП. Функцией данных ЭТДП является предоставление информации для отображения оператору. В таблице 17 приведены шесть ЭТДП, которые попадают в данную категорию. К ним относятся ЭТДП мета-идентификации, ЭТДП идентификации канала преобразователя, ЭТДП идентификации калибровки, командная ЭТДП, ЭТДП места нахождения и заголовка и ЭТДП географического места нахождения. Общие описания этих ЭТДП приведены в 5.5.2.6—5.5.2.9.

8.9.1 Доступ

Доступ к текстовой ЭТДП осуществляется с помощью команд «Query TEDS» («Запросить ЭТДП»), «Read TEDS segment» («Считать сегмент ЭТДП»), «Write TEDS segment» («Записать сегмент ЭТДП») или «Update TEDS» («Обновить ЭТДП»). Аргумент команды должен определять код доступа к текстовой ЭТДП, как показано в таблице 17.

Данный тип ЭТДП может быть реализован как ЭТДП только для чтения или ЭТДП для чтения/записи по выбору изготовителя. Если ЭТДП реализована как ЭТДП только для чтения, то команды ИМП «Write TEDS segment» («Записать сегмент ЭТДП») или канала преобразователя «Write segment» («Записать сегмент») и команды ИМП «Update TEDS» («Обновить ЭТДП») или канала преобразователя «Update TEDS» («Обновить ЭТДП») не должны применяться.

8.9.2 Блок данных

Текстовые ЭТДП представляют собой структуры, которые формируют один или более блоков текстовой информации. Каждый блок текста представлен на определенном языке. Первая часть данной ЭТДП является директорией, с помощью которой процессор определяет место и считывает один язык. Поле «XMLText» («Текст XML») содержит информацию для отображения, закодированную через XML. Изготовитель определяет число применяемых языков. В таблице 69 и на рисунке 20 приведено содержимое данной ЭТДП.

Таблица 69 — Структура блока данных текстовой ЭТДП

Тип поля	Название поля	Описание	Тип данных	Число байтов
—		Длина ЭТДП	UInt32	4
0—2	—	Зарезервировано	—	—
3	TEDSID	Заголовок для идентификации ЭТДП	UInt8	4
4—9	—	Зарезервировано	—	—
Следующие три поля содержат заголовок языка. Заголовок повторяется N раз, один раз для каждого поддерживаемого языка				
10	NumLang	Число N разных языковых блоков данной ЭТДП	UInt8	1
11	DirBlock	Описание языкового блока. Данный блок повторяется N раз	—	—
20	LangCode	Код языка согласно ИСО 639 (две буквы в USASCII)	UInt8	2
21	Offset	Смещение языка	UInt32	4
22	Length	Длина языка = LL	UInt32	4
23	Compress	Нумерация, определяющая используемую технику сжатия	UInt8	1
12	SubSum	Невизуализируемая контрольная сумма данных	UInt16	2
Следующие два поля включают в себя структуру, содержащую текст на одном языке. Структура повторяется N раз, один раз для каждого определенного языка				
—	XMLText	Текстовый блок на базе XML	text	LL – 2
—	XMLSum	Контрольная сумма текстового блока	UInt16	2
13—19	—	Зарезервировано	—	—
23—127	—	Зарезервировано	—	—
128—255	—	Открыто для изготовителей	—	—
—		Контрольная сумма	UInt16	2

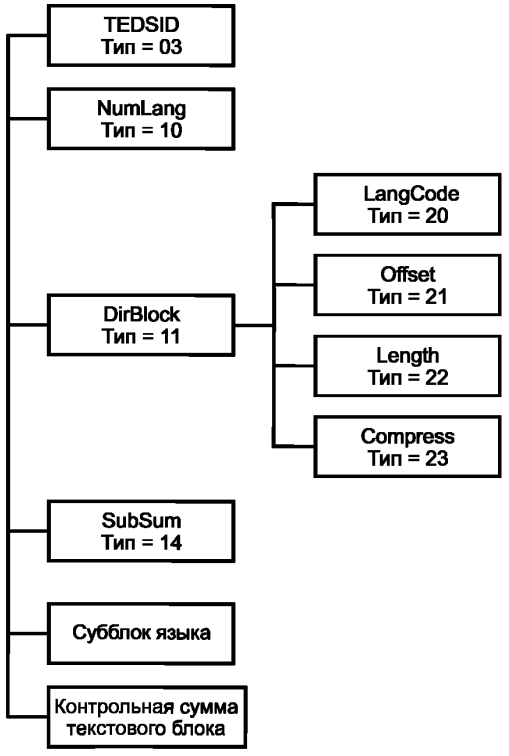


Рисунок 20 — Структура текстовой ЭТДП

8.9.2.1 Поле «TEDSID» («Идентификатор ЭТДП»)

Идентификатор ЭТДП должен соответствовать структурам, определенным в 8.3.

Данное поле является обязательным. Если данное поле опущено или содержит недопустимые значения, то СПП должен выдать сообщение о фатальной ошибке ЭТДП.

8.9.2.2 Поле «NumLang» («Число языков»)

Тип поля: 10.

Имя поля: NumLang.

Тип данных: данное поле содержит один байт, используемый для подсчета (Uint8, 1 байт).

Если данное поле отсутствует, то СПП должен считать, что представлен только один язык.

Данное поле содержит число, определяющее число языков в данной ЭТДП.

8.9.2.3 Поле «DirBlock» («Блок директории языка»)

Тип поля: 11.

Имя поля: DirBlock.

Данное поле является обязательным. Если данное поле опущено, то СПП должен выдать сообщение о фатальной ошибке ЭТДП.

Данное поле состоит из следующих подполей:

- «LangCode» («Код языка»);
- «Offset» («Смещение языка»);
- «Length» («Длина субблока языка»);
- «Compress» («Нумерация техники сжатия»).

8.9.2.4 Поле «LangCode» («Код языка»)

Тип поля: 20.

Имя поля: LangCode.

Тип данных: двухбайтовая строка символов в кодировке USASCII.

Данное поле является обязательным. Если данное поле опущено, то СПП должен выдать сообщение о фатальной ошибке ЭТДП.

В данном поле указывается язык, на котором написаны текстовые поля ЭТДП.

Значение данного поля соответствует алфавитному списку двухбуквенных языковых символов согласно стандарту ИСО 639:1988. В таблице 70 перечислены некоторые из возможных языков. Языки и диалекты, не указанные в ИСО 639, не должны использоваться в текстовых полях ЭТДП.

Таблица 70 — Примеры нумерации кодов языка

Код языка в соответствии с ИСО 639	Значение (для справки)
Зарезервировано	—
aa	афарский
da	датский
de	немецкий
en	английский
es	испанский
eu	баскский
fi	финский
fr	французский
ga	ирландский
it	итальянский
nl	голландский
no	норвежский
pl	польский
pt	португальский
ru	русский
sv	шведский
vi	вьетнамский
zu	зулусский

8.9.2.5 Поле «Offset» («Смещение языка»)

Тип поля: 21.

Имя поля: Offset.

Тип данных: 32-разрядное целое число без знака для подсчета (UInt32, 4 байта).

Данное поле является обязательным. Если данное поле опущено, то СПП должен выдать сообщение о фатальной ошибке ЭТДП.

Данное поле используется, чтобы задать положение отображаемой информации, само поле обычно не отображается.

Данное поле указывает смещение относительно начала ЭТДП, на котором расположен субблок информационных данных в формате XML для указанного языка.

8.9.2.6 Поле «Length» («Длина субблока языка»)

Тип поля: 22.

Имя поля: Length.

Тип данных: 32-разрядное целое число без знака для подсчета (UInt32, 4 байта).

Данное поле является обязательным. Если данное поле опущено, то СПП должен выдать сообщение о фатальной ошибке ЭТДП.

Данное поле используется, чтобы задать положение отображаемой информации, само поле обычно не отображается.

Данное поле показывает число байтов в субблоке языка, включая контрольную сумму. В случае полей символов длина должна равняться числу байтов, а не числу символов, так как для кодирования одного символа может потребоваться более одного байта. При интерпретации строк символов используется число байтов на символ для определения длины в символах конкретной строки.

8.9.2.7 Поле «Compress» («Нумерация техники сжатия»)

Тип поля: 23.

Имя поля: Compress.

Тип данных: 8-разрядное целое число без знака (Uint8, 1 байт).

Данное поле является необязательным. Если данное поле опущено, то система должна считать, что сжатие не используется.

Данное поле используется для идентификации алгоритма сжатия, используемого с этим языковым текстовым блоком. В таблице 71 перечислены допустимые значения для данного поля.

Таблица 71 — Нумерация алгоритмов сжатия

Нумерация	Описание
0	В языковом блоке данной ЭТДП сжатие не используется
1	WinZip
2	GZip
3	Зарезервировано
4—127	Зарезервировано
128—255	Открыто для изготовителей

8.9.2.8 Поле «SubSum» («Неотображаемая контрольная сумма данных»)

Тип поля: 14.

Имя поля: SubSum.

Тип данных: 16-разрядное целое число без знака для подсчета (UInt16, 2 байта).

Данное поле является обязательным. Если данное поле опущено, то СПП должен выдать сообщение о фатальной ошибке ЭТДП.

Данное поле содержит контрольную сумму для всех предшествующих байтов данной ЭТДП. Контрольная сумма должна представлять собой дополнение до единицы (по модулю 2^{16}) всех предыдущих байтов структуры данных, включая поле «Length» («Длина») (см. 8.1.1) и исключая данное поле «SubSum» («Неотображаемая контрольная сумма данных»).

8.9.2.9 Поле «XMLText» («Текстовый блок на основе XML»)

Тип данных: «text» («текст»).

Данное поле является обязательным. Если данное поле опущено, то СПП должен выдать сообщение о фатальной ошибке ЭТДП.

Данное поле содержит информацию для отображения с помощью приложения на основе XML (XML-savvy). Предлагаемая схема для всех текстовых ЭТДП, используемых в ИИЭР 1451.2—1997 или описанных в настоящем стандарте, приведена в приложении D.

8.9.2.10 Поле «XMLSum» («Контрольная сумма текстового блока»)

Данное поле содержит контрольную сумму для всех байтов в предыдущем поле «XMLText» («Текстовый блок на основе XML»). Контрольная сумма должна представлять собой дополнение до единицы (по модулю 2^{16}) всех байтов текстового блока на основе XML.

Данное поле является обязательным. Если данное поле отсутствует, то СПП должен выдать сообщение о фатальной ошибке ЭТДП.

8.10 Специализированная ЭТДП конечного пользователя

Данное поле является необязательной ЭТДП, которая обеспечивает хранение относящихся к приложению данных, которые пользователь хочет сохранить в ИМП или канале преобразователя. Пользователь должен определить содержание и функции специализированной ЭТДП конечного пользователя. Данная ЭТДП должна быть предназначена как для чтения, так и для записи.

8.10.1 Доступ

Специализированная ЭТДП конечного пользователя может быть связана с ИМП или каналом преобразователя. Доступ к ней осуществляется с помощью команд «Query TEDS» («Запросить ЭТДП»), «Read TEDS segment» («Считать сегмент ЭТДП»), «Write TEDS segment» («Записать сегмент ЭТДП») или «Update TEDS» («Обновить ЭТДП»). Аргумент команды должен определять код доступа к специализированной ЭТДП конечного пользователя, как показано в таблице 17.

8.10.2 Блок данных

В таблице 72 приведена структура блока данных для этой ЭТДП. Содержание и структура поля блока данных определяется пользователем.

Таблица 72[1] — Структура блока данных специализированной ЭТДП конечного пользователя

Тип поля	Имя поля	Описание	Тип данных	Число байтов
—		Длина ЭТДП	UInt32	4
0—2	—	Зарезервировано	—	—
3	TEDSID	Заголовок для идентификации ЭТДП	UInt8	4
4—9	—	Зарезервировано	—	—
10	EndUserData	Содержимое блока данных		Переменное
—		Контрольная сумма	UInt16	2

Изготовитель должен определить размер данной ЭТДП. Рекомендуется, чтобы размер данной ЭТДП вмещал в себя поле блока данных по меньшей мере в 256 байтов.

8.10.2.1 Поле «TEDSID» («Идентификатор ЭТДП»)

Идентификатор ЭТДП должен соответствовать структуре, определенной в 8.3.

Данное поле является обязательным. Если данное поле опущено или содержит недопустимые значения, то СПП должен выдать сообщение о фатальной ошибке ЭТДП.

8.10.2.2 Поле «EndUserData» («Содержимое блока данных»)

Тип поля: 10.

Имя поля: EndUserData.

Содержимое блока данных зависит от пользователя и не определено в настоящем стандарте.

8.11 ЭТДП с именем преобразователя, заданным пользователем

Данное поле является обязательной ЭТДП для ИМП и рекомендованной ЭТДП для всех преобразователей. ЭТДП с именем преобразователя, заданным пользователем, обеспечивает место для хранения имени, по которому система или конечный пользователь будут идентифицировать данный преобразователь.

Примечание — ЭТДП с именем преобразователя, заданным пользователем, предназначена для поддержки «Объектных тегов (или меток)», как определено в ИИЭР 1451.1—1999, или для других подобных целей.

8.11.1 Доступ

ЭТДП с именем преобразователя, заданным пользователем, может быть связана с ИМП или каналом преобразователя. Доступ к ней осуществляется с помощью команд «Query TEDS» («Запросить ЭТДП»), «Read TEDS segment» («Считать сегмент ЭТДП»), «Write TEDS segment» («Записать сегмент ЭТДП») или «Update TEDS» («Обновить ЭТДП»). Аргумент команды должен определять код доступа к ЭТДП с именем преобразователя, заданным пользователем, как показано в таблице 17.

Данная ЭТДП должна быть реализована как ЭТДП для чтения/записи.

8.11.2 Блок данных

В таблице 73 приведена структура блока данных для данной ЭТДП. Содержание и структура поля блока данных определяются пользователем.

Таблица 73 — Структура блока данных ЭТДП с именем преобразователя, заданным пользователем

Тип поля	Имя поля	Описание	Тип данных	Число байтов
—		Длина	UInt32	4
0—2	—	Зарезервировано	—	—
3	TEDSID	Заголовок для идентификации ЭТДП	UInt8	4
4	Format	Описание формата данной ЭТДП	UInt8	1

[1] В оригинале ISO/IEC/IEEE 21450:2010 допущены ошибки форматирования заголовка таблицы и некоторых полей.

Окончание таблицы 73

Тип поля	Имя поля	Описание	Тип данных	Число байтов
5	TCName	Имя ИМП или канала преобразователя	—	Переменное
—		Контрольная сумма	UInt16	2

Изготовитель должен определить размер данной ЭТДП, но как минимум ЭТДП должна быть достаточно большой, чтобы вмещать поле блока данных размером не менее чем 160 байтов с 32-символьным именем параметра, заголовок для идентификации ЭТДП и возможный заголовок текстовой ЭТДП. Использование заголовка текстовой ЭТДП не является обязательным для пользователя.

8.11.2.1 Поле «TEDSID» («Идентификатор ЭТДП»)

Идентификатор ЭТДП должен соответствовать структуре, определенной в 8.3.

Данное поле является обязательным. Если данное поле опущено, то СПП должен выдать сообщение о фатальной ошибке ЭТДП.

8.11.2.2 Поле «Format» («Формат»)

Тип поля: 10.

Имя поля: Format.

Тип данных: 8-разрядное целое число без знака (Uint8, 1 байт).

Данное поле является обязательным. Если данное поле опущено или содержит недопустимые значения, то СПП должен выдать сообщение о фатальной ошибке ЭТДП.

Значения для признака формата приведены в таблице 74.

Таблица 74 — Нумерация для поля «Format» признака формата

Величина	Значение
0	Определяется пользователем
1	Текстовая ЭТДП, использующая формат, определенный в 8.9
2—255	Зарезервировано для будущего расширения

В случае если поле «Format» («Формат») имеет нумерацию «0», содержание и структура поля блока данных определяются пользователем. Если блок данных является текстовым, то его содержание должно соответствовать структуре, определенной в 8.9.

Примечание — Использование нескольких языков в данной ЭТДП может вызвать проблемы с приложением пользователя.

8.11.2.3 Поле «EndUserData» («Содержимое блока данных»)

Тип поля: 11.

Имя поля: EndUserData.

Содержимое блока данных зависит от пользователя и не определено в настоящем стандарте.

8.12 ЭТДП, заданная изготовителем

ЭТДП, заданная изготовителем, может быть представлена в любом формате, требуемом программным обеспечением приложения изготовителя. Основополагающая система не должна пытаться разобрать эти ЭТДП или интерпретировать их содержание в любой форме. Если ЭТДП не является текстовой, что определяется ответом на команду «Query TEDS» («Запросить ЭТДП») (см. 7.1.1.1), то система должна просто считать данные ЭТДП и передать содержимое приложению, которое его запросило. Если ЭТДП является текстовой, то она должна соответствовать структуре, определенной в 8.9. Для ЭТДП, заданной изготовителем, которую необходимо отправить в ИМП, система должна принять информацию, применить поле длины и поля контрольной суммы и направить ее в ИМП.

8.12.1 Доступ

ЭТДП, заданная изготовителем, может быть связана с ИМП или каналом преобразователя. Доступ к ней осуществляется с помощью команд «Query TEDS» («Запросить ЭТДП»), «Read TEDS segment» («Считать сегмент ЭТДП»), «Write TEDS segment» («Записать сегмент ЭТДП») или «Update TEDS» («Обновить ЭТДП»). Аргумент команды должен определять код доступа к ЭТДП, который задается изготовителем.

Данная ЭТДП может быть реализована как ЭТДП только для чтения или как ЭТДП для чтения/записи по выбору изготовителя. В случае если ЭТДП реализована как ЭТДП только для чтения, команда ИМП «Write TEDS segment» («Записать сегмент ЭТДП»), команда канала преобразователя «Write TEDS segment» («Записать сегмент ЭТДП»), команда ИМП «Update TEDS» («Обновить ЭТДП») и команда канала преобразователя «Update TEDS» («Обновить ЭТДП») не должны применяться.

8.12.2 Блок данных

Содержание и структура данной ЭТДП определяется изготовителем и выходит за рамки настоящего стандарта.

8.12.2.1 Поле «Идентификатор ЭТДП»

Идентификатор ЭТДП должен соответствовать структуре, определенной в 8.3.

Данное поле является обязательным. Если данное поле опущено или содержит недопустимые значения, то СПП должен выдать сообщение о фатальной ошибке ЭТДП.

8.12.2.2 Содержимое блока данных

Содержание и структура данного поля контролируются изготовителем.

8.13 ЭТДП физического уровня

ЭТДП физического уровня является обязательной ЭТДП. Функция ЭТДП физического уровня — сделать доступной в интерфейсе всю информацию, необходимую для получения доступа к любому каналу, а также информацию, общую для всех каналов. Байты ЭТДП физического уровня постоянны и предназначены только для чтения.

Данная ЭТДП не описана далее в настоящем стандарте. Детали данной ЭТДП приведены в другом стандарте комплекса стандартов ИИЭР 1451.

8.13.1 Доступ

Доступ к ЭТДП физического уровня осуществляется с помощью команд «Query TEDS» («Запросить ЭТДП»), «Read TEDS segment» («Считать сегмент ЭТДП»), «Write TEDS segment» («Записать сегмент ЭТДП») или «Update TEDS» («Обновить ЭТДП»). Аргумент команды должен определять код доступа к ЭТДП физического уровня в соответствии с таблицей 17.

Данная ЭТДП должна быть реализована как ЭТДП только для чтения, чтобы предотвратить внесение изменений в ее области, так как любые изменения могут привести к непредсказуемому поведению преобразователя. Если данная ЭТДП реализована как ЭТДП только для чтения, то команда ИМП «Write TEDS segment» («Записать сегмент ЭТДП») и команда ИМП «Update TEDS» («Обновить ЭТДП») не должны применяться.

8.13.2 Блок данных

Содержание и структура данной ЭТДП определена в другом стандарте комплекса стандартов ИИЭР 1451 и выходит за рамки рассмотрения настоящего стандарта.

8.13.2.1 Поле «Идентификатор ЭТДП»

Идентификатор ЭТДП должен соответствовать структуре, определенной в 8.3.

Данное поле является обязательным. Если данное поле отсутствует или содержит недопустимые значения, то СПП должен выдать сообщение о фатальной ошибке ЭТДП.

8.13.2.2 Содержимое блока данных

Содержание и структура данного поля контролируются другими стандартами комплекса стандартов ИИЭР 1451.

9 Введение в прикладной программный интерфейс (API) уровня ИИЭР 1451.0

В настоящем стандарте рассмотрено два вида API. Данный раздел посвящен основным положениям API. В разделе 10 рассмотрен сервисный интерфейс преобразователя, который является исключительно API СПП и используется для обеспечения доступа осуществляющих измерение и контроль приложений к уровню ИИЭР 1451.0. Данный вид API содержит методы чтения и записи каналов преобразователя, чтения и записи ЭТДП, посылки команд конфигурации, контрольных команд и рабочих команд для модулей ИМП. Дополнительно интерфейс может быть также определен для применения приложением в целях обеспечения неблокирующих операций чтения и записи и для получения данных потоков измерения.

Такие определения API относятся к системам, имеющим видимые интерфейсы. Для единых модулей ИМП и единых процессоров СПП, то есть СПП и ИМП с единым набором аппаратного и программного обеспечения без учета отличительных особенностей отдельных интерфейсов ИИЭР 1451.0 и ИИЭР 1451.X, API не является обязательным до тех пор, пока сообщения видимых интерфейсов согласованы с правилами остальной части настоящего стандарта. Задача таких API заключается в об-

легчении модульного принципа проектирования до такой степени, чтобы различные поставщики могли закладывать различные функциональные возможности и при этом иметь возможность простого интегрирования различных частей по «бесшовной» технологии.

Интерфейс связи модулей служит для взаимодействия между устройствами данного стандарта и другого стандарта комплекса ИИЭР 1451. Это симметричный интерфейс, который реализуется как на стороне СПП, так и на стороне ИМП. Данный API содержит методы, которые реализуются устройствами уровня ИИЭР 1451.X и вызываются настоящим стандартом, чтобы инициировать операции связи. Аналогичным образом данный API содержит методы, которые реализуются устройствами данного стандарта и которые вызываются уровнем ИИЭР 1451.X для доставки средств связи.

Взаимодействие между интерфейсами и другими структурными элементами комплекса стандартов ИИЭР 1451 представлено в виде базовой модели на рисунках 1 и 2.

9.1 Задачи API

Основные задачи, решаемые за счет использования данных API, следующие:

- обеспечение интерфейсов API, хорошо согласованных с требованиями систем измерения, построенных на базе стандарта ИИЭР 1451 и состоящих из СПП и ИМП;
- обеспечение интерфейсов API, которые упрощают взаимодействие между приложениями СПП и ИМП, осуществляющими измерение и контроль. Основные сервисы следующие:
  1) обнаружение ИМП,
  2) доступ к преобразователю,
  3) управление преобразователем,
  4) управление ЭТДП;
- обеспечение абстрагирования связи, то есть ее независимости от технологий нижележащих уровней ИИЭР 1451.X стандарта;
- приспособление широкого ряда известных исходных технологий по стандарту ИИЭР 1451.X и разрешение группам устройств ИИЭР 1451.X использовать наиболее подходящие механизмы связи;
- приспособление широкого ряда известных процессоров и блоков оперативной памяти, использующихся в СПП и ИМП, начиная от персональных компьютеров, работающих как СПП, и заканчивая простейшими 8-битными микропроцессорами, подобными программируемым контроллерам прерываний;
- обеспечение механизмов перехода, когда уровень ИИЭР 1451.X может перехватывать процедуру запуска связи и таким образом задействовать альтернативный способ осуществления сетевых операций;
- обеспечение механизма «транзита данных», когда «интеллектуальные приложения» могут послать пользовательскую команду через уровни ИИЭР 1451.0 для исполнения местными или удаленными устройствами уровней ИИЭР 1451.X;
- обеспечение механизма перехода, когда приложения могут посылать собственные команды типа «напрямую, насквозь» к определенной приложением стороне ИМП без обработки подсистемами уровней ИИЭР 1451.0 или ИИЭР 1451.X.

Интерфейс сервисов, описанных в данном разделе, представлен в виде операций. Подписи операций представлены с использованием варианта языка определения интерфейса (IDL), определенного в ИСО/МЭК 14750:1999. Изменения заключаются в следующем:

- подпись описанной в настоящем разделе операции на языке IDL должна использовать только типы данных, определенные в данном стандарте;
- все спецификации в рамках данного раздела начинаются с надписи «**IDL:**», выделенной жирным шрифтом для облегчения автоматического выделения из электронной копии настоящего стандарта.

9.2 Проектные решения API

9.2.1 Описание API с помощью IDL и текста

Для решения задачи независимости от языка описания IDL будет использоваться для описания функций, параметров и результатов API. Сопроводительное текстовое описание будет использоваться, чтобы оперировать семантикой запросов.

Рисунок 21 иллюстрирует верхний уровень ИИЭР 1451.0 структуры API. Данная структура определяет модуль «IEEE1451Dot0» («ИИЭР1451Точка0»), который является модулем IDL верхнего уровня. В таблице 75 определены «вложенные модули».

Таблица 75 — Модули в API

Модуль	Описание
Сервисы преобразователя	Это открытый API, используемый приложениями для измерения и контроля для взаимодействия с уровнем ИИЭР 1451.0. Он содержит классы и интерфейсы для обнаружения зарегистрированных ИМП; доступа к каналам преобразователя для выполнения измерений или записи данных исполнительных устройств; управления доступом к ИМП; чтения и записи ЭТДП
Модульные связи	Это API, который используется реализуемыми на уровне ИИЭР 1451.X средствами и устройствами для обеспечения связей от СПП к ИМП. В нем определены как «двухточечные» («линейные»), так и «сетевые» интерфейсы, а также обратные сообщения
Аргументы	Данный блок содержит все аргументы уровня ИИЭР 1451.0. Здесь определяется структура данных типа массивов аргументов «ArgumentArray»
Утилиты	Данный блок содержит классы утилит и интерфейсы для конвертирования массивов аргументов («ArgumentArrays») в байтовые массивы («OctetArrays») и наоборот. Конвертирование осуществляется посредством интерфейса кодеков. Реализуемые на уровне ИИЭР 1451.X средства и устройства, требующие кодирования, регистрируют альтернативные кодеки

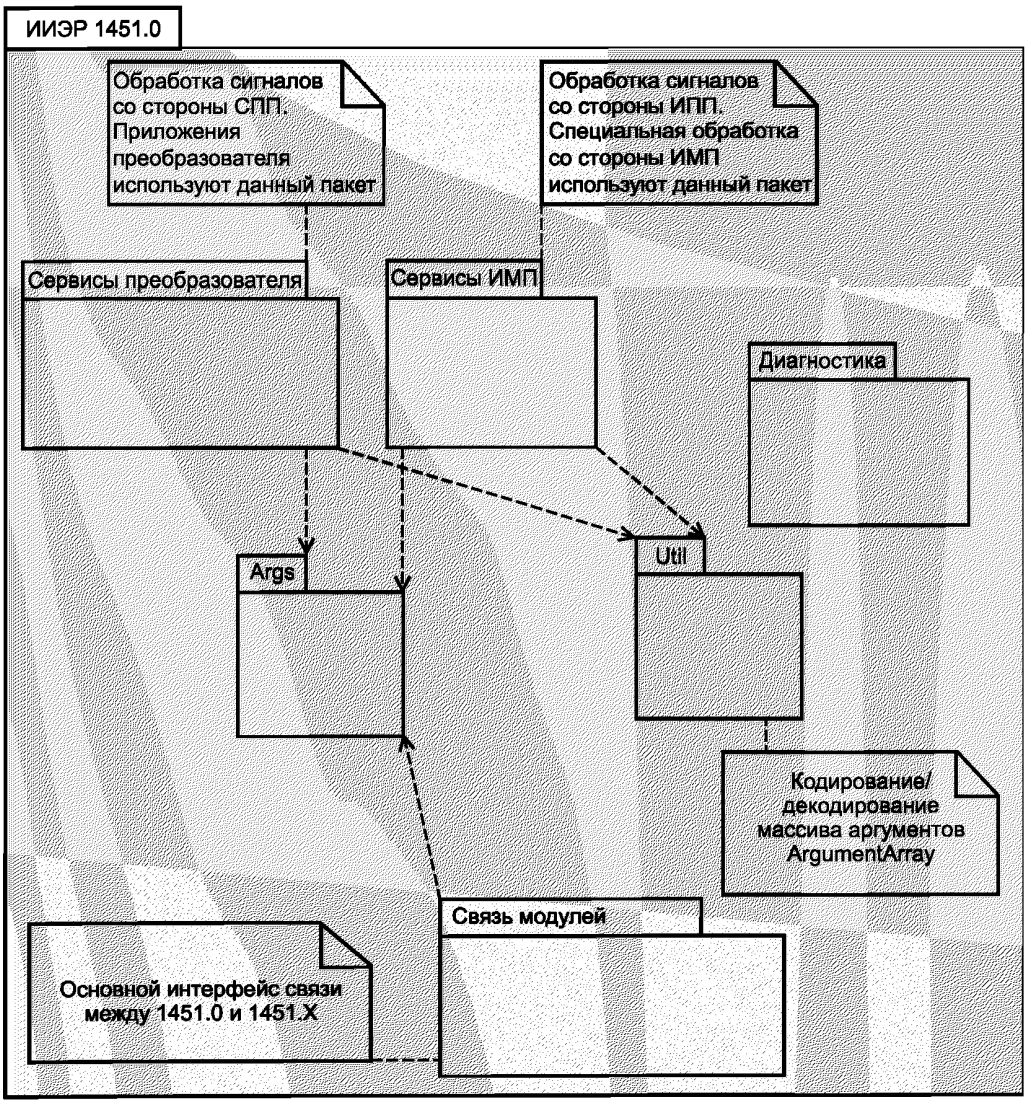


Рисунок 21 — Структура API

9.2.2 Байтовый массив полезной нагрузки ИИЭР 1451.0

Чтобы минимизировать сведения о необходимых для работы с уровнем ИИЭР 1451.X данных, вся (с точки зрения настоящего стандарта) передаваемая информация объединяется вместе в «полезную нагрузку» («payload»), которая кодируется в виде байтового массива данных.

За исключением случаев, когда уровню ИИЭР 1451.X необходимо перехватить сообщение уровня ИИЭР 1451.0, устройствам уровня ИИЭР 1451.X следует рассматривать полезную нагрузку как «невоспринимаемую». В 6.2—6.3 приводятся данные о месте нахождения информации о длине, необходимой для декодирования данных, что осуществляется внутри класса кодирования/декодирования.

С точки зрения настоящего стандарта байтовый массив и адресация получателя представляют собой логический формат соединения. Предполагается, что уровень ИИЭР 1451.X осуществляет упаковку байтового массива в соответствующий сетевой пакет для данной технологии уровня ИИЭР 1451.X. Например, могут быть добавлены соответствующие сетевые заголовки и контрольные суммы. Кроме того, уровень ИИЭР 1451.X несет ответственность за деление байтового массива на сетевые пакеты соответствующего размера и их обратное объединение в байтовый массив на удаленном узле. Аналогичным образом уровень ИИЭР 1451.X несет ответственность за шифрование, аутентификацию, сжатие и управление обменом данными.

9.2.3 Структуры данных уровня ИИЭР 1451.0 на языке IDL. Использование массива аргументов «ArgumentArray»

Чтобы упростить использование полезной нагрузки для уровня ИИЭР 1451.0 и более высоких уровней приложений, выполняющих измерения, данный стандарт также определяет структуры данных на языке IDL. В конкретных реализациях IDL структуры данных отображаются в виде соответствующих структур данных, зависящих от используемого в реализуемом приложении языка программирования (например, структуры на языке C или C++ или класс Java). Такие зависящие от языка программирования структуры данных гарантируют корректное совмещение байтов, что позволяет осуществить непосредственный доступ ко всем атрибутам данных. Например, при использовании большинства платформ потребуется совмещение чисел с плавающей точкой с границами четырехбайтовой памяти.

Согласно примеру стандарта ИИЭР 1451.1—1999 наиболее распространенной структурой данных в рамках уровня ИИЭР 1451.0 и уровней приложений является универсальный массив аргументов «ArgumentArray». Аргументы могут быть представлены всеми основными типами данных (например, UInt8, UInt16 или Float32) и простейшими массивами данных основных типов (например, UInt8Array, Float32Array или StringArray). Дополнительно могут быть определены нестандартные аргументы, необходимые для приложений, осуществляющих измерения (например, Units, TimeInstance или TimeDuration).

Структура данных «ArgumentArray» в виде массива аргументов является очень гибким механизмом для компоновки и передачи произвольных типов данных в системе, не требуя сведений о времени компиляции. Например, отсчет при измерении с канала преобразователя кодируется в соответствующий аргумент, базируясь на информации, представленной в ЭТДП (например, модель данных, число повторений данных, калибровочная информация). Данный аргумент будет храниться внутри массива данных длиной 1 байт. В качестве более сложного примера можно рассмотреть процесс считывания с прокси-канала преобразователя, определяющего группу каналов преобразователя, которые должны считываться вместе. Каждый канал преобразователя такого прокси будет представлен в виде отдельного аргумента необходимого типа, основываясь на информации ЭТДП. Данные аргументы группируются вместе в массив аргументов «ArgumentArray», который представляет данные всего прокси.

Для упрощения процесса передачи массивов аргументов через уровень ИИЭР 1451.X обеспечен универсальный механизм кодирования/декодирования посредством интерфейса кодеков. Данные методы кодируют массивы аргументов в байтовые массивы и наоборот. В большинстве случаев ответственность за вызов данных методов возлагается на настоящий стандарт, а уровень ИИЭР 1451.X работает исключительно с формами байтовых массивов. При этом устройства уровня ИИЭР 1451.X должны обеспечить лишь передачу полезной нагрузки в виде байтовых массивов от отправителя к получателю. Операции кодирования/декодирования осуществляются через библиотеки, однако при необходимости уровень ИИЭР 1451.X может использовать другую реализацию. Более подробная информация представлена в разделах 7—8.

В редких случаях уровню ИИЭР 1451.X может потребоваться отклониться от рекомендуемого для уровня ИИЭР 1451.0 формата байтового массива данных. В этом случае уровню ИИЭР 1451.X следует обеспечить другую библиотеку кодирования/декодирования.

9.2.4 Допущения о принадлежности параметров

Если не определено иначе, то параметры и возвращенные результаты принадлежат объекту — отправителю запроса. Если объекту, к которому осуществлен запрос, необходимо удержать значение параметра, то он должен сделать локальную копию. Объект — отправитель запроса имеет право освободить любые ресурсы памяти для параметров и возвращенных результатов после возвращения запроса.

9.3 Модуль «IEEE1451Dot0»

IDL: `module IEEE1451Dot0 { }.`

Все интерфейсы ИИЭР 1451.0 находятся внутри модуля IDL «IEEE1451Dot0».

9.3.1 Модуль «IEEE1451Dot0::Args»

IDL: `module Args { }.`

Внутри модуля «Args» содержатся основные и специальные типы данных. Основные типы данных, а также типы данных, используемые в других частях настоящего стандарта, определены в разделе 4.

Взаимодействие между типами данных, а также классами «Argument» и «ArgumentArray» представлено на рисунке 22.

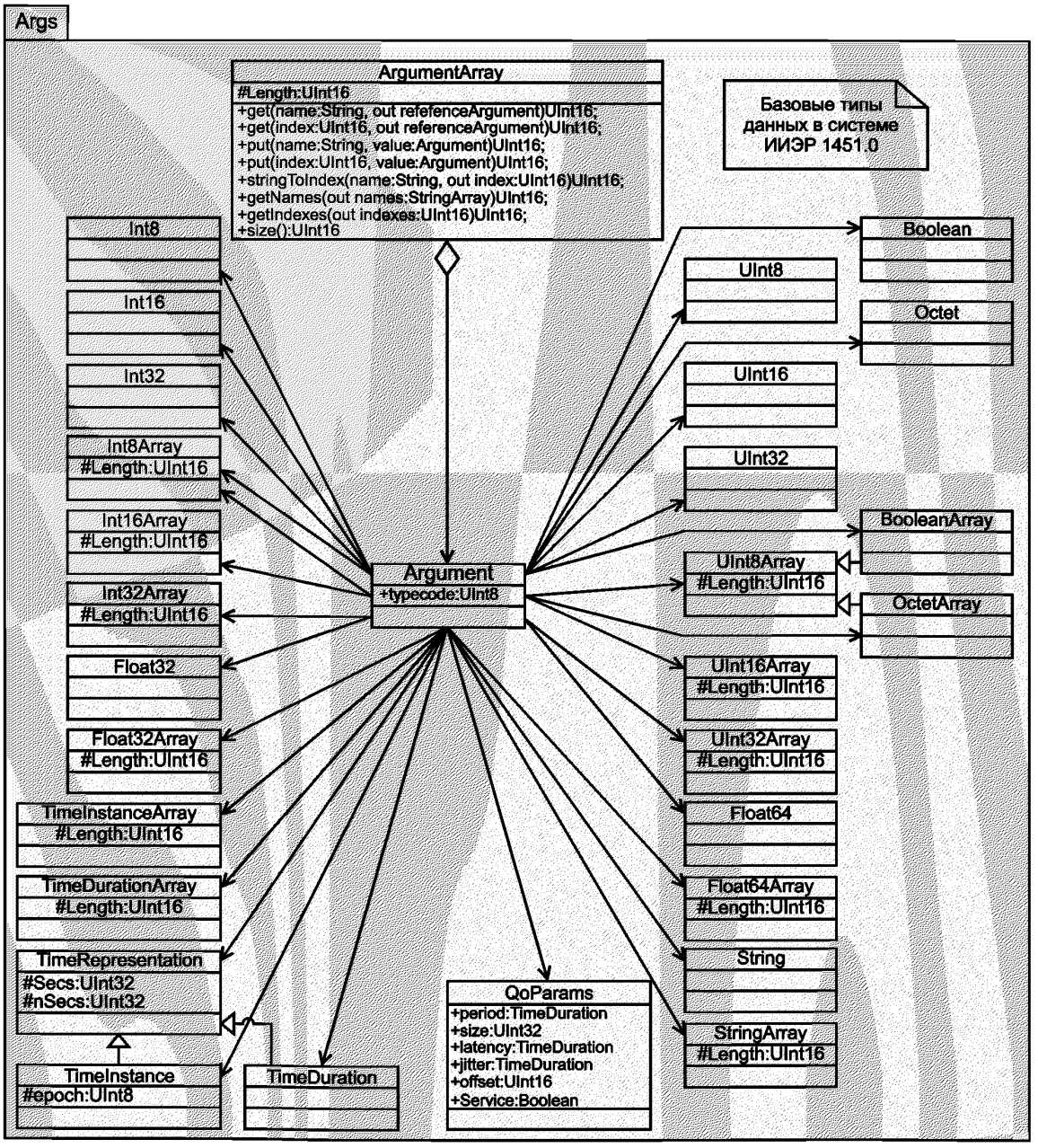


Рисунок 22 — Аргументы

9.3.1.1 Массивы основных типов

Массивы основных типов, имеющие переменную длину, задаются как последовательности IDL. Типы данных в виде массива содержат как длину данных, так и сами данные. Длина представлена в виде величины UInt16.

IDL:
```
typedef sequence<Int8>      Int8Array;
typedef sequence<Int16>     Int16Array;
typedef sequence<Int32>     Int32Array;
typedef sequence<UInt8>     UInt8Array;
typedef sequence<UInt16>    UInt16Array;
typedef sequence<UInt32>    UInt32Array;
typedef sequence<Float32>   Float32Array;
typedef sequence<Float64>   Float64Array;
typedef sequence<_String>   StringArray;
typedef sequence<_Octet>    OctetArray;
typedef sequence<_Boolean>  BooleanArray;
typedef sequence<TimeInstance>  TimeInstanceArray;
typedef sequence<TimeDuration>  TimeDurationArray.
```

9.3.1.2 Коды ошибок

Все коды ошибок представляются в виде величин UInt16. В осуществлении операции связи участвуют пять составляющих: местный уровень ИИЭР 1451.0, местный уровень ИИЭР 1451.X, удаленный уровень ИИЭР 1451.X, удаленный уровень ИИЭР 1451.0 и удаленный уровень приложения. Источник кода ошибки кодируется в трех верхних битах. Нумерация кодов ошибок кодируется в нижних битах. Биты нумеруются со старшего значащего бита до младшего значащего бита, с бита 15 до бита 0, как показано в таблице 76.

Таблица 76 — Соответствие битов с кодами ошибок

Биты	Использование
Биты с 15 по 13	Три старших значащих бита: информация об источнике кода ошибки кодируется в данных битах в соответствии с описанием в таблице 77
Биты с 12 по 0	Нумерация кодов ошибок; см. таблицу 78

Значения источников кода ошибки определены в таблице 77. Графа значений содержит биты с 15 по 13 и представляет трехбитное целое число без знака.

Таблица 77 — Нумерация источников кода ошибки

Значение	Источник ошибки
0	Ошибка местного уровня ИИЭР 1451.0
1	Ошибка местного уровня ИИЭР 1451.X
2	Ошибка удаленного уровня ИИЭР 1451.X
3	Ошибка удаленного уровня ИИЭР 1451.0
4	Ошибка удаленного уровня приложения
5	Зарезервировано
6	Зарезервировано
7	Открыто для изготовителей

Значения кодов ошибок определены в таблице 78. В данном случае биты с 12 по 0 представляют 13-битное целое число без знака.

Таблица 78 — Нумерация источников кода ошибки

Нумерация	Название кода ошибки	Описание
0	NO_ERROR	Ошибок нет, операция выполнена успешно
1	INVALID_COMMID	Недействительный «commId»
2	UNKNOWN_DESTID	Неизвестный «destId»
3	TIMEOUT	Превышено время ожидания (тайм-аут) операции
4	NETWORK_FAILURE	Получатель недоступен, ошибка сети
5	NETWORK_CORRUPTION	Связь нарушена, ошибка сети
6	MEMORY	Местная ошибка, связанная с нехваткой памяти
7	QOS_FAILURE	Нарушение качества обслуживания сети
8	MCAST_NOT_SUPPORTED	Многоадресная передача не поддерживается или недействительная операция для многоадресной передачи
9	UNKNOWN_GROUPID	Неизвестный «groupId»
10	UNKNOWN_MODULEID	Неизвестный «moduleId»
11	UNKNOWN_MSGID	Неизвестный «msgId»
12	NOT_GROUP_MEMBER	«destId» не состоит в группе
13	ILLEGAL_MODE	Параметр режима работы недействителен
14	LOCKED_RESOURCE	Доступ к запрашиваемому ресурсу закрыт
15	FATAL_TEDS_ERROR	Ошибка в ЭТДП приводит к неработоспособности устройства
16	NON-FATAL_TEDS_ERROR	Значение поля ЭТДП не может быть использовано, но устройство продолжает функционировать
17	CLOSE_ON_LOCKED_RESOURCE	Предупреждающий код ошибки, возвращение которого сигнализирует об осуществлении операции завершения на закрытом для доступа ресурсе
18	LOCK_BROKEN	Обратное сообщение содержит данный код ошибки, если в системе выполняются операции неблокирующего считывания/записи или обработки потока измерений
19	NETWORK_RESOURCE_EXCEEDED	Уровень ИИЭР 1451.X достиг ограничений сетевого ресурса
20	MEMORY_RESOURCE_EXCEEDED	Уровень ИИЭР 1451.X достиг ограничений памяти ресурса
21—4095	Зарезервировано	
4096—8191	Открыто для изготовителей	

9.3.1.3 Структура данных «IEEE1451Dot0::Args::QosParam»

Атрибуты «Quality of Service» («Качество сервиса») организованы как структура данных для повышения эффективности. Структура «QoSParams» содержит информацию, представленную в таблице 79.

IDL:
```
struct QoSParam {
    _Boolean        service;
    TimeDuration    period;
    UInt16          transmitSize;
    UInt16          replySize;
    TimeDuration    accessLatency;
    TimeDuration    transmitLatency;
 }.
```

Таблица 79 — Описание структуры «QoSParams»

Параметр	Тип	Описание
Service	Логический	Значение «True» («Истина») указывает на гарантированное качество услуг передачи данных (QoS), а значение «False» (Ложь») служит признаком низкоприоритетного сетевого трафика
Period	TimeDuration	Показывает период связи. В условиях непериодической передачи данных следует использовать нулевое значение
transmitSize	UInt32	Данный параметр назначает число байтов, которые могут быть переданы для каждого периода связи
replySize	UInt32	В условиях двухсторонней связи данный параметр назначает число байтов, посылаемых в качестве ответа для каждого периода связи. Нулевое значение данного параметра сигнализирует об односторонней связи
accessLatency	TimeDuration	Данный параметр назначает дополнительное время задержки, которое уровень ИИЭР 1451.0 допускает для начала передачи данных уровнем ИИЭР 1451.X, прежде чем данному уровню ИИЭР 1451.X следует сообщить о нарушениях связи. Нулевое значение должно интерпретироваться как то, что значение задержки доступа не задано
transmitLatency	TimeDuration	Данный параметр назначает дополнительное время задержки, которое уровень ИИЭР 1451.0 допускает для завершения передачи данных уровнем ИИЭР 1451.X, прежде чем данному уровню ИИЭР 1451.X следует сообщить о нарушениях связи. Нулевое значение должно интерпретироваться как то, что значение задержки передачи данных не задано

9.3.1.4 Структура данных «IEEE1451Dot0::Args::TypeCode»

Каждый действительный тип массива аргументов «ArgumentArray» уровня ИИЭР 1451.0 имеет уникальный типокод (код типа).

IDL:
```
enum TypeCode {
    UNKNOWN_TC,
   // Simple types
    UInt8_TC, UINT16_TC, UINT32_TC,
    FLOAT32_TC, FLOAT64TC, STRING_TC,
    OCTET_TC, BOOLEAN_TC,
    TIME_INSTANCE_TC, TIME_DURATION_TC,
    QOS_PARAMS_TC,
   // Arrays of simple types. Note no QOS array
    UInt8_ARRAY_TC, UINT16_ARRAY_TC, UINT32_ARRAY_TC,
    FLOAT32_ARRAY_TC, FLOAT64_ARRAY_TC, STRING_ARRAY_TC,
    OCTET_ARRAY_TC, BOOLEAN_ARRAY_TC,
     TIME_INSTANCE_ARRAY_TC, TIME_DURATION_ARRAY_TC
 }.
```

9.3.1.5 Структура данных «IEEE1451Dot0::Args::Argument»

Данная структура представляет собой универсальный контейнер данных. Он представлен как IDL размеченное объединение данных. Тем не менее реализации с использованием языков программирования с динамической проверкой типов могут иметь более простые способы задания.

IDL:
```
union Argument switch (TypeCode) {
case UNKNOWN_TC:            Boolean        valueError;
case UInt8_TC:              UInt8          valueInt8;
case UINT16_TC:             UInt16         valueUInt16;
case UINT32_TC:             UInt32         valueUInt32;
case FLOAT32_TC:            Float32        valueFloat32;
case FLOAT64_TC:            Float64        valueFloat64;
case STRING_TC:             _String        valueString;
case OCTET_TC:              _Octet         valueOctet;
case BOOLEAN_TC:            _Boolean       valueBoolean;
case TIME_INSTANCE_TC:      TimeInstance   valueTimeInstance;
```

```
case TIME_DURATION_TC:        TimeDuration    valueTimeDuration;
case QOS_PARAMS_TC:           QosParams       valueQosParams;
case UInt8_ARRAY_TC:          UInt8Array      valueInt8Array;
case UINT16_ARRAY_TC:         UInt16Array     valueUInt16Array;
case UINT32_ARRAY_TC:         UInt32Array     valueUInt32Array;
case FLOAT32_ARRAY_TC:        Float32Array    valueFloat32Array;
case FLOAT64_ARRAY_TC:        Float64Array    valueFloat64Array;
case STRING_ARRAY_TC:         StringArray     valueStringArray;
case OCTET_ARRAY_TC:          OctetArray      valueOctetArray;
case BOOLEAN_ARRAY_TC:        BooleanArray    valueBooleanArray;
case TIME_INSTANCE_ARRAY_TC: TimeInstanceArray
                                        valueTimeInstanceArray;
case TIME_DURATION_ARRAY_TC: TimeDurationArray
                                      valueTimeDurationArray;
}.
```

9.3.1.6 Структура данных «IEEE1451Dot0::Args::ArgumentArray»

IDL: `interface ArgumentArray { }.`

Данная структура данных представляет собой универсальный контейнер массивов данных. Все аргументы принадлежат массиву аргументов «ArgumentArray». При удалении массива аргументов «ArgumentArray» высвобождается вся память, занимаемая аргументами. В таблице 80 перечислены методы, связанные с массивами аргументов «ArgumentArrays».

Таблица 80 — Массив аргументов «ArgumentArray»

IEEE1451dot0::Args::ArgumentArray
UInt16 getByName(in _String name, out Argument reference);
UInt16 getByIndex(in UInt16 index, out Argument reference);
UInt16 putByName(in _String name, in Argument value);
UInt16 putByIndex(in UInt16 index, in Argument value);
UInt16 stringToIndex(in String name, out UInt16 index);
UInt16 getNames(out StringArray names);
UInt16 getIndexes(out UInt16Array indexes);
UInt16 size();

9.3.1.7 Метод «IEEE1451Dot0::Args::ArgumentArray::get»

IDL: `UInt16 getByName ( in _String name, out Argument reference).`

Данный метод обеспечивает функциональную возможность поиска по имени. Соответствующие названия атрибутов описаны в разделах 7—8. Метод возвращает ссылку на заданный аргумент. Необходимо обратить внимание, что отправителю запроса следует рассматривать данную ссылку как ссылку «только для чтения». При необходимости отправитель запроса должен сделать локальную копию.

Параметры

Параметр «name» — имя заданного атрибута.

Параметр [out] «reference» — ссылка на заданный аргумент.

Возвращаемый результат: код ошибки.

9.3.1.8 Метод «IEEE1451Dot0::Args::ArgumentArray::get»

IDL: `UInt16 getByIndex( in UInt16 index, out Argument reference).`

Данный метод обеспечивает функциональную возможность поиска по индексу. Соответствующие переходы между именами и индексами описаны в разделах 7—8. Метод возвращает ссылку на заданный аргумент. Необходимо обратить внимание, что отправителю запроса следует рассматривать данную ссылку как ссылку «только для чтения». Отправитель запроса при необходимости должен сделать локальную копию.

Параметры

Параметр «index» — индекс массива, начинающийся с нуля.

Параметр [out] «reference» — ссылка на заданный аргумент.

Возвращаемый результат: код ошибки.

9.3.1.9 Метод «IEEE1451Dot0::Args::ArgumentArray::put»

IDL: `UInt16 putByName( in String name, in Argument value).`

Данный метод обеспечивает функциональную возможность присвоения значения по имени. Соответствующие переходы между именами и индексами описаны в разделах 7—8. Массив аргументов «ArgumentArray» присваивает заданный аргумент. Отправитель запроса не должен освобождать память, связанную с этим аргументом. Если массив аргументов «ArgumentArray» уже содержит аргумент с таким же именем, то он будет удален.

Параметры

Параметр «name» — имя заданного атрибута.

Параметр «value» — значение аргумента, которое будет помещено в массив аргументов «ArgumentArray».

Возвращаемый результат: код ошибки.

9.3.1.10 Метод «IEEE1451Dot0::Args::ArgumentArray::put»

IDL: `UInt16 putByIndex( in UInt16 index, in Argument value).`

Данный метод обеспечивает функциональную возможность присвоения значения по индексу. Соответствующие переходы между именами и индексами описаны в разделах 7—8. Массив аргументов «ArgumentArray» присваивает заданный аргумент. Отправитель запроса не должен освобождать память, связанную с этим аргументом. Если массив аргументов «ArgumentArray» уже содержит аргумент с таким же индексом, то он будет удален.

Параметры

Параметр «index» — индекс массива, начинающийся с нуля.

Параметр «value» — значение аргумента, которое будет помещено в массив аргументов «ArgumentArray».

Возвращаемый результат: код ошибки.

9.3.1.11 Метод «IEEE1451Dot0::Args::ArgumentArray::stringToIndex»

IDL: `UInt16 stringToIndex( in _String name, out UInt16 index).`

Данный метод обеспечивает механизм преобразования имен в индекс массива. Обычно осуществление поиска по индексу является более эффективным по сравнению с поиском по имени. Данный метод позволяет приложению осуществлять первоначальные операции поиска в строке и сравнения и последующее использование индекса для доступа к массиву.

Параметры

Параметр «name» — имя заданного атрибута.

Параметр [out] «index» — индекс массива, начинающийся с нуля.

Возвращаемый результат: код ошибки.

9.3.1.12 Метод «IEEE1451Dot0::Args::ArgumentArray::getNames»

IDL: `UInt16 getNames( out StringArray names).`

Данный метод возвращает строковый массив «имен» для каждого элемента массива аргументов «ArgumentArray». Каждый из них может быть использован в методе «get( )».

Параметры

Параметр [out] «name» — строковый массив.

Возвращаемый результат: код ошибки.

9.3.1.13 Метод «IEEE1451Dot0::Args::ArgumentArray::getIndexes»

IDL: `UInt16 getIndexes( out UInt16Array indexes).`

Данный метод возвращает массив «индексов» в виде массива 16-разрядных целых чисел без знака для каждого элемента массива аргументов «ArgumentArray». Каждый из них может быть использован в методе «get( )».

Параметры

Параметр [out] «index» — массив 16-разрядных целых чисел без знака (UInt16Arrary).

Возвращаемый результат: код ошибки.

9.3.1.14 Метод «IEEE1451Dot0::Args::ArgumentArray::size»

IDL: `UInt16 size().`

Данный метод возвращает число элементов массива аргументов «ArgumentArray».

Возвращаемый результат: Число элементов.

9.3.2 Модуль «IEEE1451Dot0::Util»

IDL: `module Util { }.`

В данном модуле организованы классы и интерфейсы утилит.

9.3.2.1 Интерфейс «IEEE1451Dot0::Util::Codec»

IDL: `interface Codec { }.`

Данный интерфейс представлен уровнем ИИЭР 1451.X как дополнительный для обеспечения пользовательского кодирования и декодирования массивов аргументов в байтовые массивы и наоборот. Если интерфейс зарегистрирован, то он будет автоматически запускаться уровнем ИИЭР 1451.0. Список интерфейсов в данной группе представлен в таблице 81.

Таблица 81 — Кодеки

IEEE1451dot0::Util::Codec
Args::UInt16 encodeCommand(in Args::UInt16 channelId, in Args::UInt8 cmdClassId, in Args::UInt8 cmdFunctionId, in Args::ArgumentArray inArgs, out Args::OctetArray payload);
Args::UInt16 decodeCommand(in Args::OctetArray payload, out Args::UInt16 channelId, out Args::UInt8 cmdClassId, out Args::UInt8 cmdFunctionId, out Args::ArgumentArray inArgs);
Args::UInt16 encodeResponse(in Args::_Boolean successFlag, in Args::ArgumentArray outArgs, out Args::OctetArray payload);
Args::UInt16 decodeResponse(in Args::OctetArray payload, out Args::_Boolean successFlag, out Args::ArgumentArray outArgs);
Args::UInt16 argumentArray2OctetArray(in Args::ArgumentArray inArgs, out Args::OctetArray payload);
Args::UInt16 octetArray2ArgumentArray(in Args::OctetArray payload, out Args::ArgumentArray outArgs);

9.3.2.2 Метод «IEEE1451Dot0::Util::Codec::encodeCommand»

IDL:
```
Args::UInt16 encodeCommand(
  in  Args::UInt16        destId,
  in  Args::UInt16         channelId,
  in  Args::UInt8          cmdClassId,
  in  Args::UInt8          cmdFunctionId,
  in  Args::ArgumentArray  inArgs,
  out Args::OctetArray     payload).
```

Данный метод обеспечивается уровнем ИИЭР 1451.X и вызывается уровнем ИИЭР 1451.0 для кодирования команды в байтовый массив. Подробное описание соглашения о кодировании представлено в разделе 6. Метод вызывается на инициирующем узле до запуска запросов «ModuleConnunication::P2 PComm::write( )» или «ModuleConnunication::NetComm::writeMsg( )».

Планируется, что уровень «dot 0» считывает и использует командные ЭТДП для кодирования/декодирования неизвестных команд.

Параметры

Параметр «channelId» — заданный идентификатор канала.

«cmdClassId» — заданный класс команды.

«cmdFunctionId» — заданный код функции команды.

Параметр «inArgs» содержит специальные входные аргументы команды.

[out] «payload» — закодированный байтовый массив.

Возвращаемый результат: код ошибки.

9.3.2.3 Метод «IEEE1451Dot0::Util::Codec::decodeCommand»

IDL:
```
Args::UInt16 decodeCommand(
  in  Args::OctetArray    payload,
  out Args::UInt16        channelId,
  out Args::UInt8         cmdClassId,
  out Args::UInt8         cmdFunctionId,
  out Args::ArgumentArray inArgs).
```

Данный метод обеспечивается уровнем ИИЭР 1451.X и вызывается уровнем ИИЭР 1451.0 для декодирования байтового массива в аргументы команды. Подробное описание соглашения о кодировании представлено в разделе 6. Метод вызывается на принимающем узле после запуска запросов «ModuleCommunication::P2PComm::read( )» или «ModuleCommunication::NetComm::readMsg( )».

Параметры

«payload» — байтовый массив.

Параметр [out] «channelId» — заданный идентификатор канала.

[out] «cmdClassId» — заданный класс команды.

[out] «cmdFunctionId» — заданный код функции команды.

Параметр [out] «inArgs» содержит специальные входные аргументы команды.

Возвращаемый результат: код ошибки.

9.3.2.4 Метод «IEEE1451Dot0::Util::Codec::encodeResponse»

IDL:
```
Args::UInt16 encodeResponse(
 in  Args::_Boolean         successFlag,
 in  Args::ArgumentArray    outArgs,
 out Args::OctetArray       payload).
```

Данный метод обеспечивается уровнем ИИЭР 1451.X и вызывается уровнем ИИЭР 1451.0 для кодирования ответного сообщения в байтовый массив. Подробное описание соглашения о кодировании представлено в разделах 7—8. Метод вызывается на принимающем узле до запуска запросов «ModuleConnunication::P2PComm::write( )» или «ModuleConnunication::NetComm::writeRsp( )».

Параметры

Параметр «successFlag» — заданный код успешного завершения операции.

Параметры «outArgs» — специальные выходные аргументы ответного сообщения на команду.

[out] «payload» — кодированный байтовый массив.

Возвращаемый результат: код ошибки.

9.3.2.5 Метод «IEEE1451Dot0::Util::Codec::decodeResponse»

IDL:
```
Args::UInt16 decodeResponse(
  in  Args::OctetArray      payload,
  out Args::_Boolean        successFlag,
  out Args::ArgumentArray   outArgs).
```

Данный метод обеспечивается уровнем ИИЭР 1451.X и вызывается уровнем ИИЭР 1451.0 для декодирования байтового массива в аргументы ответного сообщения. Подробное описание соглашения о кодировании представлено в разделах 7—8. Метод вызывается на инициирующем узле после запуска запросов «ModuleCommunication::P2PComm::read( )» или «ModuleCommunication::NetComm::readMsg( )».

Параметры

«Payload» — байтовый массив.

Параметр [out] «successFlag» — метка успешного выполнения команды.

Параметры [out] «outArgs» — специальные выходные аргументы команды.

Возвращаемый результат: код ошибки.

9.3.2.6 Метод «IEEE1451Dot0::Util::Codec::encodeArgumentArray»

IDL:
```
Args::UInt16 encodeArgumentArray(
  in  Args::ArgumentArray   inArgs,
  out Args::OctetArray      payload).
```

Данный метод обеспечивается уровнем ИИЭР 1451.X и вызывается уровнем ИИЭР 1451.0 для кодирования массива аргументов в байтовый массив. Подробное описание соглашения о кодировании представлено в разделах 7—8. Необходимо убедиться в работоспособности данного метода!

Параметры

Параметр «inArgs» — входной массив аргументов.

Параметр [out] «payload» — кодированный байтовый массив.

Возвращаемый результат: код ошибки.

9.3.2.7 Метод «IEEE1451Dot0::Util::Codec::decodeOctetArray»

IDL:
```
Args::UInt16 decodeOctetArray(
  in  Args::OctetArray      payload,
  out Args::ArgumentArray   outArgs).
```

Данный метод обеспечивается уровнем ИИЭР 1451.X и вызывается уровнем ИИЭР 1451.0 для декодирования байтового массива в массив аргументов. Подробное описание соглашения о кодировании представлено в разделах 7—8.

Параметры

«Payload» — байтовый массив.

Параметр [out] «outArgs» содержит выходной массив аргументов.

Возвращаемый результат: код ошибки.

9.3.2.8 Интерфейс «IEEE1451Dot0::Util::CodecManagement»

IDL: `Interface CodecManagement { }.`

Данный класс обеспечивается уровнем ИИЭР 1451.0 и опционально вызывается уровнем ИИЭР 1451.X для регистрации дополнительных технических реализаций (приложений) кодирования/декодирования с уровнем ИИЭР 1451.0. В таблице 82 перечислены методы, доступные в данном классе.

Таблица 82 — Управление кодеками

IEEE1451dot0::Util::CodecManagement
Args::UInt16 register(in Args::UInt8 moduleId, in Codec customCodec);
Args::UInt16 unregister(in Args::UInt8 moduleId);

9.3.2.9 Метод «IEEE1451Dot0::Util::CodecManagement::register»

IDL:
```
Args::UInt16 register(
  in Args::UInt8    moduleId,
  in Codec          customCodec).
```

Данный метод обеспечивается уровнем ИИЭР 1451.0 и опционально вызывается уровнем ИИЭР 1451.X для регистрации пользовательской копии кодера/декодера. Для каждого модуля может быть зарегистрирована только одна копия.

Параметры

Параметр «moduleIds» — заданный идентификатор ИИЭР 1451.X.

Параметр «customCodec» — копия кодека для регистрации. Данную копию не следует уничтожать до тех пор, пока не будет сделан запрос «unregister( )».

Возвращаемый результат: код ошибки.

9.3.2.10 Метод «IEEE1451Dot0::Util::CodecManagement::unregister»

IDL:
```
Args::UInt16 unregister(
 in Args::UInt8 moduleId).
```

Данный метод обеспечивается уровнем ИИЭР 1451.0 и опционально вызывается уровнем ИИЭР 1451.X для отмены регистрации пользовательской копии кодера/декодера. После выполнения данного запроса используется кодер/декодер, принятый по умолчанию.

Параметры

Параметр «moduleIds» — заданный идентификатор ИИЭР 1451.X.

Возвращаемый результат: код ошибки.

10 API сервисов преобразователя

IDL: `module TransducerServices { }.`

API сервисов преобразователя обеспечивает интерфейс между приложениями, работающими в СПП, и функциями, определенными в настоящем стандарте.

В данном IDL-модуле описаны все интерфейсы, изначально используемые приложениями для «измерения и контроля».

Модуль «TransducerServices» («Сервисы преобразователя») подразделяется на пять интерфейсов, перечисленных в таблице 83. Первые четыре интерфейса подчиняются настоящему стандарту и вызываются приложением для измерений. Чтобы добавить в приложение дополнительные сервисы, необходимо использование интерфейса «AppCallback», который применяет настоящий стандарт.

Таблица 83 — Классы и интерфейсы API сервисов преобразователя

Интерфейс	Описание
TIMDiscovery	Методы обнаружения приложениями доступных коммуникационных модулей ИИЭР 1451.X, ИМП и каналов преобразователя, объединенных в данном интерфейсе
TransducerAccess	Методы данного интерфейса используются приложением, желающим получить доступ к датчику или исполнительному устройству каналов преобразователя
TransducerManager	Методы данного интерфейса используются при необходимости передачи приложению большего контроля над доступом к ИМП. Например, приложение может заблокировать ИМП для эксклюзивного доступа и посылать ему произвольные команды
TedsManager	Приложения используют методы данного интерфейса для считывания или записи ЭТДП. Данный класс интерфейсов также управляет информацией кеша ЭТДП со стороны СПП
CommManager	Данный интерфейс используется для управления доступом к коммуникационному модулю локального устройства
AppCallback	Данный интерфейс используется для того, чтобы добавить в приложение дополнительные сервисы. Например, интерфейс позволяет приложению конфигурировать потоки данных измерений, а уровню ИИЭР 1451.0 — запускать в приложении соответствующие ответные запросы

10.1 Интерфейс обнаружения ИМП «IEEE1451Dot0::TransducerServices::TIMDiscovery»

IDL: `interface TIMDiscovery { }.`

Интерфейс «TIMDiscovery» («Обнаружение ИМП») обеспечивается уровнем ИИЭР 1451.0 и вызывается приложением для обеспечения общего механизма обнаружения доступных ИМП или каналов преобразователя. Методы интерфейса перечислены в таблице 84 и более детально рассмотрены в 10.1.1—10.1.3.

Таблица 84 — Методы интерфейса «TIMDiscovery» («Обнаружение ИМП»)

IEEE1451dot0::TransducerServices::TIMDiscovery
Args::UInt16 reportCommModule(out Args::UInt8Array moduleIds);
Args::UInt16 reportTims(in Args::UInt8 moduleId, out Args::UInt16Array timIds);
Args::UInt16 reportChannels(in Args::UInt16 timId, out Args::UInt16Array channelIds, out Args::StringArray names);

10.1.1 Метод «IEEE1451Dot0::TransducerServices::TIMDiscovery::reportCommModule»

IDL: `Args::UInt16 reportCommModule( out Args::UInt8Array moduleIds).`

Данный метод сообщает о доступных интерфейсах коммуникационного модуля, зарегистрированных в соответствии с настоящим стандартом. Более подробная информация представлена для метода «IEEE1451Dot0::ModuleCommunication::NetRegistration::registerModule()», который вызывается уровнем ИИЭР 1451.X по достижении готовности к работе. В данном методе уровень ИИЭР 1451.0 назначает уникальный «moduleId» («Идентификатор модуля») для каждого интерфейса ИИЭР 1451.X. Обратите внимание, что СПП может иметь:

- единственный интерфейс ИИЭР 1451.X, соответствующий данной технологии (например, см. раздел 7 стандарта ИИЭР 1451.5-2007 [B4]);
- несколько интерфейсов, соответствующих одной и той же технологии (например, см. COM1 и COM2 по стандарту ИИЭР 1451.2);
- несколько интерфейсов, соответствующих различным технологиям (например, см. раздел 7 стандарта ИИЭР 1451.5-2007 [B4] и ИИЭР 1451.3).

Параметры

Параметр «[out] moduleId» («[выходной] идентификатор модуля») возвращается приложению уровнем ИИЭР 1451.0. Данный массив содержит все известные модули связи СПП.

Возвращаемый результат: код ошибки.

10.1.2 Метод «IEEE1451Dot0::TransducerServices::TIMDiscovery::reportTims»

IDL:
```
Args::UInt16 reportTims(
in Args::UInt8        moduleId,
out Args::UInt16Array timIds).
```

Данный метод возвращает в интерфейс все известные устройства ИМП. Более подробная информация представлена в 11.6.2 для метода «IEEE1451Dot0::ModuleCommunication::Registration::registerDestination», который вызывается уровнем ИИЭР 1451.X при регистрации в СПП нового ИМП.

Параметры

Параметр «moduleId» — заданный идентификатор для модуля связи ИИЭР 1451.X.

Параметр «[out] «timIds» возвращается приложению и содержит все известные ИМП для данного модуля ИИЭР 1451.X.

Возвращаемый результат: код ошибки.

10.1.3 Метод «IEEE1451Dot0::TransducerServices::TIMDiscovery::reportChannels»

IDL:
```
Args::UInt16 reportChannels(
in Args::UInt16       timId,
out Args::UInt16Array channelIds,
out Args::StringArray names).
```

Данный метод возвращает перечень и имена каналов преобразователя для данного ИМП. Информация возвращается от кэшированной ЭТДП (ЭТДП, записанной в кэш-память).

Параметры

Парметр «timId» — заданный ИМП.

Параметр «[out] «channelIds» возвращается приложению и содержит все известные каналы преобразователя для данного ИМП.

Параметр «[out] «names» возвращается приложению и содержит имена каналов преобразователя.

Возвращаемый результат: код ошибки.

10.2 Интерфейс доступа к преобразователю

«IEEE1451Dot0::TransducerServices::TransducerAccess»

IDL:
```
interface TransducerAccess { }.
```

Интерфейс «TransducerAccess» («Доступ к преобразователю») обеспечивается уровнем ИИЭР 1451.0 и вызывается приложением для предоставления доступа к каналам преобразователя. Большинство приложений будет преимущественно взаимодействовать через данный интерфейс для выполнения операций считывания и записи ИМП. Для сохранения небольшого размера данного интерфейса в интерфейсе «TransducerManager» («Управляющий интерфейс преобразователя») содержится еще несколько прогрессивных методов. Все данные методы приведены в таблице 85.

Таблица 85 — Методы интерфейса «TransducerAccess» («Доступ к преобразователю»)

IEEE1451dot0::TransducerServices::TransducerAccess
Args::UInt16 open(in Args::UInt16 timId, in Args::UInt16 channelId, out Args::UInt16 transCommId);
Args::UInt16 openQoS(in Args::UInt16 timId, in Args::UInt16 channelId, inout Args::QoSParams qosParams, out Args::UInt16 transCommId);
Args::UInt16 openGroup(in Args::UInt16Array timIds, in Args::UInt16Array channelIds, out Args::UInt16 transCommId);
Args::UInt16 openGroupQoS(in Args::UInt16Array timIds, in Args::UInt16Array channelIds, inout Args::QoSParams qosParams, out Args::UInt16 transCommId);
Args::UInt16 close(in Args::UInt16 transCommId);
Args::UInt16 readData (in Args::UInt16 transCommId, in Args::TimeDuration time-out, in Args::UInt8 SamplingMode, out Args::ArgumentArray result);
Args::UInt16 writeData (in Args::UInt16 transCommId, in Args::TimeDuration time-out, in Args::UInt8 SamplingMode, in Args::ArgumentArray value);
Args::UInt16 startReadData(in Args::UInt16 transCommId, in Args::TimeInstance triggerTime, in Args::TimeDuration time-out, in Args::UInt8 SamplingMode, in AppCallback callback, out Args::UInt16 operationId);

Окончание таблицы 85

IEEE1451dot0::TransducerServices::TransducerAccess
Args::UInt16 startWriteData(in Args::UInt16 transCommId, in Args::TimeInstance triggerTime, in Args::TimeDuration time-out, in Args::UInt8 SamplingMode, in Args::ArgumentArray value, in AppCallback callback, out Args::UInt16 operationId);
Args::UInt16 startStream(in Args::UInt16 transCommId, in AppCallback callback, out Args::UInt16 operationId);
Args::UInt16 cancel(in Args::UInt16 operationId);

10.2.1 Метод «IEEE1451Dot0::TransducerServices::TransducerAccess::open»

IDL:
```
Args::UInt16 open(
in Args::UInt16  timId,
in Args::UInt16  channelId,
out Args::UInt16 transCommId).
```

Данный метод открывает канал связи требуемого ИМП или канала преобразователя и возвращает параметр «transCommId», который будет использован для последующих запросов. Для «Quality of Service» («Качество сервиса») используется значение по умолчанию.

Параметры

Параметр «timId» назначает требуемый ИМП.

Параметр «channelId» назначает требуемый канал преобразователя. Данное поле позволяет задать адресацию к одному каналу преобразователя, прокси-каналу преобразователя, группе каналов преобразователя или всем каналам преобразователя, подсоединенным к СПП. Для адресации к ИМП используется нулевое значение данного параметра (нулевое значение канала преобразователя). Более подробная информация представлена в 5.3.

Параметр «[out] «transCommId» возвращается приложению уровнем ИИЭР 1451.0. Данный идентификатор будет использован для последующих запросов.

Возвращаемый результат: код ошибки.

10.2.2 Метод «IEEE1451Dot0::TransducerServices::TransducerAccess::openQoS»

IDL:
```
Args::UInt16 openQoS(
in Args::UInt16          timId,
in Args::UInt16          channelId,
inout Args::QoSParams    qosParams,
out Args::UInt16          transCommId).
```

Данный метод открывает канал связи с заданным ИМП/каналом преобразователя и возвращает параметр «transCommId», который будет использован для последующих запросов. Применяется особое значение для «Quality of Service (QoS)» («Качество сервиса») связи. В случае неудачного запроса параметр «qosParams» будет изменен и возвращен приложению с целью установки параметра «hint» («Подсказка»), при котором может быть обеспечено приемлемое качество сервиса связи «QoS».

Параметры

Параметр «timId» назначает требуемый ИМП.

Параметр «channelId» назначает требуемый канал преобразователя. Данное поле позволяет задать адресацию к одному каналу преобразователя, прокси-каналу преобразователя, группе каналов преобразователя или всем каналам преобразователя, подсоединенным к СПП. Для адресации к ИМП используется нулевое значение данного параметра (нулевое значение канала преобразователя). Более подробная информация представлена в 5.3.

Входной/выходной параметр «qosParams» задает требуемое качество сервисных параметров связи. Более подробная информация представлена в 9.3.1.3.

Выходной параметр «[out] «transCommId» возвращается приложению уровнем ИИЭР 1451.0. Данный идентификатор будет использован для последующих запросов.

Возвращаемый результат: код ошибки.

10.2.3 Метод «IEEE1451Dot0::TransducerServices::TransducerAccess::openGroup»

Целью данного метода является присвоение некоторому числу ИМП групповых адресов, которые в дальнейшем будут использованы, чтобы приписать каналы преобразователя к адресным группам. Описание адресных групп представлено в 5.3.2.

IDL:
```
Args::UInt16 openGroup(
in Args::UInt16Array timIds,
in Args::UInt16Array channelIds,
out Args::UInt16     transCommId).
```

Данный метод открывает групповой канал связи к заданным ИМП/каналам преобразователя и возвращает параметр «transCommId», который будет использован для последующих запросов. Применяется значение «Quality of Service (QoS)» («Качество сервиса») связи по умолчанию.

Существует однозначное соответствие между положением массивов «timId» и «channelId». Каналы преобразователя могут находиться в одном или разных ИМП. Все ИМП должны быть прикреплены к одному и тому же модулю связи. Если заданному «timId» соответствует несколько «channelId», то параметр «timId» должен быть продублирован для каждого «channelId» внутри такого ИМП, чтобы оба списка имели одинаковую длину.

Параметры

Параметр «timIds» назначает требуемые ИМП.

Параметр «channelIds» назначает требуемые каналы преобразователя. Данное поле позволяет задать адресацию к одному каналу преобразователя, прокси-каналу преобразователя, группе каналов преобразователя или всем каналам преобразователя, подсоединенным к СПП. Более подробная информация представлена в 5.3.

Выходной параметр «[out] «transCommId» возвращается приложению уровнем ИИЭР 1451.0. Данный идентификатор будет использован для последующих запросов.

Возвращаемый результат: код ошибки.

10.2.4 Метод «IEEE1451Dot0::TransducerServices::TransducerAccess::openGroupQoS»

Целью данного метода является присвоение некоторому числу ИМП групповых адресов, которые в дальнейшем будут использованы, чтобы приписать каналы преобразователя к адресным группам. Описание адресных групп представлено в 5.3.2.

IDL:
```
Args::UInt16 openGroupQoS(
in Args::UInt16Array      timIds,
in Args::UInt16Array      channelIds,
inout Args::QoSParams     qosParams,
out Args::UInt16          transCommId).
```

Данный метод открывает групповой канал связи к заданным ИМП/каналам преобразователя и возвращает параметр «transCommId», который будет использован для последующих запросов. Применяется особое значение «Quality of Service (QoS)» («Качество сервиса») связи. В случае неудачного запроса параметр «qosParams» будет изменен и возвращен приложению с целью установки параметра «hint» («Подсказка»), при котором может быть обеспечено приемлемое качество сервиса связи «QoS».

Существует однозначное соответствие между положением массивов «timId» и «channelId». Каналы преобразователя могут находиться в одном или разных ИМП. Все ИМП должны быть прикреплены к одному и тому же модулю связи.

Параметры

Параметр «timIds» назначает требуемые ИМП.

Параметр «channelIds» назначает требуемые каналы преобразователя. Данное поле позволяет задать адресацию к одному каналу преобразователя, прокси-каналу преобразователя, группе каналов преобразователя или всем каналам преобразователя, подсоединенным к СПП. Более подробная информация представлена в 5.3.

Входной/выходной параметр «qosParams» задает требуемое качество сервисных параметров связи. Более подробная информация представлена в 9.3.1.3.

Выходной параметр «[out] «transCommId» возвращается приложению уровнем ИИЭР 1451.0. Данный идентификатор будет использован для последующих запросов.

Возвращаемый результат: код ошибки.

10.2.5 Метод «IEEE1451Dot0::TransducerServices::TransducerAccess::close»

IDL:
```
Args::UInt16 close( in Args::UInt16 transCommId).
```

Данный метод закрывает сессию связи преобразователя. Приложение должно считать параметр «transCommId» недействительным. Следует отметить, что последующий запрос «open» («открыть») может вернуть предыдущее значение.

См. также метод «TransducerManager::unlock( )» об информации вызова метода «close( )» для заблокированных (закрытых для изменений) параметров «transCommId».

Параметры

Параметр «transCommId» определяет сессию связи, которая должна быть закрыта.

Возвращаемый результат: код ошибки.

10.2.6 Метод «IEEE1451Dot0::TransducerServices::TransducerAccess::readData»

```
IDL: Args::UInt16 readData(
in Args::UInt16           transCommId,
in Args::TimeDuration     timeout,
in Args::UInt8            SamplingMode,
out Args::ArgumentArray   result).
```

Данный метод осуществляет блокирующее считывание назначенного канала (каналов) преобразователя. Массив аргументов «ArgumentArray» может содержать множество атрибутов, как показано в разделах 7—8, причем каждый атрибут представлен отдельным «аргументом» в массиве аргументов «ArgumentArray». Приложение может контролировать, какие атрибуты возвращаются при использовании запроса «TransducerManager::configureAttributes( )».

В случае считывания одного канала преобразователя всегда будет «результирующий» аргумент, который содержит считываемый канал преобразователя. Тип такого аргумента определяется моделью данных канала преобразователя и при условии, что канал преобразователя имеет ЭТДП калибровки. Например, считывание простого канала преобразователя, не имеющего ЭТДП калибровки, всегда будет происходить в собственном формате канала преобразователя (например, UInt8 или Float32Array). В случае если канал преобразователя все-таки определяет коррекции со стороны СПП при помощи ЭТДП калибровки, типом данных всегда будет являться Float32 или Float32Array.

В случаях считывания группы каналов преобразователя всегда будет вложенный «результирующий» массив аргументов «ArgumentArray», который содержит аргумент для каждого канала преобразователя в группе. Такие аргументы будут иметь порядковые номера в массиве, начиная с «0», что соответствует порядку пар ИМП/канал преобразователя при запросах «openGroup( )» или «openGroupQoS( )». Тип данных каждого возвращаемого аргумента будет таким же, как и при считывании одного канала преобразователя, описанном в предыдущем абзаце.

Параметры

Параметр «transCommId» обозначает используемую сессию связи преобразователя.

Параметр «timeout» назначает время ожидания для выполнения считывания без генерации ошибки временной задержки (тайм-аута). Следует отметить, что временная задержка может возникнуть в результате ошибок связи или ошибок триггерных сигналов.

Параметр «SamplingMode» определяет триггерный механизм. Подробнее см. 5.11 и 7.1.2.4.

Выходной «[out] «result» ArgumentArray — массив аргументов с возвращаемыми значениями.

Возвращаемый результат: код ошибки.

10.2.7 Метод «IEEE1451Dot0::TransducerServices::TransducerAccess::writeData»

```
IDL: Args::UInt16 writeData(
in Args::UInt16           transCommId,
in Args::TimeDuration     timeout,
in Args::UInt8            SamplingMode,
in Args::ArgumentArray    value).
```

Данный метод осуществляет блокирующую запись определенных каналов преобразователя. Массив аргументов ArgumentArray имеет множество атрибутов, как показано в разделе 7; каждый атрибут представлен отдельным «аргументом» в массиве аргументов.

В случае записи одного канала преобразователя отправитель запроса должен предоставить значение «value» аргумента, которое содержит значение канала преобразователя. Отправитель запроса должен предоставить результат с использованием совместимого типа данных, который требуется для канала преобразователя и оговорен в ЭТДП канала преобразователя. Простые преобразования всех типов числовых данных совершаются уровнем ИИЭР 1451.0 в СПП. Следует отметить, что при этом может возникнуть потеря точности, если тип результирующих данных имеет меньший размер. Например, если для исполнительного устройства требуется тип данных UInt8, то предоставленный тип Float32 будет преобразован до UInt8 с соответствующей потерей точности еще до передачи каналу преобразователя. В случаях, когда коррекции осуществляются СПП (как указано в ЭТДП калибровки для данного канала преобразователя), тип данных должен быть числовым. Он будет преобразован в тип данных Float32 или Float32Array еще до прохождения процедур коррекции. Данные на выходе процедуры коррекции будут преобразованы в форму, требуемую моделью данных исполнительного устройства, определенную в ЭТДП канала преобразователя.

В случаях записи для группы каналов преобразователя всегда будет вложенное значение «value» массива аргументов, которое содержит аргумент для каждого канала преобразователя из группы. Доступ к каналам преобразователя осуществляется согласно их порядковым номерам в массиве, начиная с «0», что соответствует порядку пар ИМП/канал преобразователя для запросов openGroup( ) или openGroupQoS( ). Тип данных каждого аргумента должен подчиняться правилам записи для одного канала преобразователя, рассмотренным в предыдущем абзаце.

Параметры

Параметр «transCommId» определяет, какую сессию связи преобразователя необходимо использовать.

Параметр «timeout» назначает время ожидания для выполнения считывания без генерации ошибки временной задержки (тайм-аута). Следует отметить, что временная задержка может возникнуть в результате ошибок связи или ошибок триггерных сигналов.

Параметр «SamplingMode» определяет триггерный механизм. Подробнее см. 5.11 и 7.1.2.4.

Значения «value» массива аргументов «ArgumentArray» представляют собой значения входных сигналов исполнительного устройства.

Возвращаемый результат: код ошибки.

10.2.8 Метод «IEEE1451Dot0::TransducerServices::TransducerAccess::startReadData»

IDL:

```
Args::UInt16 startReadData(
in Args::UInt16          transCommId,
in Args::TimeInstance    triggerTime,
in Args::TimeDuration    timeout,
in Args::UInt8           SamplingMode,
in AppCallback           callback,
out Args::UInt16         operationId).
```

Данный метод начинает неблокирующее считывание определенных каналов преобразователя. После завершения считывания для обратного вызова объекта запускается запрос «AppCallback::measurementUpdate( )».

Параметры

Параметр «transCommId» определяет, какую сессию связи преобразователя необходимо использовать.

Параметр «triggerTime» назначает время начала операции считывания. Если назначенное значение «triggerTime» соответствует времени в прошлом, то это приводит к немедленному сбою временной задержки (тайм-аута). Если считывание должно начинаться немедленно, то требуется установить значения «secs==0», «nsecs==0».

Параметр «timeout» назначает время ожидания после запуска процедуры считывания без генерации ошибки временной задержки (тайм-аута). Следует отметить, что временная задержка может возникнуть в результате ошибок связи или ошибок триггерных сигналов.

Параметр «SamplingMode» определяет триггерный механизм. Подробнее см. 5.11 и 7.1.2.4.

Параметр «callback» связан с интерфейсом, который должен быть запущен после завершения считывания. Интерфейс также должен быть запущен в случае сбоев.

Выходной параметр «[out] «operationId» представляет собой идентификатор, который может быть использован для отмены запроса на считывание.

Возвращаемый результат: код ошибки.

10.2.9 Метод «IEEE1451Dot0::TransducerServices::TransducerAccess::startWriteData»

IDL:

```
Args::UInt16 startWriteData(
in Args::UInt16          transCommId,
in Args::TimeInstance    triggerTime,
in Args::TimeDuration    timeout,
in Args::UInt8           SamplingMode,
in Args::ArgumentArray   value,
in AppCallback           callback,
out Args::UInt16         operationId).
```

Данный метод начинает неблокирующую операцию записи в определенные каналы преобразователя. После завершения записи для обратного вызова объекта используется запрос «AppCallback::measurementUpdate( )».

Параметры

Параметр «transCommId» определяет, какую сессию связи преобразователя необходимо использовать.

Параметр «triggerTime» назначает время начала операции записи. Если назначенное значение «triggerTime» соответствует времени в прошлом, то это приводит к немедленному сбою временной задержки (тайм-аута). Если запись должна начинаться немедленно, то требуется установить значения «secs==0», «nsecs==0».

Параметр «timeout» назначает время ожидания после запуска процедуры записи без генерации ошибки временной задержки (тайм-аута). Следует отметить, что временная задержка может возникнуть в результате ошибок связи или ошибок триггерных сигналов.

Параметр «SamplingMode» определяет триггерный механизм. Подробнее см. 5.11 и 7.1.2.4.

Значения «value» массива аргументов «ArgumentArray» представляют собой заданные значения входных сигналов исполнительного устройства. Более подробная информация представлена в 10.2.7.

Параметр «callback» связан с интерфейсом, который должен быть запущен после завершения записи. Интерфейс также должен быть запущен в случае сбоев.

Выходной параметр «[out] «operationId» представляет собой идентификатор, который может быть использован для отмены запроса на запись.

Возвращаемый результат: код ошибки.

10.2.10 Метод «IEEE1451Dot0::TransducerServices::TransducerAccess::startStream»

IDL:

```
Args::UInt16 startStream(
in Args::UInt16     transCommId,
in AppCallback      callback,
out Args::UInt16    operationId).
```

Данный метод начинает работу потока измерений. Параметр «transCommId» должен быть создан с запросами «openQoS( )» или «openGroupQoS( )». В случае последнего все каналы преобразователя должны относиться к одному и тому же ИМП. Каждый раз в случае доступности новых данных измерений для обратного вызова объекта используется запрос «AppCallback::measurementUpdate( )».

Параметры

Параметр «transCommId» определяет, какую сессию связи преобразователя необходимо использовать.

Параметр «callback» связан с интерфейсом, запускаемым после завершения записи набора данных в исполнительное устройство или получения набора данных от датчика. Интерфейс также запускается в случае сбоев.

Выходной параметр «[out] «operationId» представляет собой идентификатор, который может быть использован для отмены потока измерений.

Возвращаемый результат: код ошибки.

10.2.11 Метод «IEEE1451Dot0::TransducerServices::TransducerAccess::cancel»

IDL: `Args::UInt16 cancel( in Args::UInt16 operationId).`

Данный метод отменяет блокирующее считывание, блокирующую запись или поток измерений. Обратный вызов будет сделан с отметкой «CANCEL» в коде ошибки.

Параметры

Параметр «operationId» назначает операцию для отмены.

Возвращаемый результат: код ошибки.

10.3 Интерфейс управления преобразователем «IEEE1451Dot0::TransducerServices::TransducerManager»

IDL: `Interface TransducerManager { }.`

Интерфейс «TransducerManager» («Управление преобразователем») (см. таблицу 86) обеспечивается данной системой и вызывается приложением для предоставления доступа к дополнительным функциям. Большинство приложений не будут взаимодействовать с данным интерфейсом, тем не менее преимущественно взаимодействуя с интерфейсом доступа к преобразователю «TransducerAccess» для выполнения операций считывания и записи канала преобразователя. Интерфейс управления преобразователем «TransducerManager» создан для размещения дополнительных методов, что позволяет сохранять небольшой размер класса «TransducerAccess».

Таблица 86 — Методы интерфейса «TransducerManager» («Управление преобразователем»)

IEEE1451dot0::TransducerServices::TransducerManager
Args::UInt16 lock(in Args::UInt16 transCommId, in Args::TimeDuration time-out);
Args::UInt16 unlock(in Args::UInt16 transCommId);
Args::UInt16 reportLocks(out Args::UInt16Array transCommIds);
Args::UInt16 breakLock(in Args::UInt16 transCommId);
Args::UInt16 sendCommand(in Args::UInt16 transCommId, in Args::TimeDuration time-out, in Args::UInt8 cmdClassId, in Args::UInt8 cmdFunctionId, in Args::ArgumentArray inArgs, out Args::ArgumentArray outArgs);
Args::UInt16 startCommand(in Args::UInt16 transCommId, in Args::TimeInstance triggerTime, in Args::TimeDuration time-out, in Args::UInt8 cmdClassId, in Args::UInt8 cmdFunctionId, in Args::ArgumentArray inArgs, in AppCallback callback, out Args::UInt16 operationId);
Args::UInt16 configureAttributes(in Args::UInt16 transCommId, in Args::StringArray attributeNames);
Args::UInt16 trigger(in Args::UInt16 transCommId, in Args::TimeInstance triggerTime, in Args::TimeDuration time-out, in Args::UInt16 SamplingMode);
Args::UInt16 startTrigger(in Args::UInt16 transCommId, in Args::TimeInstance triggerTime, in Args::TimeDuration time-out, in Args::UInt16 SamplingMode, in AppCallback callback, out Args::UInt16 operationId);
Args::UInt16 clear(in Args::UInt16 transCommId, in Args::TimeDuration time-out, in Args::UInt8 clearMode);
Args::UInt16 registerStatusChange(in Args::UInt16 transCommId, in Args::TimeDuration time-out, in AppCallback callback, out Args::UInt16 operationId);
Args::UInt16 unregisterStatusChange(in Args::UInt16 transCommId);

10.3.1 Метод «IEEE1451Dot0::TransducerServices::TransducerManager::lock»

IDL:
```
Args::UInt16 lock(
in Args::UInt16       transCommId,
in Args::TimeDuration timeout).
```

Данный метод запрещает доступ (запирает) к ИМП/каналам преобразователя, заданным параметром «transCommId», что предотвращает доступ прочих приложений к таким ресурсам. Чтобы не допустить взаимного закрытия доступа в многопоточных средах, приложение должно запирать ресурсы согласованно.

Применение данного метода должно допускать множественные запреты доступа к ресурсам по одному и тому же запросу без блокировки.

В случаях, когда параметр «transCommId» задает группу, все ИМП/каналы преобразователя такой группы будут закрыты последовательно в порядке, который определен в запросах «openGroup( )» или «openGroupQoS( )».

Параметры

Параметр «transCommId» задает требуемую сессию связи преобразователя.

Параметр «timeout» назначает время ожидания после срабатывания запрета доступа. Величины «secs == 0», «nsecs == 0» означают отсутствие ожидания и могут быть использованы для тестирования существующего запрета. Величины «secs == 0», «nsecs == –1» означают непрерывное (бесконечное) ожидание. Использование значения бесконечного ожидания является крайне опасным, так как может вызвать взаимные запреты доступа и «зависания» системы.

Возвращаемый результат: код ошибки.

10.3.2 Метод «IEEE1451Dot0::TransducerServices::TransducerManager::unlock»

IDL:
```
Args::UInt16 unlock( in Args::UInt16 transCommId).
```

Данный метод открывает доступ к ИМП/каналам преобразователя, заданным параметром «transCommId», что приводит к возобновлению доступа к таким ресурсам для прочих приложений.

В случаях, когда приложение запустило запрос «lock( )» на запрет доступа для одного получателя несколько раз, приложение должно убедиться в том, что запрос «unlock( )» о снятии запрета к доступу вызван такое же число раз. Ресурсы становятся доступными после вызова последнего запроса «unlock( )» о снятии запрета доступа.

Применение данного метода должно допускать альтернативные источники запросов «unlock( )» о снятии запрета доступа. Это является действительным только в случае, если запрос «lock( )» о запрете доступа был сделан единственный раз. Примером может служить неблокирующая операция. Запускающая цепочка вызывает запросы «open( )» («открыть»), «lock( )» («закрыть доступ») и «non-blocking read( )» («неблокирующее чтение»). После завершения считывания запускается обратный вызов «AppCallback::measurementUpdate( )». После этого данная цепочка может осуществить запрос «unlock( )» («снять запрет доступа»).

Запрос «close( )» («закрыть») приведет к вызову запроса «unlock( )» («снять запрет доступа») необходимое число раз. При этом будет возвращен предупреждающий код ошибки, сообщающий о попытке осуществления закрытия ресурса, для которого действует запрет доступа.

Параметры

Параметр «transCommId» обозначает требуемую сессию связи преобразователя.

Возвращаемый результат: код ошибки.

10.3.3 Метод «IEEE1451Dot0::TransducerServices::TransducerManager::reportLocks»

IDL:

```
Args::UInt16 unlock(out Args::UInt16Array transCommIds).
```

Данный метод выдает отчет о всех значениях «transCommIds» для устройств, доступ к которым запрещен.

Параметры

Выходной параметр («[out] «transCommId») возвращает массив идентификаторов (ID) устройств, доступ к которым запрещен.

Возвращаемый результат: код ошибки.

10.3.4 Метод «IEEE1451Dot0::TransducerServices::TransducerManager::breakLock»

IDL:

```
Args::UInt16 breakLock( in Args::UInt16 transCommId).
```

Данный метод снимает запрет доступа. В случае если идет процесс неблокирующего считывания или записи или поток измерений обратный вызов будет содержать соответствующий код ошибки. Список кодов ошибки см. в таблице 78.

Параметры

Параметр «transCommId» задает сессию связи преобразователя для снятия запрета доступа.

Возвращаемый результат: код ошибки.

10.3.5 Метод «IEEE1451Dot0::TransducerServices::TransducerManager::sendCommand»

IDL:

```
Args::UInt16 sendCommand(
in Args::UInt16          transCommId,
in Args::TimeDuration    timeout,
in Args::UInt8           cmdClassId,
in Args::UInt8           cmdFunctionId,
in Args::ArgumentArray   inArgs,
out Args::ArgumentArray  outArgs).
```

Данный метод осуществляет операцию блокировки. Формат входящих и исходящих аргументов зависит от вида команды. Отправитель запроса должен убедиться в том, что для каждого входящего аргумента используются корректные типы данных.

Поскольку данная команда является специальной, приложение должно использовать командную ЭТДП, и данный массив аргументов должен содержать байтовый массив, содержащий команду.

Параметры

Параметр «transCommId» задает сессию связи преобразователя.

Параметр «timeout» представляет собой максимальное время ожидания до формирования ошибки времени ожидания (тайм-аута). Значения «secs == 0», «nsecs == –1» задают «непрерывное (бесконечное) ожидание».

Параметр «cmdClassId» задает требуемый код класса команды. Подробная информация представлена в таблице 15.

Параметр «cmdFunctionId» задает требуемый функциональный код команды. Подробная информация представлена в разделе 7.

Параметр «inArgs» представляет собой входные аргументы в форме массива аргументов.

Параметр [out] «outArgs» представляет собой возвращаемые исходящие аргументы.

Возвращаемый результат: код ошибки.

10.3.6 Метод «IEEE1451Dot0::TransducerServices::TransducerManager::sendCommandRaw»

IDL:

```
Args::UInt16 sendCommandRaw(
in Args::UInt16          transCommId,
```

```
in Args::TimeDuration    timeout,
in Args::UInt8           cmdClassId,
in Args::UInt8           cmdFunctionId,
in Args::OctetArray      inArgs,
out Args::OctetArray     outArgs).
```

Данный метод осуществляет операцию блокировки. Формат входящих и исходящих аргументов зависит от вида команды. Отправитель запроса должен убедиться в том, что для каждого входящего аргумента используются корректные типы данных.

Поскольку данная команда является специальной, приложение должно использовать командную ЭТДП, и данный массив аргументов должен содержать байтовый массив, содержащий команду.

Параметры

Параметр «transCommId» задает сессию связи преобразователя.

Параметр «timeout» представляет собой максимальное время ожидания до формирования ошибки времени ожидания (тайм-аута). Значения «secs == 0», «nsecs == –1» задают «непрерывное (бесконечное) ожидание».

Параметр «cmdClassId» задает требуемый код класса команды. Подробная информация представлена в таблице 15.

Параметр «cmdFunctionId» задает требуемый функциональный код команды. Подробная информация представлена в разделе 7.

Параметр «inArgs» представляет собой входные аргументы в форме массива аргументов.

Параметр [out] «outArgs» представляет собой возвращаемые исходящие аргументы.

Возвращаемый результат: код ошибки.

10.3.7 Метод «IEEE1451Dot0::TransducerServices::TransducerManager::startCommand»

IDL:

```
Args::UInt16 startCommand(
in Args::UInt16          transCommId,
in Args::TimeInstance    triggerTime,
in Args::TimeDuration    timeout,
in Args::UInt8           cmdClassId,
in Args::UInt8           cmdFunctionId,
in Args::ArgumentArray   inArgs,
in AppCallback           callback,
out Args::UInt16         operationId).
```

Данный метод начинает неблокирующую операцию. Формат входящих аргументов зависит от вида команды. Отправитель запроса должен убедиться в том, что для каждого входящего аргумента используются корректные типы данных.

Параметры

Параметр «transCommId» задает сессию связи преобразователя.

Параметр «triggerTime» задает время начала операции. Если назначенное значение «triggerTime» соответствует времени в прошлом, то это приводит к немедленному сбою временной задержки (тайм-аута). В специальном случае, если действие должно быть осуществлено немедленно, требуется установить значения «secs==0», «nsecs==0».

Параметр «timeout» представляет собой максимальное время ожидания до формирования ошибки времени ожидания (тайм-аута). Значения «secs == 0», «nsecs == –1» задают «непрерывное (бесконечное) ожидание».

Параметр «cmdClassId» задает требуемый код класса команды. Подробная информация представлена в таблице 15.

Параметр «cmdFunctionId» задает требуемый функциональный код команды. Подробная информация представлена в разделе 7.

Параметр «inArgs» представляет собой входные аргументы в форме массива аргументов. Данные параметры зависят от вида команды.

Параметр «callback» задает интерфейс обратного вызова. Для этого используется метод «AppCallback::commandComplete( )».

Параметр [out] «operationId» представляет собой возвращаемый идентификатор (ID) операции.

Возвращаемый результат: код ошибки.

10.3.8 Метод «IEEE1451Dot0::TransducerServices::TransducerManager::configureAttributes»

IDL:
```
Args::UInt16 configureAttributes(
in Args::UInt16         transCommId,
in Args::StringArray    attributeNames).
```

Данный метод конфигурирует параметр «transCommId» для операций считывания или потока измерений. Он задает необходимые атрибуты, которые должны быть включены в возвращаемый массив аргументов. Более подробно соответствующие наименования приведены в разделах 7—8.

Параметры

Параметр «transCommId» задает сессию связи преобразователя.

Параметр «attributeNames» задает наименования требуемых атрибутов.

Возвращаемый результат: код ошибки.

10.3.9 Метод «IEEE1451Dot0::TransducerServices::TransducerManager::trigger»

IDL:
```
Args::UInt16 trigger(
in Args::UInt16          transCommId,
in Args::TimeInstance    triggerTime,
in Args::TimeDuration    timeout,
in Args::UInt16          SamplingMode).
```

Данный метод осуществляет запуск блокирующего триггера для требуемого «transCommId».

Параметры

Параметр «transCommId» задает сессию связи преобразователя.

Параметр «triggerTime» задает время начала операции. Если назначенное значение «triggerTime» соответствует времени в прошлом, то это приводит к немедленному сбою временной задержки (тайм-аута). В специальном случае, если действие должно быть осуществлено немедленно, требуется установить значения «secs==0», «nsecs==0».

Параметр «timeout» представляет собой максимальное время ожидания до формирования ошибки времени ожидания (тайм-аута). Значения «secs == 0», «nsecs == –1» задают «непрерывное (бесконечное) ожидание».

Параметр «SamplingMode» определяет триггерный режим. Подробнее см. 5.11 и 7.1.2.4.

Возвращаемый результат: код ошибки.

10.3.10 Метод «IEEE1451Dot0::TransducerServices::TransducerManager::startTrigger»

IDL:
```
Args::UInt16 startTrigger(
in Args::UInt16          transCommId,
in Args::TimeInstance    triggerTime,
in Args::TimeDuration    timeout,
in Args::UInt16          SamplingMode,
in AppCallback           callback,
out Args::UInt16         operationId).
```

Данный метод осуществляет запуск неблокирующего триггера для требуемого «transCommId».

Параметры

Параметр «transCommId» задает сессию связи преобразователя.

Параметр «triggerTime» задает время начала операции. Если назначенное значение «triggerTime» соответствует времени в прошлом, то это приводит к немедленному сбою временной задержки (тайм-аута). В специальном случае, если действие должно быть осуществлено немедленно, требуется установить значения «secs==0», «nsecs==0».

Параметр «timeout» представляет собой максимальное время ожидания до формирования ошибки времени ожидания (тайм-аута). Значения «secs == 0», «nsecs == –1» задают «непрерывное (бесконечное) ожидание».

Параметр «SamplingMode» определяет триггерный режим. Подробнее см. 5.11 и 7.1.2.4.

Параметр «callback» задает интерфейс обратного вызова. Для этого используется метод «AppCallback::triggerComplete( )».

Параметр [out] «operationId» представляет собой возвращаемый идентификатор (ID) операции.

Возвращаемый результат: код ошибки.

10.3.11 Метод «IEEE1451Dot0::TransducerServices::TransducerManager::clear»

IDL:
```
Args::UInt16 clear(
in Args::UInt16         transCommId,
```

```
in Args::TimeDuration  timeout,
in Args::UInt8         clearMode).
```

Данный метод осуществляет очистку требуемого «transCommId».

Параметры

Параметр «transCommId» задает сессию связи преобразователя.

Параметр «timeout» представляет собой максимальное время ожидания до формирования ошибки времени ожидания (тайм-аута). Значения «secs == 0», «nsecs == –1» задают «непрерывное (бесконечное) ожидание».

Параметр «clearMode» задает режим очистки, как показано в таблице 87.

Таблица 87 — Опции режима очистки

Нумерация	Описание
0	Зарезервировано
1	Очистить все
2	Очистить канал связи
3	Очистить буферы
4	Перезагрузить состояние ИМП
5	Очистить кэш ЭТДП
6—127	Зарезервировано
128—255	Открыто для изготовителей

Возвращаемый результат: код ошибки.

10.3.12 Метод «IEEE1451Dot0::TransducerServices::TransducerManager::registerStatusChange»

IDL:
```
Args::UInt16 registerStatusChange(
in Args::UInt16       transCommId,
in Args::TimeDuration timeout,
in AppCallback        callback,
out Args::UInt16      operationId).
```

Данный метод регистрирует ответный вызов приложения для событий изменения состояния ИМП требуемого «transCommId».

Параметры

Параметр «transCommId» задает сессию связи преобразователя.

Параметр «timeout» представляет собой максимальное время ожидания до формирования ошибки времени ожидания (тайм-аута). Значения «secs == 0», «nsecs == –1» задают «непрерывное (бесконечное) ожидание».

Параметр «callback» задает интерфейс обратного вызова. Для этого используется метод «AppCallback::statusChange( )».

Параметр [out] «operationId» представляет собой возвращаемый идентификатор (ID) операции.

Возвращаемый результат: код ошибки.

10.3.13 Метод «IEEE1451Dot0::TransducerServices::TransducerManager::unregisterStatusChange»

IDL:
```
Args::UInt16 unregisterStatusChange( in Args::UInt16 transCommId).
```

Данный метод отменяет регистрацию ответного вызова приложения для событий изменения состояния ИМП требуемого «transCommId».

Параметры

Параметр «transCommId» задает сессию связи преобразователя.

Возвращаемый результат: код ошибки.

10.4 Интерфейс управления ЭТДП «IEEE1451Dot0::TransducerServices::TedsManager»

IDL:
```
interface TedsManager { }.
```

Интерфейс управления ЭТДП «TedsManager» обеспечивается уровнем ИИЭР 1451.0 и вызывается приложением для обеспечения доступа к ЭТДП. Методы данного интерфейса перечислены в таблице 88.

Таблица 88 — Методы доступа к ЭТДП

IEEE1451Dot0::TransducerServices::TedsManager
Args::UInt16 readTeds(in Args::UInt16 transCommId, in Args::TimeDuration time-out, in Args::UInt8 tedsType, out Args::ArgumentArray teds);
Args::UInt16 writeTeds(in Args::UInt16 transCommId, in Args::TimeDuration time-out, in Args::UInt8 tedsType, in Args::ArgumentArray teds);
Args::UInt16 readRawTeds(in Args::UInt16 transCommId, in Args::TimeDuration time-out, in Args::UInt8 tedsType, out Args::OctetArray rawTeds);
Args::UInt16 writeRawTeds(in Args::UInt16 transCommId, in Args::TimeDuration time-out, in Args::UInt8 tedsType, in Args::OctetArray rawTeds);
Args::UInt16 updateTedsCache(in Args::UInt16 transCommId, in Args::TimeDuration time-out, in Args::UInt8 tedsType);

10.4.1 Метод «IEEE1451Dot0::TransducerServices::TedsManager::readTeds»

IDL:
```
Args::UInt16 readTeds(
in Args::UInt16          transCommId,
in Args::TimeDuration    timeout,
in Args::UInt8           tedsType,
out Args::ArgumentArray  teds).
```

Данный метод считывает требуемый блок ЭТДП из кэша ЭТДП. Если считывание ЭТДП из кэша недоступно, то ЭТДП будет считываться из ИМП. Информация об ЭТДП возвращается в виде массива аргументов.

Параметры

Параметр «transCommId» задает сессию связи преобразователя.

Параметр «timeout» задает продолжительность ожидания до возвращения ошибки времени ожидания (тайм-аута), если отклик не был получен. Значения «secs == 0», «nsecs == 0» задают отсутствие ожидания и могут использоваться для назначения считывания только из кэша. Значения «secs == 0», «nsecs == —1» задают «непрерывное (бесконечное) ожидание».

Параметр «tedsType» определяет ЭТДП, которую необходимо вернуть. Коды доступа к ЭТДП представлены в таблице 17.

Массив аргументов «[out] «teds» содержит информацию об ЭТДП. Значения могут быть получены в соответствии с именем атрибута. Наименования полей ЭТДП представлены в разделе 8.

Возвращаемый результат: код ошибки.

10.4.2 Метод «IEEE1451Dot0::TransducerServices::TedsManager::writeTeds»

IDL:
```
Args::UInt16 writeTeds(
in Args::UInt16          transCommId,
in Args::TimeDuration    timeout,
in Args::UInt8           tedsType,
in Args::ArgumentArray   teds).
```

Данный метод записывает требуемый блок ЭТДП в ИМП. По окончании записи кэш ЭТДП также обновляется. Предоставляемая информация ЭТДП кодируется в массив аргументов. Производится внутреннее преобразование в корректную форму «кортежа» и передача данных ИМП в виде байтового массива.

Массив аргументов должен включать все обязательные поля ЭТДП, требуемые для типа ЭТДП, запись которой осуществляется. В случае если какое-либо обязательное поле ЭТДП пропущено, возвращается ошибка.

Параметры

Параметр «transCommId» задает сессию связи преобразователя.

Параметр «timeout» задает продолжительность ожидания до возвращения ошибки времени ожидания (тайм-аута), если отклик не был получен. Значения «secs == 0», «nsecs == 0» задают отсутствие ожидания и могут использоваться для назначения считывания только из кэша. Значения «secs == 0», «nsecs == –1» задают «непрерывное (бесконечное) ожидание».

Параметр «tedsType» определяет ЭТДП, которую необходимо вернуть. Коды доступа к ЭТДП представлены в таблице 17.

Массив аргументов «[out] «teds» содержит информацию об ЭТДП. Значения могут быть извлечены в соответствии с именем атрибута. Наименования полей ЭТДП представлены в разделе 8.

Возвращаемый результат: код ошибки.

10.4.3 Метод «IEEE1451Dot0::TransducerServices::TedsManager::readRawTeds»

IDL:
```
Args::UInt16 readRawTeds(
in Args::UInt16          transCommId,
in Args::TimeDuration    timeout,
in Args::UInt8            tedsType,
out Args::OctetArray     rawTeds).
```

Данный метод считывает требуемый блок ЭТДП непосредственно из ЭТДП в обход кэша ЭТДП. Информация ЭТДП возвращается в изначальной форме байтового массива. Кэш ЭТДП не обновляется.

Параметры

Параметр «transCommId» задает сессию связи преобразователя.

Параметр «timeout» задает продолжительность ожидания до возвращения ошибки времени ожидания (тайм-аута), если отклик не был получен. Значения «secs == 0», «nsecs == 0» задают отсутствие ожидания и могут использоваться для назначения считывания только из кэша. Значения «secs == 0», «nsecs == –1» задают «непрерывное (бесконечное) ожидание».

Параметр «tedsType» определяет ЭТДП, которую необходимо вернуть. Коды доступа к ЭТДП представлены в таблице 17.

Параметр «[out] «rawTeds» представляет собой байтовый массив, содержащий информацию ЭТДП в изначальной форме в виде «кортежа».

Возвращаемый результат: код ошибки.

10.4.4 Метод «IEEE1451Dot0::TransducerServices::TedsManager::writeRawTeds»

IDL:
```
Args::UInt16 writeRawTeds(
in Args::UInt16          transCommId,
in Args::TimeDuration    timeout,
in Args::UInt8           tedsType,
in Args::OctetArray      rawTeds).
```

Данный метод записывает требуемый блок ЭТДП в ИМП в обход кэша ЭТДП. Предоставляемая информация ЭТДП кодируется в форме «кортежа» в байтовом массиве. Проверка байтового массива не проводится.

ПРЕДУПРЕЖДЕНИЕ

Убедитесь, что все обязательные поля ЭТДП включены в байтовый массив, так как запись, сделанная этим методом, полностью заменяет всю ЭТДП.

Параметры

Параметр «transCommId» задает сессию связи преобразователя.

Параметр «timeout» задает продолжительность ожидания до возвращения ошибки времени ожидания (тайм-аута), если отклик не был получен. Значения «secs == 0», «nsecs == –1» задают «непрерывное (бесконечное) ожидание».

Параметр «tedsType» определяет ЭТДП, которую необходимо записать. Коды доступа к ЭТДП представлены в таблице 17.

Параметр «rawTeds» представляет собой байтовый массив, содержащий информацию ЭТДП в изначальной форме в виде «кортежа».

Возвращаемый результат: код ошибки.

10.4.5 Метод «IEEE1451Dot0::TransducerServices::TedsManager::updateTedsCache»

IDL:
```
Args::UInt16 updateTedsCache(
in Args::UInt16          transCommId,
in Args::TimeDuration    timeout,
in Args::UInt8           tedsType).
```

Данный метод осуществляет обновление кэша ЭТДП. Контрольная сумма ЭТДП будет считана из ИМП и сопоставлена с контрольной суммой из кэша ЭТДП. Если контрольные суммы отличаются, то ЭТДП будет считана из ИМП и сохранена в кэше.

Параметры

Параметр «transCommId» задает сессию связи преобразователя.

Параметр «timeout» задает продолжительность ожидания до возвращения ошибки времени ожидания (тайм-аута), если отклик не был получен. Значения «secs == 0», «nsecs == –1» задают «непрерывное (бесконечное) ожидание».

Параметр «tedsType» определяет ЭТДП, которую необходимо считать. Коды доступа к ЭТДП представлены в таблице 17.

Возвращаемый результат: код ошибки.

10.5 Интерфейс управления связями «IEEE1451Dot0::TransducerServices::CommManager»

IDL: `interface CommManager { }.`

Интерфейс управления коммуникациями (связями) «CommManager» предоставляется уровнем ИИЭР 1451.0 и вызывается приложением для предоставления общего механизма управления доступными коммуникациями СПП. Методы данного интерфейса перечислены в таблице 89 и рассмотрены в 10.5.1.

Таблица 89 — Методы интерфейса «CommManager»

IEEE1451dot0::TransducerServices::CommManager
Args::UInt16 getCommModule(in Args::UInt8 moduleId, out ModuleCommunication::Comm commObject, out Args::UInt8 type, out Args::UInt8 technologyId);

10.5.1 Метод «IEEE1451Dot0::TransducerServices::CommManager::getCommModule»

IDL:
```
Args::UInt16 getCommModule(
in Args::UInt8                  moduleId,
out ModuleCommunication::Comm   commObject,
out Args::UInt8                 type,
out Args::UInt8                 technologyId).
```

Данный метод возвращает абстрактный «Comm» объект приложениям, которым необходимо обойти обработку данных на уровне ИИЭР 1451.0 и взаимодействовать напрямую с объектом связей более низкого уровня. Ввод параметра «type» позволяет приложению безопасно перейти на более низкий уровень объекта «P2PComm» или «NetComm».

Приложения должны использовать крайние меры предосторожности при доступе к нижележащим объектам «Comm», так как их некорректное использование может негативно отразиться на самом уровне ИИЭР 1451.0. Данный метод предоставляется для обеспечения возможности выхода за пределы архитектуры уровня ИИЭР 1451.0.

Параметры

Параметр «moduleId» — требуемый идентификатор (ID) модуля связи.

Параметр «[out] «commObject» возвращается приложению и представляет собой ссылку на нижележащий объект.

Параметр «[out] «type» возвращается приложению для того, чтобы позволить безопасно перейти на нижний уровень. Действительные значения представлены параметром «[out] «technologyId», который определяет технологию взаимодействия с нижележащим уровнем ИИЭР 1451.X. См. таблицу 90.

Параметр «[out] «technologyId» определяет технологию взаимодействия с нижележащим уровнем ИИЭР 1451.X. См. таблицу 99.

Таблица 90 — Нумерация типа «Comm»

Нумерация	Наименование типа кода	Описание
0	P2P_TYPE	Определяет объект P2PComm
1	NET_COMM_TYPE	Определяет объект NetComm
2—255	Зарезервировано	

Возвращаемый результат: код ошибки.

10.6 Интерфейс ответа приложениям «IEEE1451Dot0::TransducerServices::AppCallback»

IDL: `interface AppCallback { }.`

Интерфейс ответа приложениям «AppCallback» обеспечивается приложениями и вызывается уровнем ИИЭР 1451.0 для предоставления доступа к неблокирующим входам/выходам (I/O) и потокам измерения. Методы интерфейса перечислены в таблице 91.

Таблица 91 — Методы интерфейса ответа приложениям «AppCallback»

IEEE1451Dot0::TransducerServices::AppCallback
Args::UInt16 measurementUpdate(in Args::UInt16 operationId, in Args::ArgumentArray measValues, in Args::UInt16 status);
Args::UInt16 actuationComplete (in Args::UInt16 operationId, in Args::UInt16 status);
Args::UInt16 statusChange(in Args::UInt16 operationId, in Args::UInt16 status);
Args::UInt16 commandComplete(in Args::UInt16 operationId, in Args::ArgumentArray outArgs, in Args::UInt16 status);
Args::UInt16 triggerComplete(in Args::UInt16 operationId, in Args::UInt16 status);

10.6.1 Метод «IEEE1451Dot0::TransducerServices::AppCallback::measurementUpdate»

IDL:
```
Args::UInt16 measurementUpdate(
in Args::UInt16          operationId,
in Args::ArgumentArray   measValues,
in Args::UInt16          status).
```

Данный метод запускается после запросов «startRead( )» («Начать чтение») или «startStream( )» («Начать поток измерений»). В случае неблокирующих операций измерения возвращаются приложению. В случае потока измерений обратный вызов запускается каждый раз при наличии новых доступных данных измерения.

Параметры

Параметр «operationId» задает требуемый идентификатор (ID), который был возвращен по запросам «startRead( )» («Начать чтение») или «startStream( )» («Начать поток измерений»).

Параметр «measValues» содержит информацию об измерениях. Величины могут быть возвращены в имени атрибута. Наименования атрибутов см. в разделе 7. Более подробное описание «read( )» («Считать») приведено в 10.2.6.

Параметр «status» задает код ошибки неблокирующей операции считывания или операции с потоком.

Возвращаемый результат: приложение должно вернуть код состояния уровню ИИЭР 1451.0. Коды ошибок представлены в 9.3.1.2.

10.6.2 Метод «IEEE1451Dot0::TransducerServices::AppCallback::actuationComplete»

IDL:
```
Args::UInt16 actuationComplete (
in Args::UInt16     operationId,
in Args::UInt16     status).
```

Данный метод запускается после запроса «startWrite( )» («Начать запись»). В случае неблокирующих операций информация о состоянии возвращается приложению.

Параметры

Параметр «operationId» задает требуемый идентификатор (ID), который был возвращен по запросу «startWrite( )» («Начать запись»).

Параметр «status» задает код ошибки неблокирующей операции записи.

Возвращаемый результат: приложение должно вернуть код состояния уровню ИИЭР 1451.0. Коды ошибок представлены в 9.3.1.2.

10.6.3 Метод «IEEE1451Dot0::TransducerServices::AppCallback::statusChange»

IDL:
```
Args::UInt16 statusChange(
in Args::UInt16      operationId,
in Args::UInt16      status).
```

Данный метод запускается после запроса «registerStatusChange( )» («Сменить состояние регистра»).

Параметры

Параметр «operationId» задает требуемый идентификатор (ID), который был возвращен по запросу «registerStatusChange( )» («Сменить состояние регистра»).

Параметр «status» определяет информацию о состоянии ИМП или канала преобразователя.

Возвращаемый результат: приложение должно вернуть код состояния уровню ИИЭР 1451.0. Коды ошибок представлены в 9.3.1.2.

10.6.4 Метод «IEEE1451Dot0::TransducerServices::AppCallback::commandComplete»

IDL:
```
Args::UInt16 commandComplete(
in Args::UInt16          operationId,
in Args::ArgumentArray   outArgs,
in Args::UInt16          status).
```

Данный метод запускается после запроса «startCommand( )» («Начать команду»). Он возвращает приложению исходящий массив аргументов.

Параметры

Параметр «operationId» задает требуемый идентификатор (ID), который был возвращен по запросам «startRead( )» («Начать чтение») или «startStream( )» («Начать поток измерений»).

Параметр «outArgs» содержит возвращаемый массив аргументов. Конкретная информация отличается для каждой команды.

Параметр «status» определяет код ошибки в результате неблокирующей операции отправки команды.

Возвращаемый результат: приложение должно вернуть код состояния уровню ИИЭР 1451.0. Коды ошибок представлены в 9.3.1.2.

10.6.5 Метод «IEEE1451Dot0::TransducerServices::AppCallback::triggerComplete»

IDL:
```
Args::UInt16 triggerComplete(
in Args::UInt16          operationId,
in Args::UInt16          status).
```

Данный метод запускается после запроса «startTrigger( )» («Начать триггер»). Метод предоставляет приложению информацию о состоянии завершения работы триггера.

Параметры

Параметр «operationId» задает требуемый идентификатор (ID), который был возвращен по запросу «startTrigger( )» («Начать триггер»).

Параметр «status» определяет код ошибки в результате неблокирующей операции триггерной команды.

Возвращаемый результат: приложение должно вернуть код состояния уровню ИИЭР 1451.0. Коды ошибок представлены в 9.3.1.2.

11 API модульных связей

IDL:
```
module ModuleCommunication { }.
```

API модульных связей обеспечивает интерфейс между функциями, определенными уровнем ИИЭР 1451.0, и функциями связи, определенными другими стандартами комплекса ИИЭР 1451.

Данные интерфейсы ИИЭР 1451.0 находятся внутри IDL модуля «IEEE1451Dot0».

Примечание — В данном разделе обозначения «ИИЭР 1451.0» или «1451.0» относятся к устройству или части устройства, соответствующего настоящему стандарту. Термины «ИИЭР 1451.X» или «1451.X» относятся к устройству или части устройства, соответствующего стандартам ИИЭР 1451.2—1997, ИИЭР 1451.3—2003, ИИЭР 1451.5—2007 [B4], ИИЭР P1451.6 [B3] или иному аналогичному стандарту. Стандарты ИИЭР 1451.1—1999 и ИИЭР 1451.4—2004 исключены из данного списка.

Таблица 92 — Классы и интерфейсы API сервисов преобразователя

Интерфейс	Описание
Comm	Абстрактный интерфейс «Comm» обеспечивает механизмы контроля жизненного цикла объекта ИИЭР 1451.X
P2PComm	Интерфейс «P2PComm» обеспечивает выполнение операций «point-to-point» («линейных») связей

Окончание таблицы 92

Интерфейс	Описание
NetComm	Интерфейс «NetComm» обеспечивает выполнение операций сетевых связей
Registration	Интерфейс «Registration» обеспечивает методы регистрации модуля ИИЭР 1451.X на уровне ИИЭР 1451.0
P2PRegistration	Интерфейс «P2PRegistration» обеспечивает методы регистрации особых ИМП на уровне ИИЭР 1451.0
NetRegistration	Интерфейс «NetRegistration» обеспечивает методы регистрации особых ИМП и групп ИМП на уровне ИИЭР 1451.0
Recieve	Абстрактный интерфейс «Recieve» не определяет какие-либо обобщенные методы. Он предоставлен для возможности будущего расширения
P2PRecieve	Интерфейс «P2PRecieve» обеспечивает методы для модуля линейных связей ИИЭР 1451.X для оповещения уровня ИИЭР 1451.0 о полученных сообщениях. Также он обеспечивает метод прерывания операции
NetRecieve	Интерфейс «NetRecieve» обеспечивает методы для модуля сетевых связей ИИЭР 1451.X для оповещения уровня ИИЭР 1451.0 о полученных сообщениях. Также он обеспечивает метод прерывания операции

11.1 Абстрактный интерфейс «IEEE1451Dot0::ModuleCommunication::Comm»

IDL: `interface Comm { }.`

Абстрактный интерфейс «Comm» обеспечивается уровнем ИИЭР 1451.X и вызывается уровнем ИИЭР 1451.0, чтобы осуществить общий механизм контроля жизненного цикла объекта ИИЭР 1451.X. Методы абстрактного интерфейса «Comm» перечислены в таблице 93.

Таблица 93 — Методы абстрактного интерфейса «Comm»

IEEE1451Dot0::ModuleCommunication::Comm
Args::UInt16 init( );
Args::UInt16 shutdown( );
Args::UInt16 sleep(in Args::TimeDuration duration);
Args::UInt16 wakeup();
Args::UInt16 setLocalConfiguration(in Args::ArgumentArray params);
Args::UInt16 getLocalConfiguration(out Args::ArgumentArray params);
Args::UInt16 sendLocalCommand(in Args::UInt8 cmdClassId, in Args::UInt8 cmdFunctionId, in Args::ArgumentArray inArgs, out Args::ArgumentArray outArgs);
Args::UInt16 describe(out Args::UInt8 logicalType, out Args::UInt8 physicalType, out Args::_String name);

11.1.1 Метод «IEEE1451Dot0::ModuleCommunication::Comm::init»

IDL: `Args::UInt16 init( ).`

Данный метод вызывается при запуске после подачи питания и когда уровню ИИЭР 1451.0 требуется перевод объекта ИИЭР 1451.X в режим включения питания. В случае динамического адреса каждый объект ИИЭР 1451.X должен вызвать соответствующий метод «Registration::Register1451.X( )» для регистрации данного объекта внутри уровня ИИЭР 1451.0.

Параметры: отсутствуют.

Возвращаемый результат: код ошибки.

11.1.2 Метод EEE1451Dot0::ModuleCommunication::Comm::shutdown»

IDL: `Args::UInt16 shutdown( ).`

Данный метод будет вызываться для координации упорядоченного отключения объекта ИИЭР 1451.X. В случае динамического адреса каждый объект ИИЭР 1451.X должен вызвать метод «Registrat ion::unRegister1451.X( )» для отмены регистрации в системе ИИЭР 1451.0.

Параметры: отсутствуют.

Возвращаемый результат: код ошибки.

11.1.3 Метод «IEEE1451Dot0::ModuleCommunication::Comm::sleep»

IDL: `Args::UInt16 sleep(in Args::TimeDuration duration).`

Данный метод вызывается уровнем ИИЭР 1451.0 для перевода объекта ИИЭР 1451.X в спящий режим (то есть режим низкого энергопотребления). Данная операция происходит, если уровень ИИЭР 1451.0 не собирается использовать связь через интерфейс ИИЭР 1451.X в течение значительного периода времени.

Параметры

Параметр «duration» — продолжительность времени «сна». Отрицательное значение времени означает, что устройство ИИЭР 1451.X должно находиться в спящем режиме до вызова метода выхода из спящего режима.

Возвращаемый результат: код ошибки.

11.1.4 Метод «IEEE1451Dot0::ModuleCommunication::Comm::wakeup»

IDL: `Args::UInt16 wakeup().`

Данный метод вызывается уровнем ИИЭР 1451.0 для вывода объекта ИИЭР 1451.X из спящего режима (то есть режима низкого энергопотребления). Данная операция происходит, если уровень ИИЭР 1451.0 предварительно перевел интерфейс ИИЭР 1451.X в спящий режим и в настоящий момент требует возобновления связи.

Параметры: отсутствуют.

Возвращаемый результат: код ошибки.

11.1.5 Метод «IEEE1451Dot0::ModuleCommunication::Comm::performOperation»

IDL:
```
Args::UInt16 performOperation(
in Args::UInt16          operationId,
in Args::TimeDuration    timeout,
in Args::ArgumentArray   inArgs,
out Args::ArgumentArray  outArgs).
```

Данный низкоуровневый механизм отправляет произвольную команду локальному уровню ИИЭР 1451.X. Данный метод осуществляет блокирующую операцию. Формат входящих и исходящих аргументов зависит от вида команды. Отправитель запроса должен убедиться в том, что для каждого входящего аргумента используются верные типы данных.

Параметры

Параметр «operationId» задает требуемый код класса команды. Дозволенные операции определяет уровень ИИЭР 1451.X.

Параметр «timeout» задает максимальное время, на которое отправитель команды осуществляет блокировку перед возвращением ошибки времени ожидания (тайм-аута). Значения «secs == 0», «nsecs == –1» назначают «непрерывное (бесконечное) ожидание».

Параметр «inArgs» содержит входящие аргументы в форме массива аргументов.

Параметр «[out] «outArgs» содержит возвращаемые исходящие аргументы.

Возвращаемый результат: код ошибки.

11.1.6 Метод «IEEE1451Dot0::ModuleCommunication::Comm::setLocalConfiguration(delete)»

IDL: `Args::UInt16 setLocalConfiguration (in Args::ArgumentArray params).`

Данный метод вызывается уровнем ИИЭР 1451.0 для конфигурирования интерфейса локального устройства ИИЭР 1451.X. Содержание массива аргументов «params» определяется уровнем ИИЭР 1451.X.

Параметры

Параметр «params» представляет собой массив аргументов из переменных состояния конфигурации. Каждая переменная состояния, необходимая для уровня ИИЭР 1451.X, должна быть возвращена из такого массива для установки объекта ИИЭР 1451.X.

Возвращаемый результат: код ошибки.

11.1.7 Метод «IEEE1451Dot0::ModuleCommunication::Comm::setLocalConfiguration(delete)»

IDL: `Args::UInt16 getLocalConfiguration (outout Args::ArgumentArray params).`

Данный метод вызывается уровнем ИИЭР 1451.0 для возвращения конфигурации интерфейса локального устройства уровня ИИЭР 1451.X. Содержание массива аргументов «params» зависит от ИИЭР 1451.X.

Параметры

Параметр «[out] «params» — массив аргументов из переменных состояния конфигурации. Каждая переменная состояния, необходимая для уровня ИИЭР 1451.X, должна быть возвращена в данном массиве для последующей настройки объекта уровня ИИЭР 1451.X.

Возвращаемый результат: код ошибки.

11.1.8 Метод «IEEE1451Dot0::ModuleCommunication::Comm::sendLocalCommand(Delete)»

IDL:

```
Args::UInt16 sendLocalCommand(
in Args::UInt8          cmdClassId,
in Args::UInt8          cmdFunctionId,
in Args::ArgumentArray  inArgs,
out Args::ArgumentArray outArgs).
```

Данный механизм низкого уровня отправляет произвольную команду локальному уровню ИИЭР 1451.X. Данный метод производит блокирующую операцию. Формат входящих и исходящих аргументов зависит от вида команды. Отправитель запроса должен убедиться в том, что для каждого входящего аргумента используются корректные типы данных.

Параметры

Параметр «cmdClassId» задает требуемый код класса команды. Подробная информация представлена в таблице 15.

Параметр «cmdFunctionId» задает требуемый функциональный код команды. Подробная информация представлена в разделе 7.

Параметр «inArgs» представляет собой входящие аргументы в форме массива аргументов.

Параметр «[out] «outArgs» представляет собой возвращаемые исходящие аргументы.

Возвращаемый результат: код ошибки.

11.1.9 Метод «IEEE1451Dot0::ModuleCommunication::Comm::describe»

IDL:

```
UInt16 describe (
out UInt8   logicalType,
out UInt8   physicalType,
out _String name ).
```

Данный обобщенный механизм описывает тип объекта более низкого уровня ИИЭР 1451.X.

Параметры

Параметр «[out] «logicalType» задает тип интерфейса. Подробная информация представлена в таблице 94.

Параметр «[out] «physicalType» задает тип физического уровня. Подробная информация представлена в таблице 99.

Параметр «[out] «name» представляет собой удобочитаемое для человека имя, присвоенное в процессе регистрации.

Возвращаемый результат: код ошибки.

Таблица 94 — Логические типы интерфейса

Нумерация	Логический тип
0	Зарезервировано
1	Линейный «Point to point (P2P)» («От точки к точке»)
2	Сетевой «Network (Net)»
3—127	Зарезервировано
128—255	Открыто для изготовителей

11.2 Интерфес «IEEE1451Dot0::ModuleCommunication::P2PComm»

IDL:

```
interface P2PComm { }.
```

Интерфейс «P2PComm» полезен в случаях, когда узел только принимает и опционально отвечает на входящие сообщения (то есть данный интерфейс никогда не начинает операции связи). Также интерфейс «P2PComm» может использоваться в случаях, когда узел инициирует операцию связи только для одного адресата.

Интерфейс «P2PComm» обеспечивается уровнем ИИЭР 1451.X и вызывается уровнем ИИЭР 1451.0 для выполнения операций связи. Линейные методы «Comm» («point to point» — «от точки к точке») перечислены в таблице 95.

Таблица 95 — Методы «Comm» («point to point» — «от точки к точке»)

IEEE1451Dot0::ModuleCommunication::P2PComm
Args::UInt16 read(in Args::TimeDuration time-out, inout Args::UInt32 len, out Args::OctetArray payload, out Args::_Boolean last);
Args::UInt16 write(in Args::TimeDuration time-out, in Args::OctetArray payload, in Args::_Boolean last);
Args::UInt16 flush( );
Args::UInt16 readSize(out Args::UInt32 cacheSize);
Args::UInt16 setPayloadSize(in Args::UInt32 size);
Args::UInt16 abort( );
Args::UInt16 commStatus(out Args::UInt16 statusCode);
Args::UInt16 setRemoteConfiguration(in Args::TimeDuration time-out, in Args::ArgumentArray params);
Args::UInt16 getRemoteConfiguration(in Args::TimeDuration time-out, out Args::ArgumentArray params);
Args::UInt16 sendRemoteCommand(in Args::TimeDuration time-out, in Args::UInt8 cmdClassId, in Args::UInt8 cmdFunctionId, in Args::ArgumentArray inArgs, out Args::ArgumentArray outArgs);

11.2.1 Метод «IEEE1451Dot0::ModuleCommunication::P2PComm::read»

IDL:
```
Args::UInt16 read(
in Args::TimeDuration timeout,
in Args::UInt32       maxLen,
out Args::OctetArray  payload,
out Args::_Boolean    last).
```

Данный метод вызывается уровнем ИИЭР 1451.0 для извлечения информации о текущей операции связи. Данный метод всегда вызывается уровнем ИИЭР 1451.0 на принимающем узле как для одностороннего, так и для двустороннего направлений связи. В случае двустороннего направления связи данный метод также вызывается на инициирующем узле для получения ответа.

В случаях, когда уровень ИИЭР 1451.0 осуществляет многочисленные запросы «read( )» («Считать») для больших полезных нагрузок, длина каждой возвращаемой передачи определяется значением «len», а не общей длиной. Для управления кэшированной длиной используется метод «readSize( )» («Считать размер»), описанный в 11.2.4.

Параметры

Параметр «timeout» задает максимальное время, на которое отправитель команды осуществляет блокировку перед возвращением ошибки времени ожидания (тайм-аута). Значения «secs == 0», «nsecs == –1» назначают «постоянное (бесконечное) ожидание».

Параметр «maxLen» показывает максимальное число байтов, которые должны быть переданы через уровень ИИЭР 1451.X.

Параметр «[out] «payload» представляет собой байтовый массив, который обеспечивается уровнем ИИЭР 1451.0 для уровня ИИЭР 1451.X. Уровень ИИЭР 1451.X передает доступные данные в такой массив.

Параметр «[out] «last» показывает, следует ли уровню ИИЭР 1451.0 выполнять дополнительные запросы «read( )» («считать»). Значение «False» («Ложь») означает, что уровень ИИЭР 1451.X содержит еще несколько байтов для связи с уровнем ИИЭР 1451.0 и что уровень ИИЭР 1451.0 должен сделать дополнительные запросы «read( )». Значение «True» («Истина») означает, что вся полезная нагрузка полностью передана от уровня ИИЭР 1451.X уровню ИИЭР 1451.0.

Возвращаемый результат: код ошибки.

11.2.2 Метод «IEEE1451Dot0::ModuleCommunication::P2PComm::write»

IDL:
```
Args::UInt16 write(
in Args::TimeDuration  timeout,
```

```
in Args::OctetArray    payload,
in Args::_Boolean      last).
```

Данный метод вызывается уровнем ИИЭР 1451.0 для начала или продолжения операции связи. Данный метод всегда вызывается на инициирующем узле для начала операции связи. В случае двухстороннего направления связи данный метод также вызывается на принимающем узле для обеспечения ответного сообщения.

Параметры

Параметр «timeout» задает максимальное время, на которое отправитель команды осуществляет блокировку перед возвращением ошибки времени ожидания (тайм-аута). Значения «secs == 0», «nsecs == –1» назначают «непрерывное (бесконечное) ожидание».

Параметр «payload» представляет собой байтовый массив для связи.

Параметр «last» показывает, следует ли уровню ИИЭР 1451.0 выполнять дополнительные запросы «write( )» («записать») для обеспечения дополнительных фрагментов полезной нагрузки. Значение «False» («Ложь») означает, что уровень ИИЭР 1451.0 содержит еще несколько байтов для отправки и еще несколько раз будет выполнять запросы «write( )». Значение «True» («Истина») означает, что вся полезная нагрузка полностью передана от уровня ИИЭР 1451.0 уровню ИИЭР 1451.X.

Возвращаемый результат: код ошибки.

11.2.3 Метод «IEEE1451Dot0::ModuleCommunication::P2PComm::flush»

IDL: `Args::UInt16 flush().`

Данный метод вызывается уровнем ИИЭР 1451.0 для сброса информации из кэш-памяти на удаленную сторону. В большинстве случаев данный запрос не используется, так как уровень ИИЭР 1451.X всегда выполняет передачу данных после запроса «write( )» с параметром «last», имеющим значение «True» («Истина»).

Параметры: отсутствуют.

Возвращаемый результат: код ошибки.

11.2.4 Метод «IEEE1451Dot0::ModuleCommunication::P2PComm::readSize»

IDL: `Args::UInt16 readSize( out Args::UInt32 cacheSize).`

Данный метод вызывается уровнем ИИЭР 1451.0 для получения информации о числе байтов, доступных для немедленного считывания. Данный метод предоставляет данные о размере кэшированных данных. Данная величина может быть меньше полного размера полезной нагрузки, если уровень ИИЭР 1451.X осуществил сегментирование данных.

Параметры

Параметр «[out] «cacheSize» возвращает число байтов, доступных для немедленного считывания.

Возвращаемый результат: код ошибки.

11.2.5 Метод «IEEE1451Dot0::ModuleCommunication::P2PComm::setPayloadSize»

IDL: `Args::UInt16 setPayloadSize( in Args::UInt32 size).`

Данный метод вызывается уровнем ИИЭР 1451.0 для установки полного числа байтов, доступных при полной полезной нагрузке. Данный метод следует использовать в случае, если уровень ИИЭР 1451.0 собирается осуществить многочисленные запросы для вызова метода «write( )» («записать») для передачи полезной нагрузки уровню ИИЭР 1451.X по частям. В случае одиночного запроса «write( )» [то есть когда значение параметра «last» установлено как «True» («Истина») уже для первого вызова] данный запрос не нужен, так как уровень ИИЭР 1451.X должен выполнить внутренний запрос для вызова данного метода и записи размера полезной нагрузки.

Параметры

Параметр «size» задает полный размер полезной нагрузки.

Возвращаемый результат: код ошибки.

11.2.6 Метод «IEEE1451Dot0::ModuleCommunication::P2PComm::abort»

IDL: `Args::UInt16 abort().`

Данный метод вызывается уровнем ИИЭР 1451.0 на инициирующем узле для прерывания или перезагрузки канала связи. Если уровень ИИЭР 1451.X уже инициировал операцию связи с удаленным узлом, то уровень ИИЭР 1451.X должен совершить попытку прерывания удаленной обработки. Для последующих операций связи после данного запроса может быть использован параметр «commId». Данный метод не закрывает канал связи.

Параметры: отсутствуют.

Возвращаемый результат: код ошибки.

11.2.7 Метод «IEEE1451Dot0::ModuleCommunication::P2PComm::commStatus»

IDL: `Args::UInt16 commStatus( out Args::UInt16 statusCode ).`

Данный метод вызывается уровнем ИИЭР 1451.0 на инициирующем узле для получения информации о состоянии локального или удаленного устройства. Список доступных состояний приведен в таблице 96.

Параметры

Параметр «[out] «statusCode» возвращает информацию о состоянии локального (верхний байт) или удаленного (нижний байт) устройств.

Возвращаемый результат: код ошибки.

Таблица 96 — Коды состояния машины

Нумерация	Состояние
1	Idle (режим ожидания)
2	InitiatorWriting (запись инициирующего узла)
3	InitiatorClosing (завершение работы инициирующего узла)
4	InitiatorReceivePending (задержка получения инициирующего узла)
5	InitiatorReading (чтение инициирующего узла)
6	InitiatorAborting (прерывание инициирующего узла)
7	ReceiverIncomingMsg (входящее сообщение принимающего узла)
8	ReceiverReading (чтение принимающего узла)
9	ReceiverWriting (запись принимающего узла)
10	ReceiverOutgoingMsg (исходящее сообщение принимающего узла)
11	ReceiverAborting (прерывание принимающего узла)
12—127	Зарезервировано
128—255	Открыто для изготовителей

11.2.8 Метод «IEEE1451Dot0::ModuleCommunication::P2PComm::setRemoteConfiguration»

IDL:
```
Args::UInt16 setRemoteConfiguration (
in Args::TimeDuration   timeout,
in Args::ArgumentArray  params ).
```

Данный метод вызывается уровнем ИИЭР 1451.0 на инициирующем узле для установки конфигурации на удаленном узле. В случаях когда массив «params» должен быть передан на удаленный узел, рекомендуется использовать стандартный механизм кодер/декодер для преобразования массива аргументов в/из байтовый массив.

Параметры

Параметр «timeout» задает максимальное время ожидания до ошибки времени ожидания (таймаута). Значения «secs == 0», «nsecs == –1» задают «непрерывное (бесконечное) ожидание».

Параметр «params» представляет собой массив аргументов переменных конфигурации состояния. Каждая переменная состояния, необходимая для уровня ИИЭР 1451.X, должна быть получена из данного массива для настройки удаленного объекта уровня ИИЭР 1451.X.

Возвращаемый результат: код ошибки.

11.2.9 Метод «IEEE1451Dot0::ModuleCommunication::P2PComm::getRemoteConfiguration»

IDL:
```
Args::UInt16 getRemoteConfiguration (
in Args::TimeDuration   timeout,
out Args::ArgumentArray params ).
```

Данный метод вызывается уровнем ИИЭР 1451.0 на инициирующем узле для получения конфигурации от удаленного узла. В случаях когда массив «params» должен быть передан от удаленного узла, рекомендуется использовать стандартный механизм кодер/декодер для преобразования массива аргумента в/из байтовый массив.

Параметры

Параметр «timeout» задает максимальное время ожидания до ошибки времени ожидания (таймаута). Значения «secs == 0», «nsecs == –1» задают «непрерывное (бесконечное) ожидание».

Параметр «[out] «params» представляет собой массив аргументов переменных конфигурации состояния. Каждая переменная состояния, необходимая для уровня ИИЭР 1451.X, должна храниться в данном массиве.

Возвращаемый результат: код ошибки.

11.2.10 Метод «IEEE1451Dot0::ModuleCommunication::P2PComm::sendRemoteCommand»

IDL:
```
Args::UInt16 sendRemoteCommand(
in Args::TimeDuration    timeout,
in Args::UInt8           cmdClassId,
in Args::UInt8           cmdFunctionId,
in Args::ArgumentArray   inArgs,
out Args::ArgumentArray  outArgs ).
```

Данный механизм низкого уровня отправляет произвольную команду удаленному уровню ИИЭР 1451.X. Уровню ИИЭР 1451.X рекомендуется использовать стандартный механизм кодер/декодер для преобразования массивов аргументов в/из байтовые массивы. Данный метод осуществляет блокирующую операцию. Формат входящих и исходящих аргументов зависит от вида команды. Отправитель запроса должен убедиться в том, что для каждого входящего аргумента используются верные типы данных.

Параметры

Параметр «timeout» задает максимальное время ожидания до ошибки времени ожидания (таймаута). Значения «secs == 0», «nsecs == –1» задают «непрерывное (бесконечное) ожидание».

Параметр «cmdClassId» задает требуемый код класса команды. Подробная информация представлена в таблице 15.

Параметр «cmdFunctionId» задает требуемый функциональный код команды. Подробная информация представлена в разделе 7.

Параметр «inArgs» представляет собой входящие аргументы в форме массива аргументов.

Параметр «[out] «outArgs» представляет собой возвращаемые исходящие аргументы.

Возвращаемый результат: код ошибки.

11.3 Интерфейс «IEEE1451Dot0::ModuleCommunication::NetComm»

IDL:
```
interface NetComm { }.
```

Интерфейс «NetComm» полезен в случаях, когда узлу требуется инициировать доступ к одному или более адресатам.

Интерфейс «NetComm» обеспечивается уровнем ИИЭР 1451.X и вызывается уровнем ИИЭР 1451.0 для выполнения операций связи. Методы, используемые данным интерфейсом, перечислены в таблице 97.

Таблица 97 — Методы интерфейса «Network Comm»

IEEE1451Dot0::ModuleCommunication::NetComm
Args::UInt16 open(in Args::UInt16 destId, in Args::_Boolean twoWay, out Args::UInt16 maxPayloadLen, out Args::UInt16 commId);
Args::UInt16 openQoS(in Args::UInt16 destId, in Args::_Boolean twoWay, out Args::UInt16 maxPayloadLen, out Args::UInt16 commId, inout Args::QoSParams qosParams);
Args::UInt16 close(in Args::UInt16 commId);
Args::UInt16 readMsg(in Args::UInt16 commId, in Args::TimeDuration time-out, inout Args::UInt32 len, out Args::OctetArray payload, out Args::_Boolean last);
Args::UInt16 readRsp(in Args::UInt16 commId, in Args::TimeDuration time-out, in Args::UInt16 msgId, in Args::UInt32 maxLen, out Args::OctetArray payload, out Args::_Boolean last);
Args::UInt16 writeMsg(in Args::UInt16 commId, in Args::TimeDuration time-out, in Args::OctetArray payload, in Args::_Boolean last, in Args::UInt16 msgId);
Args::UInt16 writeRsp(in Args::UInt16 commId, in Args::TimeDuration time-out, in Args::OctetArray payload, in Args::_Boolean last);

Окончание таблицы 97

IEEE1451Dot0::ModuleCommunication::NetComm
Args::UInt16 flush(in Args::UInt16 commId);
Args::UInt16 readSize(in Args::UInt16 commId, out Args::UInt32 cacheSize);
Args::UInt16 setPayloadSize(in Args::UInt16 commId, in Args::UInt32 size);
Args::UInt16 abort(in Args::UInt16 commId);
Args::UInt16 commStatus(in Args::UInt16 commId, in Args::UInt16 msgId, out Args::UInt16 statusCode);
Args::UInt16 discoverDestinations( );
Args::UInt16 joinGroup(in Args::UInt16 groupId, in Args::UInt16 destId);
Args::UInt16 leaveGroup(in Args::UInt16 groupId, in Args::UInt16 destId);
Args::UInt16 lookupDestId(in Args::UInt16 commId, out Args::UInt16 destId);
Args::UInt16 setRemoteConfiguration(in Args::UInt16 commId, in Args::TimeDuration time-out, in Args::ArgumentArray params);
Args::UInt16 getRemoteConfiguration(in Args::UInt16 commId, in Args::TimeDuration time-out, out Args::ArgumentArray params);
Args::UInt16 sendRemoteCommand(in Args::UInt16 commId, in Args::TimeDuration time-out, in Args::UInt8 cmdClassId, in Args::UInt8 cmdFunctionId, in Args::ArgumentArray inArgs, out Args::ArgumentArray outArgs);

11.3.1 Метод «IEEE1451Dot0::ModuleCommunication::NetComm::open»

IDL:
```
Args::UInt16 open(
in Args::UInt16        destId,
in Args::_Boolean      twoWay,
out Args::UInt16       maxPayloadLen,
out Args::UInt16       commId).
```

Данный метод вызывается уровнем ИИЭР 1451.0 на инициирующем узле для открытия канала связи.

Параметры

Параметр «destId» задает адресата для принимающего СПП или ИМП.

В случае если параметр «twoWay» имеет значение «True» («Истина»), это означает, что инициирующая сторона ожидает ответного результата от операции связи.

Параметр «[out] «maxPayloadLen» отображает максимальный размер полезной нагрузки, которая будет принята при последующих операциях записи или считывания.

Параметр «[out] «commId» возвращается при данном запросе.

Возвращаемый результат: код ошибки.

11.3.2 Метод «IEEE1451Dot0::ModuleCommunication::NetComm::openQoS»

IDL:
```
Args::UInt16 openQoS(
in Args::UInt16          destId,
in Args::_Boolean        twoWay,
out Args::UInt16         maxPayloadLen,
out Args::UInt16         commId,
inout Args::QoSParams    qosParams).
```

Данный метод вызывается уровнем ИИЭР 1451.0 на инициирующем узле для открытия канала связи с параметрами «quality of service» («Качество сервиса») связи. В случае неудачного запроса, завершенного с ошибкой «QOS_FAILURE», параметр «qosParams» будет модифицирован, чтобы показать значения, которые уровень ИИЭР 1451.X может обеспечить.

Параметры

Параметр «destId» задает адресата для принимающего СПП или ИМП.

В случае если параметр «twoWay» имеет значение «True» («Истина»), это означает, что инициирующая сторона ожидает ответного результата от операции связи.

Параметр «[out] «maxPayloadLen» отображает максимальный размер полезной нагрузки, которая будет принята при последующих операциях записи или считывания.

Параметр «[out] «commId» возвращается при данном запросе.

Параметр «[inout] «qosParams» обеспечивает требуемые параметры «quality of service».

Возвращаемый результат: код ошибки.

11.3.3 Метод «IEEE1451Dot0::ModuleCommunication::NetComm::close»

IDL:
```
Args::UInt16 close(Args::UInt16 commId).
```

Данный метод вызывается уровнем ИИЭР 1451.0 для закрытия канала связи. Данный метод вызывается уровнем ИИЭР 1451.0 на инициирующем узле. После закрытия канала никакие операции связи для канала с данным «commId» не допускаются.

Параметры

Параметр «commId» задает канал связи.

Возвращаемый результат: код ошибки.

11.3.4 Метод «IEEE1451Dot0::ModuleCommunication::NetComm::readMsg»

IDL:
```
Args::UInt16 readMsg(
in Args::UInt16          commId,
in Args::TimeDuration    timeout,
inout Args::UInt32       len,
out Args::OctetArray     payload,
out Args::_Boolean       last).
```

Данный метод вызывается уровнем ИИЭР 1451.0 для получения информации о текущей операции связи. Данный метод всегда вызывается уровнем ИИЭР 1451.0 на принимающем узле как для одностороннего, так и для двустороннего направления связи.

В случаях когда уровень ИИЭР 1451.0 осуществляет многочисленные запросы «readMsg( )», предполагающие большие полезные нагрузки, значение параметра «len» представляет собой длину каждой передачи, а не общую длину. Метод управления кэшированной длиной с помощью «readSize( )» представлен в 11.3.9.

Параметры

Параметр «commId» задает канал связи.

Параметр «timeout» задает максимальное время, в течение которого отправитель запроса осуществляет блокировку, до момента, когда должна быть возвращена ошибка времени ожидания (таймаута). Значения «secs == 0», «nsecs == –1» задают «непрерывное (бесконечное) ожидание».

В случае функции «in» параметр «len» показывает максимальное число байтов, которое должно быть передано через уровень ИИЭР 1451.X. В ответном сообщении параметр «len» показывает число переданных байтов.

Параметр «[out] «payload» представляет собой байтовый массив, который предоставляется уровнем ИИЭР 1451.0 для уровня ИИЭР 1451.X. Уровень ИИЭР 1451.X передает доступные данные в такой массив. Следует отметить, что длина возвращенного байтового массива может быть меньше параметра «maxLen».

Параметр «[out] «last» показывает, должны ли быть выполнены дополнительные запросы «readMsg( )» уровнем ИИЭР 1451.0. Значение «False» («Ложь») для данного параметра означает, что уровень ИИЭР 1451.X содержит еще несколько байтов для связи с уровнем ИИЭР 1451.0 и что уровень ИИЭР 1451.0 должен сделать дополнительные запросы «readMsg( )». Значение «True» («Истина») для данного параметра означает, что вся полезная нагрузка была полностью передана от уровня ИИЭР 1451.X к уровню ИИЭР 1451.0.

Возвращаемый результат: код ошибки.

11.3.5 Метод «IEEE1451Dot0::ModuleCommunication::NetComm::readRsp»

IDL:
```
Args::UInt16 readRsp(
in Args::UInt16          commId,
in Args::TimeDuration    timeout,
in Args::UInt16          msgId,
in Args::UInt32          maxLen,
out Args::OctetArray     payload,
out Args::_Boolean       last).
```

Данный метод вызывается уровнем ИИЭР 1451.0 для получения ответной информации о текущей операции связи. Данный метод всегда вызывается уровнем ИИЭР 1451.0 на инициирующем узле как для одностороннего, так и для двустороннего направления связи.

В случаях когда уровень ИИЭР 1451.0 осуществляет многочисленные запросы «readRsp( )», предполагающие большие полезные нагрузки, значение параметра «len» представляет собой длину каждой передачи, а не общую длину. Метод управления кэшированной длиной с помощью «readSize( )» представлен в 11.3.9.

Параметры

Параметр «commId» задает канал связи.

Параметр «timeout» задает максимальное время, в течение которого отправитель запроса осуществляет блокировку, до момента, когда должна быть возвращена ошибка времени ожидания (таймаута). Значения «secs == 0», «nsecs == –1» задают «непрерывное (бесконечное) ожидание».

Параметр «msgId» должен иметь такое же значение, которое было предоставлено в запросе «writeMsg( )», запустившем данную транзакцию. Для информации о способе оповещения инициирующей стороны о готовом ответе см. обратный запрос «notifyRsp( )».

Параметр «maxLen» показывает максимальное число байтов, передаваемых через уровень ИИЭР 1451.X.

Параметр «[out] «payload» представляет собой байтовый массив, который предоставляется уровнем ИИЭР 1451.0 для уровня ИИЭР 1451.X. Уровень ИИЭР 1451.X передает доступные данные в такой массив. Следует отметить, что длина возвращенного байтового массива может быть меньше параметра «maxLen».

Параметр «[out] «last» показывает, должны ли быть выполнены дополнительные запросы «readRsp( )» уровнем ИИЭР 1451.0. Значение «False» («Ложь») для данного параметра означает, что уровень ИИЭР 1451.X содержит еще несколько байтов для связи с уровнем ИИЭР 1451.0 и что уровень ИИЭР 1451.0 должен сделать дополнительные запросы «readRsp( )». Значение «True» («Истина») для данного параметра означает, что вся полезная нагрузка была полностью передана от уровня ИИЭР 1451.X к уровню ИИЭР 1451.0.

Возвращаемый результат: код ошибки.

11.3.6 Метод «IEEE1451Dot0::ModuleCommunication::NetComm::writeMsg»

IDL:

```
Args::UInt16 writeMsg(
in Args::UInt16           commId,
in Args::TimeDuration     timeout,
in Args::OctetArray       payload,
in Args::_Boolean         last,
in Args::UInt16           msgId).
```

Данный метод вызывается уровнем ИИЭР 1451.0 для начала или продолжения операции связи. Данный метод всегда вызывается уровнем ИИЭР 1451.0 на инициирующем узле для начала операции связи. В случае двустороннего направления связи информация о том, как принимающий узел осуществляет ответ, представлена в описании метода «writeRsp( )».

Параметры

Параметр «commId» задает канал связи.

Параметр «timeout» задает максимальное время, в течение которого отправитель запроса осуществляет блокировку, до момента, когда должна быть возвращена ошибка времени ожидания (таймаута). Значения «secs == 0», «nsecs == –1» задают «непрерывное (бесконечное) ожидание».

Параметр «payload» представляет собой байтовый массив для связи.

Параметр «last» показывает, будут ли осуществлены дополнительные запросы для вызова метода «writeMsg( )» уровнем ИИЭР 1451.0 для обеспечения дополнительных фрагментов полезной нагрузки. Значение «False» («Ложь») для данного параметра означает, что уровень ИИЭР 1451.0 содержит еще несколько байтов для отправки и сделает еще несколько дополнительных запросов для вызова метода «writeMsg( )». Значение «True» («Истина») для данного параметра означает, что вся полезная нагрузка была передана от уровня ИИЭР 1451.0 к уровню ИИЭР 1451.X.

Параметр «msgId» задает «message ID» («идентификатор сообщения») для данной транзакции. В случае двусторонних транзакций уровень ИИЭР 1451.X должен передать обратно такое же значение параметра «msgId» инициирующей вызов стороне в запросе «NetReceive::notifyRsp( )». Значение «0» показывает, что запрос является односторонним.

Возвращаемый результат: код ошибки.

11.3.7 Метод «IEEE1451Dot0::ModuleCommunication::NetComm::writeRsp»

IDL:

```
Args::UInt16 writeRsp(
in Args::UInt16           commId,
```

```
in Args::TimeDuration     timeout,
in Args::OctetArray       payload,
in Args::_Boolean         last).
```

Данный метод вызывается уровнем ИИЭР 1451.0, чтобы вернуть ответ для команды.

Параметры

Параметр «commId» задает канал связи.

Параметр «timeout» задает максимальное время, в течение которого отправитель запроса осуществляет блокировку, до момента, когда должна быть возвращена ошибка времени ожидания (таймаута). Значения «secs == 0», «nsecs == –1» задают «непрерывное (бесконечное) ожидание».

Параметр «payload» представляет собой байтовый массив для связи.

Параметр «last» показывает, будут ли осуществлены дополнительные запросы для вызова метода «writeRsp( )» уровнем ИИЭР 1451.0 для обеспечения дополнительных фрагментов полезной нагрузки. Значение «False» («Ложь») для данного параметра означает, что уровень ИИЭР 1451.0 содержит еще несколько байтов для отправки и сделает еще несколько дополнительных запросов для вызова метода «writeRsp( )». Значение «True» («Истина») для данного параметра означает, что вся полезная нагрузка была передана от уровня ИИЭР 1451.0 к уровню ИИЭР 1451.X.

Возвращаемый результат: код ошибки.

11.3.8 Метод «IEEE1451Dot0::ModuleCommunication::NetComm::flush»

IDL: `Args::UInt16 flush(in Args::UInt16 commId).`

Данный метод вызывается уровнем ИИЭР 1451.0 для сброса информации из кэш-памяти на удаленную сторону. В большинстве случаев данный запрос не используется, так как уровень ИИЭР 1451.X всегда выполняет передачу данных после запроса «writeMsg( )» или «writeRsp( )» с параметром «last», имеющим значение «True» («Истина»).

Параметры

Параметр «commId» задает канал связи.

Возвращаемый результат: код ошибки.

11.3.9 Метод «EEE1451Dot0::ModuleCommunication::NetComm::readSize»

IDL:

```
Args::UInt16 readSize(
in Args::UInt16   commId,
out Args::UInt32  cacheSize).
```

Данный метод вызывается уровнем ИИЭР 1451.0 для получения информации о числе байтов, доступных для немедленного считывания. Данный метод предоставляет размер кэшированных данных. Данное значение может быть меньше полного размера полезной нагрузки, если уровень ИИЭР 1451.X произвел сегментирование.

Параметры

Параметр «commId» задает канал связи.

Параметр «[out] «cacheSize» возвращает число байтов, доступных для немедленного считывания.

Возвращаемый результат: код ошибки.

11.3.10 Метод «IEEE1451Dot0::ModuleCommunication::NetComm::setPayloadSize»

IDL:

```
Args::UInt16 setPayloadSize(
in Args::UInt16   commId,
in Args::UInt32   size).
```

Данный метод вызывается уровнем ИИЭР 1451.0 для установки полного числа байтов, доступных для полной полезной нагрузки. Данный метод должен вызываться, когда уровень ИИЭР 1451.0 собирается выполнить множественные запросы для вызова методов «writeMsg( )» или «writeRsp( )» для передачи полезной нагрузки уровню ИИЭР 1451.X по частям.

В случае одиночного запроса «writeMsg( )» или «writeRsp( )» (то есть когда значение параметра «last» установлено как «True» («Истина») уже для первого вызова) данный запрос не нужен, так как уровень ИИЭР 1451.X должен выполнить внутренний запрос для вызова данного метода и записи размера полезной нагрузки.

Параметры

Параметр «commId» задает канал связи.

Параметр «size» задает полный размер полезной нагрузки.

Возвращаемый результат: код ошибки.

11.3.11 Метод «IEEE1451Dot0::ModuleCommunication::NetComm::abort»

IDL: `Args::UInt16 abort(in Args::UInt16 commId).`

Данный метод вызывается уровнем ИИЭР 1451.0 на инициирующем узле для прерывания или перезагрузки канала связи. Если уровень ИИЭР 1451.X уже инициировал операцию связи с удаленным узлом, уровень ИИЭР 1451.X должен совершить попытку прерывания удаленной обработки. Для последующих операций связи после данного запроса может быть использован параметр «commId». Данный метод не закрывает канал связи.

Параметры

Параметр «commId» задает канал связи.

Возвращаемый результат: код ошибки.

11.3.12 Метод «IEEE1451Dot0::ModuleCommunication::NetComm::commStatus»

IDL:
```
Args::UInt16 commStatus(
in Args::UInt16   commId,
in Args::UInt16   msgId,
out Args::UInt16  statusCode).
```

Данный метод вызывается уровнем ИИЭР 1451.0 на инициирующем узле для получения информации о состоянии локального или удаленного устройства. Список доступных состояний приведен в таблице 96.

Параметры

Параметр «commId» задает канал связи.

Параметр «msgId» задает «ID» (идентификатор) сообщения для данной транзакции.

Параметр «[out] «statusCode» возвращает информацию о состоянии локального (верхний байт) или удаленного (нижний байт) устройства.

Возвращаемый результат: код ошибки.

11.3.13 Метод «IEEE1451Dot0::ModuleCommunication::NetComm::discoverDestinations»

IDL:
```
Args::UInt16 discoverDestinations( ).
```

Данный метод вызывается уровнем ИИЭР 1451.0 для запуска процессов обнаружения и регистрации всех ИМП, доступных через данный объект уровня ИИЭР 1451.X. В качестве ответного действия уровень ИИЭР 1451.X вызывает метод «Register::registerDest( )» для каждого адресата.

При нормальном ходе работы уровень ИИЭР 1451.X должен автоматически осуществлять вызов метода «call registerDest( )» во время инициализации и во время события «горячей замены».

Параметры: отсутствуют.

Возвращаемый результат: код ошибки.

11.3.14 Метод «IEEE1451Dot0::ModuleCommunication::NetComm::joinGroup»

IDL:
```
Args::UInt16 joinGroup(
In Args::UInt16   groupId,
In Args::UInt16   destId).
```

Данный метод вызывается уровнем ИИЭР 1451.0 для добавления адресата ИМП в группу для многоадресной передачи. В случае если группы не существует, она будет создана при данном вызове.

Примечание — Данный вызов не приписывает каналы преобразователя к какой-либо адресной группе. Приписку к адресной группе можно осуществить при помощи команды для определения адресной группы, представленной в 7.1.2.3.

Параметры

Параметр «groupId» задает адресную группу для присоединения.

Параметр «destId» задает узел для добавления в адресную группу.

Возвращаемый результат: код ошибки.

11.3.15 Метод «IEEE1451Dot0::ModuleCommunication::NetComm::leaveGroup»

IDL:
```
Args::UInt16 leaveGroup(
in Args::UInt16   groupId,
in Args::UInt16   destId).
```

Данный метод вызывается уровнем ИИЭР 1451.0 для удаления адресата из группы для многоадресной передачи. Запрос для вызова метода «close( )» с параметром «groupId» приведет к удалению из группы всех адресатов.

Параметры

Параметр «groupId» задает адресную группу для удаления.

Параметр «destId» задает узел для удаления из адресной группы.

Возвращаемый результат: код ошибки.

11.3.16 Метод «IEEE1451Dot0::ModuleCommunication::NetComm::lookupDestId»

IDL:
```
Args::UInt16 lookupDestId (
in Args::UInt16   commId,
out Args::UInt16  destId).
```

Данный метод вызывается уровнем ИИЭР 1451.0 для преобразования параметра «commId» назад в параметр «destId». Данный метод может использоваться принимающей стороной для поиска «destination ID» («идентификатора адресата») отправляющей стороны.

Параметры

Параметр «commId» задает канал связи.

Параметр «[out] «destId» задает адресата, связанного с параметром «commId».

Возвращаемый результат: код ошибки.

11.3.17 Метод «IEEE1451Dot0::ModuleCommunication::NetComm::setRemoteConfiguration»

IDL:
```
Args::UInt16 setRemoteConfiguration(
in Args::UInt16        commId,
in Args::TimeDuration  timeout,
in Args::ArgumentArray params).
```

Данный метод вызывается уровнем ИИЭР 1451.0 для установки конфигурации для удаленного адресата. Содержание массива аргументов определяется уровнем ИИЭР 1451.X. В случаях когда массив аргументов должен быть передан на удаленный узел, рекомендуется использовать стандартный механизм кодер/декодер для преобразования массива аргументов в/из байтовый массив.

Параметры

Параметр «commId» задает канал связи.

Параметр «timeout» задает максимальное время, в течение которого отправитель запроса осуществляет блокировку, до момента, когда должна быть возвращена ошибка времени ожидания (таймаута). Значения «secs == 0», «nsecs == –1» задают «непрерывное (бесконечное) ожидание».

Параметр «params» задает параметры конфигурации.

Возвращаемый результат: код ошибки.

11.3.18 Метод «IEEE1451Dot0::ModuleCommunication::NetComm::getRemoteConfiguration»

IDL:
```
Args::UInt16 getRemoteConfiguration(
in Args::UInt16          commId,
in Args::TimeDuration    timeout,
out Args::ArgumentArray  params).
```

Данный метод вызывается уровнем ИИЭР 1451.0 для получения конфигурации от удаленного адресата. Содержание массива аргументов определяется уровнем ИИЭР 1451.X. В случаях когда массив аргументов должен быть передан на удаленный узел, рекомендуется использовать стандартный механизм кодер/декодер для преобразования массива аргументов в/из байтовый массив.

Параметры

Параметр «commId» задает канал связи.

Параметр «timeout» задает максимальное время, в течение которого отправитель запроса осуществляет блокировку, до момента, когда должна быть возвращена ошибка времени ожидания (таймаута). Значения «secs == 0», «nsecs == –1» задают «непрерывное (бесконечное) ожидание».

Параметр «[out] «params» задает параметры конфигурации.

Возвращаемый результат: код ошибки.

11.3.19 Метод «IEEE1451Dot0::ModuleCommunication::NetComm::sendRemoteCommand»

IDL:
```
Args::UInt16 sendRemoteCommand(
in Args::UInt16          commId,
in Args::TimeDuration    timeout,
in Args::UInt8           cmdClassId,
in Args::UInt8           cmdFunctionId,
in Args::ArgumentArray   inArgs,
out Args::ArgumentArray  outArgs).
```

Данный механизм низкого уровня отправляет произвольную команду удаленному уровню ИИЭР 1451.X. Уровню ИИЭР 1451.X рекомендуется использовать стандартный механизм кодер/декодер для преобразования массивов аргументов в/из байтовые массивы. Данный метод осуществляет блокирующую операцию. Формат входящих и исходящих аргументов зависит от вида команды. Отправитель запроса должен убедиться в том, что для каждого входящего аргумента используются верные типы данных.

Параметры

Параметр «commId» задает канал связи.

Параметр «cmdClassId» задает требуемый код класса команды. Подробная информация представлена в таблице 15.

Параметр «cmdFunctionId» задает требуемый функциональный код команды. Подробная информация представлена в разделе 7.

Параметр «timeout» представляет собой максимальное время ожидания до ошибки времени ожидания (тайм-аута). Значения «secs == 0», «nsecs == –1» задают «непрерывное (бесконечное) ожидание».

Параметр «inArgs» представляет собой входящие аргументы в форме массива аргументов.

Параметр «[out] «outArgs» представляет собой возвращаемые исходящие аргументы.

Возвращаемый результат: код ошибки.

11.4 Интерфейс регистрации «IEEE1451Dot0::ModuleCommunication::Registration»

IDL: `interface Registration { }.`

Абстрактный интерфейс регистрации «Registration» является набором методов, которые обеспечиваются уровнем ИИЭР 1451.0 и вызываются уровнем ИИЭР 1451.X. Методы перечислены в таблице 98. Дополнительные методы представлены в описании интерфейсов «P2PRegistration» и «NetRegistration».

Таблица 98 — Методы интерфейса «Registration»

IEEE1451Dot0::ModuleCommunication::Registration
Args::UInt16 unRegisterModule (in Args::UInt8 moduleId);
Args::UInt16 reportModules (in Args::UInt8 maxLen, in Args::UInt8 offset, out Args::UInt8Array moduleIds);
Args::UInt16 getCommModule(in Args::UInt8 moduleId, out Comm commObject, out Args::UInt8 type, out Args:UInt8 technologyId);

11.4.1 Метод «EEE1451Dot0::ModuleCommunication::Registration::registerModule»

Для регистрации интерфейса ИИЭР 1451.X на уровне ИИЭР 1451.0 используются два метода. Первый метод «P2PRegistration» (см. 11.5.1) вызывается устройствами с применением методов «P2PComm». Второй метод «NetRegistration» (см. 11.6.1) применяется с устройствами, использующими методы «NetComm». Нумерация, применяемая для идентификации различных интерфейсов, представлена в таблице 99.

Таблица 99 — Идентификаторы модуля связи

Нумерация	Стандарт
0	Зарезервировано
1	ИИЭР 1451.2—1997 с использованием 10-wire TII
2	ИИЭР 1451.2—1997 с использованием RS-232
3	ИИЭР 1451.3—2003
4	ИИЭР 1451.5—2007 (раздел 6) [B4]
5	ИИЭР 1451.5—2007 (раздел 8) [B4]
6	ИИЭР 1451.5—2007 (раздел 7) [B4]
7	ИИЭР 1451.5—2007 (раздел 9) [B4]
8	ИИЭР P1451.6 [B3]
9—254	Зарезервировано
255	Не соответствует какому-либо изданному или находящемуся в разработке стандарту

11.4.2 Метод «IEEE1451Dot0::ModuleCommunication::Registration::unRegisterModule»

IDL: `Args::UInt16 unRegisterModule (in Args::UInt8 moduleId).`

Данный метод обеспечивается уровнем ИИЭР 1451.0 и вызывается уровнем ИИЭР 1451.X для отмены регистрации интерфейса. Цель данного метода заключается в информировании уровня ИИЭР 1451.0 о том, что объект уровня ИИЭР 1451.X более недоступен.

Параметры

Параметр «moduleId» задает интерфейс, регистрацию которого необходимо отменить. Уровень ИИЭР 1451.0 удалит всю информацию на данном интерфейсе из своего кэша.

Возвращаемый результат: код ошибки.

11.4.3 Метод «IEEE1451Dot0::ModuleCommunication::Registration::reportModules»

IDL:
```
Args::UInt16 reportModules (
in Args::UInt16         maxLen,
in Args::UInt16         offset,
out Args::UInt8Array    moduleIds).
```

Данный метод обеспечивается уровнем ИИЭР 1451.0 и вызывается уровнем ИИЭР 1451.X или другими уровнями, чтобы сообщить об известных интерфейсах.

Параметры

Параметр «maxLen» показывает максимальное число интерфейсов, о которых можно сообщить.

Параметр «offset» обозначает стартовое значение, когда число зарегистрированных модулей больше значения параметра «maxLen». Начальное значение параметра — «ноль».

Параметр «[out] «moduleId» содержит возвращаемые значения, представляющие собой параметры «moduleId» для каждого доступного интерфейса модуля. Длина данного массива может быть меньше значений параметра «maxLen».

Возвращаемый результат: код ошибки.

11.4.4 Метод «IEEE1451Dot0::ModuleCommunication::Registration::getCommModule»

IDL:
```
Args::UInt16 getCommModule(
in Args::UInt8    moduleId,
out Comm          commObject,
out Args::UInt8   type,
out Args::UInt8   technologyId).
```

Данный метод возвращает абстрактный объект «Comm».

Ввод параметра «type» позволяет безопасно перейти на более низкий уровень объекта «P2PComm» или «NetComm».

Следует использовать крайние меры предосторожности при доступе к объектам более низкого уровня «Comm», так как некорректное использование может подвергать опасности уровни ИИЭР 1451.0 и ИИЭР 1451.X. Данный метод предоставляется для выхода за пределы архитектуры уровня ИИЭР 1451.0. Нумерация аргумента «type» приведена в таблице 94.

Параметры

Параметр «moduleId» представляет собой требуемый идентификатор (ID) модуля связи.

Параметр «[out] «commObject» возвращается приложению и является ссылкой на объект более низкого уровня.

Параметр «[out] «type» возвращается приложению для обеспечения безопасного спуска на нижний уровень. Действительные значения данного параметра представлены в таблице 94.

Параметр «[out] «technologyId» задает технологию более низкого уровня ИИЭР 1451.X. Нумерация, используемая для идентификации данных технологий, представлена в таблице 99.

Возвращаемый результат: код ошибки.

11.5 Интерфейс «IEEE1451Dot0::ModuleCommunication::P2PRegistration»

IDL: `interface P2PRegistration { }.`

Интерфейс «P2PRegistration» обеспечивается уровнем ИИЭР 1451.0 и вызывается уровнем ИИЭР 1451.X для случая «P2P» связей. Методы данного интерфейса перечислены в таблице 100.

Таблица 100 — Методы интерфейса «P2PRegistration»

IEEE1451Dot0::ModuleCommunication::P2PRegistration
Args::UInt16 registerModule(in P2PComm commInterface, in Args::UInt8 technologyId, in Args::_String name, out Args::UInt8 moduleId);

Окончание таблицы 100

IEEE1451Dot0::ModuleCommunication::P2PRegistration
Args::UInt16 registerDestination(in Args::UInt8 moduleId, in Args::UInt16 maxPayloadSize);
Args::UInt16 unregisterDestination(in Args::UInt8 moduleId);

11.5.1 Метод «IEEE1451Dot0::ModuleCommunication::P2PRegistration::registerModule»

IDL:
```
Args::UInt16 registerModule(
in P2PComm        commInterface,
in Args::UInt8    technologyId,
in Args::_String  name,
out Args::UInt8   moduleId).
```

Данный метод обеспечивается уровнем ИИЭР 1451.0 и вызывается уровнем ИИЭР 1451.X во время конфигурирования в случае «P2P» связей. Целью данного метода является информирование уровня ИИЭР 1451.0 о доступных уровнях ИИЭР 1451.X.

Параметры

Параметр «commInterface» задает интерфейс «P2PComm». Возможные значения параметра представлены в таблице 94. Уровень ИИЭР 1451.0 записывает информацию в кэш-память и использует ее для вызова соответствующих методов во время выполнения операций связи.

Параметр «technologyId» определяет технологию более низкого уровня ИИЭР 1451.X. Нумерация, используемая для идентификации данных технологий, представлена в таблице 99.

Параметр «name» представляет собой строку символов для задания имени для отображения в удобочитаемом для человека виде.

Выходной параметр «out «moduleId» возвращается в качестве «interface identifier» («идентификатор интерфейса»), который может быть использован в методе «unRegister( )».

Возвращаемый результат: код ошибки.

11.5.2 Метод «IEEE1451Dot0::ModuleCommunication::P2PRegistration::registerDestination»

IDL:
```
Args::UInt16 registerDestination(
in Args::UInt8    moduleId,
in Args::UInt16   maxPayloadSize).
```

Данный метод обеспечивается уровнем ИИЭР 1451.0 и вызывается уровнем ИИЭР 1451.X во время конфигурирования в случае «P2P» связей. Целью данного метода является информирование уровня ИИЭР 1451.0 о подсоединении к интерфейсу и доступности действительных ИМП.

Параметры

Параметр «moduleId» задает «interface identifier» («идентификатор интерфейса») для «P2PComm».

Параметр «maxPayloadLen» показывает максимальный размер полезной нагрузки, которая будет принята при последующих операциях считывания «read( )» или записи «write( )».

Возвращаемый результат: код ошибки.

11.5.3 Метод «IEEE1451Dot0::ModuleCommunication::P2PRegistration::unregisterDestination»

IDL:
```
Args::UInt16 unregisterDestination(in Args::UInt8 moduleId).
```

Данный метод обеспечивается уровнем ИИЭР 1451.0 и вызывается уровнем ИИЭР 1451.X во время конфигурирования в случае «P2P» связей. Целью данного метода является информирование уровня ИИЭР 1451.0 о том, что ИМП отсоединен от интерфейса и более недоступен.

Параметры

Параметр «moduleId» задает «interface identifier» («идентификатор интерфейса») для «P2PComm».

Возвращаемый результат: код ошибки.

11.6 Интерфейс «IEEE1451Dot0::ModuleCommunication::NetRegistration»

IDL:
```
interface NetRegistration { }.
```

Интерфейс «NetRegistration» обеспечивается уровнем ИИЭР 1451.0 и вызывается уровнем ИИЭР 1451.X в случае сетевых связей. Методы данного интерфейса перечислены в таблице 101.

Таблица 101 — Методы интерфейса «NetRegistration»

IEEE1451Dot0::ModuleCommunication::NetRegistration
Args::UInt16 registerModule (in NetComm commInterface, in Args::UInt8 technologyId, in Args::_String name,out Args::UInt8 moduleId);

Окончание таблицы 101

IEEE1451Dot0::ModuleCommunication::NetRegistration
Args::UInt16 registerDestination (in Args::UInt8 moduleId, out Args::UInt16 destId);
Args::UInt16 unRegisterDestination (in Args::UInt8 moduleId, in Args::UInt16 destId);
Args::UInt16 reportDestinations (in Args::UInt8 moduleId, in Args::UInt16 maxLen, in Args::UInt16 offset, out Args::UInt16Array destinations);
Args::UInt16 reportGroups (in Args::UInt8 moduleId, in Args::UInt16 maxLen, in Args::UInt16 offset, out Args::UInt16Array groups);
Args::UInt16 reportGroupMembers (in Args::UInt8 moduleId, in Args::UInt16 groupId, in Args::UInt16 maxLen, in Args::UInt16 offset, out Args::UInt16Array groups);

11.6.1 Метод «IEEE1451Dot0::ModuleCommunication::NetRegistration::registerModule»

IDL:
```
Args::UInt16 registerModule(
in NetComm        commInterface,
in Args::UInt8    technologyId,
in Args::_String  name,
out Args::UInt8   moduleId).
```

Данный метод обеспечивается уровнем ИИЭР 1451.0 и вызывается уровнем ИИЭР 1451.X во время конфигурирования в случае сетевых связей «Net». Целью данного метода является информирование уровня ИИЭР 1451.0 о доступности уровня ИИЭР 1451.X.

Параметры

Параметр «commInterface» задает интерфейс «NetComm». Возможные значения параметра представлены в таблице 94. Уровень ИИЭР 1451.0 записывает информацию в кэш-память и использует ее для вызова соответствующих методов во время выполнения операций связи.

Параметр «technologyId» определяет технологию более низкого уровня ИИЭР 1451.X. Нумерация, используемая для идентификации данных технологий, представлена в таблице 99.

Параметр «name» представляет собой строку символов для задания имени для отображения в удобочитаемом для человека виде.

Параметр «out «moduleId» возвращается в качестве «interface identifier» («идентификатор интерфейса»), который может быть использован в методе «unRegister( )».

Возвращаемый результат: код ошибки.

11.6.2 Метод «IEEE1451Dot0::ModuleCommunication::NetRegistration::registerDestination»

IDL:
```
Args::UInt16 registerDestination (
in Args::UInt8    moduleId,
out Args::UInt16  destId ).
```

Данный метод обеспечивается уровнем ИИЭР 1451.0 и вызывается уровнем ИИЭР 1451.X для регистрации нового адресата. Целью данного метода является информирование уровня ИИЭР 1451.0 о доступности нового адресата. Ответственность за обнаружение новых ИМП, появившихся в системе, и их регистрацию на уровне ИИЭР 1451.0 лежит на уровне ИИЭР 1451.X.

Параметры

Параметр «moduleId» задает идентификатор (ID) объекта для данного интерфейса уровня ИИЭР 1451.X.

Параметр «[out] «destId» представляет собой возвращаемый параметр. Идентификатор адресата «destination identifier» приписан к уровню ИИЭР 1451.0. Уровню ИИЭР 1451.X необходимо кэшировать соответствующую информацию о конечной точке назначения для данного «destId». Параметр «destId» будет передан обратно уровню ИИЭР 1451.X при запросах «open( )» или «openQoS( )».

Возвращаемый результат: код ошибки.

11.6.3 Метод «IEEE1451Dot0::ModuleCommunication::NetRegistration::unRegisterDestination»

IDL:
```
Args::UInt16 unRegisterDestination (
in Args::UInt8    moduleId,
in Args::UInt16   destId).
```

Данный метод обеспечивается уровнем ИИЭР 1451.0 и вызывается уровнем ИИЭР 1451.X для отмены регистрации адресата. Целью данного метода является информирование уровня ИИЭР 1451.0 о том, что адресат более недоступен. Ответственность за обнаружение ИМП, недоступных в системе, и отмену их регистрации на уровне ИИЭР 1451.0 несет уровень ИИЭР 1451.X.

Параметры

Параметр «moduleId» задает идентификатор (ID) объекта для интерфейса ИИЭР 1451.X.

Параметр «destId» представляет собой идентификатор адресата, регистрацию которого необходимо отменить.

Возвращаемый результат: код ошибки.

11.6.4 Метод «IEEE1451Dot0::ModuleCommunication::NetRegistration::reportDestinations»

IDL:

```
Args::UInt16 reportDestinations (
in Args::UInt8          moduleId,
in Args::UInt16         maxLen,
in Args::UInt16         offset,
out Args::UInt16Array   destinations).
```

Данный метод обеспечивается уровнем ИИЭР 1451.0 и вызывается уровнем ИИЭР 1451.X или другими уровнями для сообщения об известных адресатах, связанных с интерфейсом ИИЭР 1451.X. Следует отметить, что при данном запросе не возвращается информация об адресатах группы. В системах с ограниченным объемом памяти могут быть использованы параметры «maxLen» и «offset» для получения только части информации об известных модулях.

Параметры

Параметр «moduleId» задает идентификатор (ID) объекта для данного интерфейса ИИЭР 1451.X.

Параметр «maxLen» показывает максимальное число интерфейсов, о которых можно сообщить.

Параметр «offset» показывает требуемую стартовую позицию с первоначальным значением «ноль».

Параметр «[out] «destinations» обеспечивает возвращаемые значения, представляющие собой значения параметров «destId» для каждого известного адресата.

Возвращаемый результат: код ошибки.

11.6.5 Метод «IEEE1451Dot0::ModuleCommunication::NetRegistration::reportGroups»

IDL:

```
Args::UInt16 reportGroups (
in Args::UInt8          moduleId,
in Args::UInt16         maxLen,
in Args::UInt16         offset,
out Args::UInt16Array   groups).
```

Данный метод обеспечивается уровнем ИИЭР 1451.0 и вызывается уровнем ИИЭР 1451.X или другими уровнями для сообщения об известных группах, связанных с интерфейсом ИИЭР 1451.X.

Параметры

Параметр «moduleId» задает идентификатор (ID) объекта для данного интерфейса ИИЭР 1451.X.

Параметр «maxLen» показывает максимальное число интерфейсов, о которых можно сообщить.

Параметр «offset» показывает требуемую стартовую позицию с первоначальным значением «ноль».

Параметр «[out] «groups» обеспечивает массив для уровня ИИЭР 1451.0, который используется для возврата значений, представляющих собой «destId» для каждой известной группы.

Возвращаемый результат: код ошибки.

11.6.6 Метод «IEEE1451Dot0::ModuleCommunication::NetRegistration::reportGroupMembers»

IDL:

```
Args::UInt16 reportGroupMembers (
in Args::UInt8          moduleId,
in Args::UInt16         groupId,
in Args::UInt16         maxLen,
in Args::UInt16         offset,
out Args::UInt16Array   groups).
```

Данный метод обеспечивается уровнем ИИЭР 1451.0 и вызывается уровнем ИИЭР 1451.X или другими уровнями для сообщения об известных членах конкретной группы «groupId», связанной с интерфейсом ИИЭР 1451.X.

Параметры

Параметр «moduleId» задает идентификатор (ID) объекта для данного интерфейса ИИЭР 1451.X.

Параметр «groupId» показывает требуемый идентификатор (ID) группы.

Параметр «maxLen» показывает максимальное число интерфейсов, о которых можно сообщить.

Параметр «offset» показывает требуемую стартовую позицию с первоначальным значением «ноль».

Параметр «[out] «groups» обеспечивает массив для уровня ИИЭР 1451.0, который используется для возврата значений, представляющих собой «destId» для каждого известного адресата группы.

Возвращаемый результат: код ошибки.

11.7 Интерфейс «IEEE1451Dot0::ModuleCommunication::Receive»

IDL: `interface Receive { }.`

Данный абстрактный интерфейс «Receive» не определяет какие-либо обобщенные методы. Он предоставлен для будущей разработки.

11.8 Интерфейс «IEEE1451Dot0::ModuleCommunication::P2PReceive»

IDL: `interface P2PReceive { }.`

Интерфейс «P2PReceive» является набором методов, которые обеспечиваются уровнем ИИЭР 1451.0 и вызываются уровнем ИИЭР 1451.X. Методы, используемые при применении данного интерфейса, перечислены в таблице 102.

Таблица 102 — Методы интерфейса «P2PReceive»

IEEE1451Dot0::ModuleCommunication::P2PReceive
Args::UInt16 abort(in Args::UInt16 status);
Args::UInt16 notifyMsg(in Args::_Boolean twoWay, in Args::UInt32 payloadLen, in Args::UInt32 cacheLen, in Args::UInt16 status);
Args::UInt16 notifyRsp(in Args::UInt32 payloadLen, in Args::UInt32 cacheLen, in Args::UInt16 status);

11.8.1 Метод «IEEE1451Dot0::ModuleCommunication::P2PReceive::abort»

IDL: `Args::UInt16 abort( in Args::UInt16 status ).`

Данный метод вызывается уровнем ИИЭР 1451.X для прерывания текущей операции. Данный метод вызывается уровнем ИИЭР 1451.X принимающего узла, когда выполнение команды уровня ИИЭР 1451.0 уже было запущено и должно быть прекращено.

Параметры

Параметр «status» предоставляет код ошибки для причины, по которой был сделан запрос «abort( )».

Возвращаемый результат: код ошибки, возвращаемый уровню ИИЭР 1451.X. Как правило, данное возвращаемое значение игнорируется уровнем ИИЭР 1451.X.

11.8.2 Метод «IEEE1451Dot0::ModuleCommunication::P2PReceive::notifyMsg»

IDL:
```
Args::UInt16 notifyMsg(
in Args::_Boolean twoWay,
in Args::UInt32   payloadLen,
in Args::UInt32   cacheLen,
in Args::UInt16   status).
```

Данный метод обеспечивается уровнем ИИЭР 1451.0 и вызывается уровнем ИИЭР 1451.X при доступности входящего сообщения. Данный метод вызывается на принимающем узле.

Параметры

Параметр «twoWay» показывает, что команда не выдала ответное сообщение. Значение «True» («Истина») означает, что ответ ожидается.

Параметр «payloadLen» показывает общий размер полезной нагрузки.

Параметр «cacheLen» показывает число байтов, которое может быть немедленно считано.

Параметр «status» предоставляет код ошибки для причины, по которой был сделан запрос «notifyMsg( )».

Возвращаемый результат: код ошибки, возвращаемый уровню ИИЭР 1451.X. Как правило, данное возвращаемое значение игнорируется уровнем ИИЭР 1451.X.

11.8.3 Метод «IEEE1451Dot0::ModuleCommunication::P2PReceive::notifyRsp»

IDL:
```
Args::UInt16 notifyRsp(
in Args::UInt32   payloadLen,
in Args::UInt32   cacheLen,
in Args::UInt16   status).
```

Данный метод обеспечивается уровнем ИИЭР 1451.0 и вызывается уровнем ИИЭР 1451.X при доступности входящего ответного сообщения. Данный метод вызывается на инициирующем узле для двустороннего типа связи.

Параметры

Параметр «payloadLen» показывает общий размер полезной нагрузки.

Параметр «cacheLen» показывает число байтов, которое может быть немедленно считано.

Параметр «status» предоставляет код ошибки для причины, по которой был сделан запрос «notifyRsp ( )».

Возвращаемый результат: код ошибки, возвращаемый уровню ИИЭР 1451.X. Как правило, данное возвращаемое значение игнорируется уровнем ИИЭР 1451.X.

11.9 Интерфейс «IEEE1451Dot0::ModuleCommunication::NetReceive»

IDL: `interface NetReceive { }.`

Интерфейс «NetReceive» является набором методов, которые обеспечиваются уровнем ИИЭР 1451.0 и вызываются уровнем ИИЭР 1451.X. Методы, используемые при применении данного интерфейса, перечислены в таблице 103.

Таблица 103 — Методы интерфейса «NetReceive»

IEEE1451Dot0::ModuleCommunication::NetReceive
Args::UInt16 abort(in Args::UInt16 commId, in Args::UInt16 status);
Args::UInt16 notifyMsg(in Args::UInt16 rcvCommId, in Args::_Boolean twoWay, in Args::UInt32 payloadLen, in Args::UInt32 cacheLen, in Args::UInt16 maxPayloadLen, in Args::UInt16 status);
Args::UInt16 notifyRsp(in Args::UInt16 rcvCommId, in Args::UInt16 msgId, in Args::UInt32 payloadLen, in Args::UInt32 cacheLen, in Args::UInt16 maxPayloadLen, in Args::UInt16 status);

11.9.1 Метод «IEEE1451Dot0::ModuleCommunication::NetReceive::abort»

IDL:
```
Args::UInt16 abort(
in Args::UInt16 commId,
in Args::UInt16 status).
```

Данный метод вызывается уровнем ИИЭР 1451.X для прерывания текущей операции. Данный метод вызывается уровнем ИИЭР 1451.X принимающего узла, когда выполнение команды уровня ИИЭР 1451.0 уже было запущено и должно быть прекращено.

В случае если аргумент «commId» — «ноль», данный запрос вызовет прерывание всех активных операций.

Параметры

Параметр «commId» задает активный канал связи.

Параметр «status» предоставляет код ошибки для причины, по которой был сделан запрос «notifyRsp ( )».

Возвращаемый результат: код ошибки, возвращаемый уровню ИИЭР 1451.X. Как правило, данное возвращаемое значение игнорируется уровнем ИИЭР 1451.X.

11.9.2 Метод «IEEE1451Dot0::ModuleCommunication::NetReceive::notifyMsg»

IDL:
```
Args::UInt16 notifyMsg(
in Args::UInt16   rcvCommId,
in Args::_Boolean twoWay,
in Args::UInt32   payloadLen,
in Args::UInt32   cacheLen,
in Args::UInt16   maxPayloadLen,
in Args::UInt16   status).
```

Данный метод обеспечивается уровнем ИИЭР 1451.0 и вызывается уровнем ИИЭР 1451.X при доступности входящего сообщения. Данный метод вызывается на принимающем узле.

Параметры

Параметр «rcvCommId» задает активный канал связи. Следует отметить, что уровень ИИЭР 1451.0 принимающего узла не делает запросы «open( )» или «close( )» для данного «rcvCommId». Управление параметром «rcvCommId» осуществляется на уровне ИИЭР 1451.X.

Параметр «twoWay» имеет значение «True» («Истина»), когда ожидается ответ на данную транзакцию.

Параметр «payloadLen» показывает общий размер полезной нагрузки.

Параметр «cacheLen» показывает число байтов, которое может быть немедленно считано.

Параметр «maxPayloadLen» показывает максимальный размер полезной нагрузки, которая будет принята при последующих операциях «readMsg( )» и «writeRsp( )».

Параметр «status» предоставляет код ошибки для причины, по которой был сделан запрос «notifyRsp ( )».

Возвращаемый результат: код ошибки, возвращаемый уровню ИИЭР 1451.X. Как правило, данное возвращаемое значение игнорируется уровнем ИИЭР 1451.X.

11.9.3 Метод «IEEE1451Dot0::ModuleCommunication::NetReceive::notifyRsp»

IDL:

```
Args::UInt16 notifyRsp(
in Args::UInt16 rcvCommId,
in Args::UInt16 msgId,
in Args::UInt32 payloadLen,
in Args::UInt32 cacheLen,
in Args::UInt16 maxPayloadLen,
in Args::UInt16 status).
```

Данный метод обеспечивается уровнем ИИЭР 1451.0 и вызывается уровнем ИИЭР 1451.X при доступности входящего ответного сообщения. Данный метод вызывается на инициирующем узле для двустороннего типа связи.

Примечание — Уровень ИИЭР 1451.0 принимающего узла не осуществляет запрос «open( )», поэтому при данном вызове автоматически предоставляется «rcvCommId». Управление данным параметром «rcvCommId» осуществляется уровнем ИИЭР 1451.X.

В случае двустороннего типа связи принимающий уровень ИИЭР 1451.0 сделает запрос «writeRsp( )» и в конечном счете «close( )» для завершения транзакции с данной стороны.

Параметры

Параметр «rcvCommId» задает активный канал связи. Следует отметить, что уровень ИИЭР 1451.0 принимающего узла не делает запросы «open( )» или «close( )» для данного «rcvCommId». Управление параметром «rcvCommId» осуществляется на уровне ИИЭР 1451.X.

Параметр «msgId» передается уровню ИИЭР 1451.0 для ассоциации данного ответа с соответствующим вызовом «writeMsg( )».

Параметр «payloadLen» показывает общий размер полезной нагрузки.

Параметр «cacheLen» показывает число байтов, которое может быть немедленно считано.

Параметр «maxPayloadLen» показывает максимальный размер полезной нагрузки, которая будет принята при последующих операциях «readRsp( )».

Параметр «status» предоставляет код ошибки для причины, по которой был сделан запрос «notifyRsp ( )».

Возвращаемый результат: код ошибки, возвращаемый уровню ИИЭР 1451.X. Как правило, данное возвращаемое значение игнорируется уровнем ИИЭР 1451.X.

12 Протокол HTTP

HTTP — это протокол, который используется для передачи или перемещения информации в Интернет. HTTP является протоколом на основе технологии «клиент — сервер», с помощью которого два процессора могут обмениваться информацией через соединение TCP/IP[1)]. HTTP-сервер представляет собой программу, которая хранится в процессоре и получает информацию через порт для HTTP-запросов. HTTP-клиент осуществляет соединение TCP/IP с сервером через сокет, передает запрос, а затем ожидает ответа от сервера. В настоящем стандарте модель «клиент — сервер», которая используется для определения протокола HTTP, аналогична модели «СПП — конечный пользователь», описанной в стандарте ИИЭР 1451.0. СПП можно сравнить с «сервером», так как он предоставляет данные в присоединенную к нему сеть, а конечного пользователя можно сравнить с «клиентом», так как конечный пользователь получает данные датчиков для просмотра от сервера и посылает команды и данные на сервер для управления исполнительными устройствами.

Запрос HTTP-клиента. HTTP-клиент отправляет сообщение-запрос, отформатированное в соответствии с правилами стандарта HTTP, — HTTP-запрос. В данном сообщении указывается ресурс, который клиент желает найти, или содержится информация для предоставления на сервер.

[1)] Transmission Control Protocol (TCP) и Internet Protocol (IP).

Ответ HTTP-сервера. Сервер считывает и интерпретирует запрос. Сервер осуществляет действия, относящиеся к запросу, и создает ответное сообщение в формате HTTP, которое отсылается обратно клиенту. В ответном сообщении указывается, успешно ли выполнен запрос, и при необходимости может содержаться запрошенная клиентом информация данного ресурса.

В протоколе HTTP определены восемь методов, отобажающих требуемые действия, которые должны быть выполнены на заданном ресурсе. Этими методами являются: «GET» («Получить»), «POST» («Отправить»), «HEAD» («Заголовок»), «PUT» («Поместить»), «DELETE» («Удалить»), «TRACE» («Отследить») и «OPTIONS» («Опции»). Представленные в настоящем разделе API используют только методы «GET» («Получить») и «POST» («Отправить»).

Метод «GET» («Получить»). Получение какой-либо информации осуществляется с помощью URI-запроса. Данный процесс аналогичен работе с формой запроса. Взаимодействие сводится к формированию вопроса, схожего с запросом, считыванием или поиском информации. Метод «GET» («Получить») можно использовать для получения данных с сервера. В этом API метод «GET» («Получить») может использоваться для считывания и записи данных преобразователя и ЭТДП преобразователя.

Метод «POST» («Отправить»). Данный метод используется для запроса о том, принимает ли исходный сервер данные, заключенные в тело запроса, в качестве новых зависимых данных ресурса, определенного URI-запросом в строке запроса. В случае с протоколом HTTP данный процесс похож на работу с представлением сложных данных. Взаимодействие больше похоже на приказ и приводит к изменению состояния ресурса в соответствии с восприятием пользователя (например, подписка на услугу) или с учетом ответственности пользователя за результаты взаимодействия. Данный метод может быть использован для получения команды или обязательства от пользователя. Он часто используется для передачи данных различных форм на сервер. В данном API метод «POST» («Отправить») может использоваться для считывания и записи данных преобразователей и ЭТДП.

12.1 API на основе протокола HTTP стандарта ИИЭР 1451.0

Сервисный интерфейс преобразователя (TSI) представляет собой API, предназначенный только для СПП, который используется приложениями для измерения и контроля для получения доступа к уровню ИИЭР 1451.0. Данный API содержит операции считывания и записи каналов преобразователей, считывания и записи ЭТДП, отправки команд конфигурации, контроля и управления в ИМП. Сервисный интерфейс преобразователя (TSI) уровня ИИЭР 1451.0 содержит пять интерфейсов: «TransducerAccess» («Доступ к преобразователю»), «TransducerManager» («Управление преобразователем»), «TIMDiscovery» («Обнаружение ИМП»), «TEDSManager» («Управление ЭТДП») и «AppCallback» («Обратный вызов приложения»). Первые четыре интерфейса реализуются уровнем ИИЭР 1451.0 и вызываются приложениями для измерений. Если для приложения требуются расширенные дополнительные функции, то в нем должен быть реализован интерфейс «AppCallback» («Обратный вызов приложения»), который будет вызываться уровнем ИИЭР 1451.0.

«Discovery» («Обнаружение»). Данный интерфейс содержит методы, предназначенные для приложений и служащие для обнаружения доступных модулей связи уровня ИИЭР 1451.X, ИМП и каналов преобразователя.

«TransducerAccess» («Доступ к преобразователю»). Методы данного интерфейса используются приложениями, которым нужно получить доступ к преобразователю и исполнительному устройству каналов преобразователя.

«TransducerManager» («Управление преобразователем»). Методы данного интерфейса используются приложениями, которым требуется дополнительный контроль над доступом к ИМП, например, для блокировки ИМП для исключительного использования и отправки ИМП произвольных команд.

«TEDSManager» («Управление ЭТДП»). Методы данного интерфейса используются приложениями для считывания и записи ЭТДП. Данный класс также управляет информацией кэш-памяти ЭТДП со стороны СПП.

«AppCallback» («Обратный вызов приложения»). Данный интерфейс реализуется в приложениях, которые требуют расширенных функций. Например, данный интерфейс позволяет приложению настроить потоки измерений, и уровень ИИЭР 1451.0 будет запускать соответствующие обратные вызовы в данном приложении.

Примечание — В настоящем стандарте «AppCallback» API (интерфейс «обратного вызова приложения») не рассматривается.

API на основе протокола HTTP стандарта ИИЭР 1451.0 ориентирован в основном на доступ к данным преобразователя и ЭТДП с использованием протокола HTTP 1.1. На рисунке 23 показано, как с помощью протокола HTTP 1.1 осуществляется доступ к СПП уровня 1451.0, на котором запущен HTTP-сервер. На рисунке 23 обозначение «S» в функциональном блоке ИМП означает «датчик», обозначение «A» означает «исполнительное устройство». Пользователи могут отправить HTTP-запрос на HTTP-сервер на СПП и получить HTTP-ответ с HTTP-сервера. Процесс «запрос — ответ» может быть описан следующим образом:

a) пользователь или клиент отправляет HTTP-запрос на HTTP-сервер в СПП уровня ИИЭР 1451.0;

b) HTTP-сервер в СПП получает HTTP-запрос, обрабатывает его и затем вызывает соответствующий API уровня ИИЭР 1451.0;

c) API уровня ИИЭР 1451.0 вызывает API уровня ИИЭР 1451.X для взаимодействия с ИМП уровня ИИЭР 1451.X и получения результатов от ИМП;

d) HTTP-сервер в СПП получает результаты от API уровня ИИЭР 1451.0 и затем возвращает HTTP-ответ пользователю.

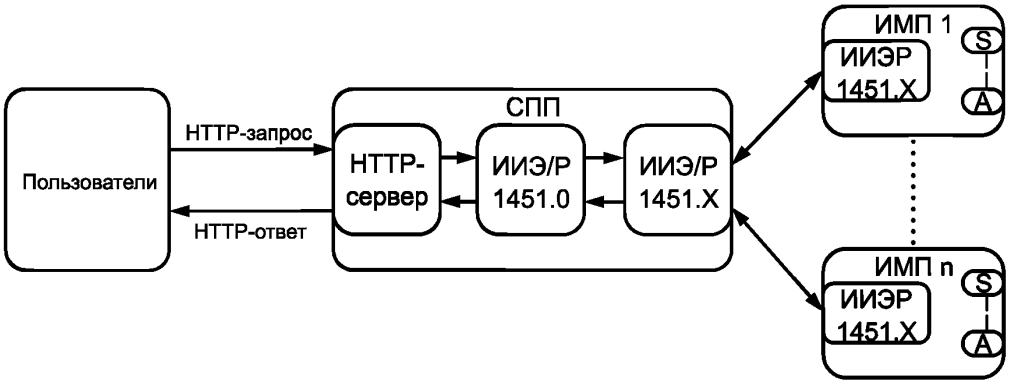


Рисунок 23 — Доступ к СПП уровня ИИЭР 1451.0
на основе протокола HTTP

12.1.1 Формат HTTP-сообщения

В настоящем разделе описан порядок использования протокола HTTP для передачи сообщений от удаленного клиента к СПП уровня ИИЭР 1451.0, который действует как сервер для данной модели, предоставляя данные преобразователя удаленному клиенту. В таблице 104 представлен формат HTTP-сообщения, в таблице 105 приведены примеры возможных аргументов, а в таблице 106 перечислены API на основе протокола HTTP. HTTP-сообщение от модуля удаленного клиента передается по сети, соединяющей удаленного клиента с узлами преобразователя уровня ИИЭР 1451.X. Сообщение должно соответствовать синтаксису URL HTTP (RFC 2616), как указано ниже:

```
http://<host>:<port>/<path>?<parameters>
```

Таблица 104 — Формат HTTP-сообщений для обмена информацией

Поле	Определение	Пример
<host>:	Часть «host» («хост») данной строки включает в себя требуемое доменное имя узла ИИЭР 1451	<host> = “192.168.1.91”
<port>	Номер порта является необязательным, по умолчанию используется порт № 80, если иное не указано в запросе. Может оказаться полезным выбор неиспользуемого номера порта (не порта № 80), который станет основным портом стандарта ИИЭР 1451. Недостатком использования нестандартного номера порта является то, что многие маршрутизаторы безопасности допускают использование нестандартных номеров портов только при явном изменении правил	<port>=”80”

Окончание таблицы 104

Поле	Определение	Пример
<path>	«path» («путь») указывает путь ИИЭР 1451, включающий саму команду	<path>="1451/TransducerAccess/ReadData"
<parameters>	Параметры, связанные с командой	<parameters> ="timId=1&channelId =2&timeout=14&samplingMode= continuous&format=text"

Таблица 105 — Возможные параметры

Поле	Определение	Пример
timId	Идентификатор выбранного ИМП (см. 10.1.2)	1
channelId	Идентификатор выбранного канала преобразователя ИМП (см. 10.2.1)	2
timeout	Данный аргумент определяет время ожидания считывания до появления ошибки тайм-аута	14
samplingMode	Данный аргумент определяет механизм выборки (см. 5.10.1 и 7.1.2.4)	Continuous
format	«format» («формат») определяет желаемый формат возвращаемого значения. Варианты формата: «text» («текст»), «HTML» или «XML». Содержимое строки нечувствительно к регистру	text

Таблица 106 — API на основе протокола HTTP

Тип API	Имя	Путь
API обнаружения	TIMDiscovery (Обнаружение ИМП)	1451/Discovery/TIMDiscovery
	TransducerDiscovery (Обнаружение преобразователя)	1451/Discovery/TransducerDiscovery
	ReadData (Считать данные)	1451/Discovery/ReadData
	StartReadData (Начать считывание данных)	1451/Discovery/StartReadData
	MeasurementUpdate (Обновление измерения)	1451/Discovery/MeasurementUpdate
	WriteData (Записать данные)	1451/Discovery/WriteData
	StartWriteData (Начать запись данных)	1451/Discovery/StartWriteData
API управления ЭТДП	ReadTeds (Считать ЭТДП)	1451/TEDSManager/ReadTeds
	ReadRawTeds (Считать исходную ЭТДП)	1451/TEDSManager/ReadRawTeds
	WriteTeds (Записать ЭТДП)	1451/TEDSManager/WriteTeds
	WriteRawTeds (Записать исходную ЭТДП)	1451/TEDSManager/WriteRawTeds
	UpdateTedsCache (Обновить кэш ЭТДП)	1451/TEDSManager/UpdateTedsCache
API управления преобразователем	SendCommand (Отправить команду)	1451/TransducerManager/SendCommand
	StartCommand (Запустить команду)	1451/TransducerManager/StartCommand
	CommandComplete (Завершить команду)	1451/TransducerManager/CommandComplete
	Trigger (Триггер)	1451/TransducerManager/Trigger
	StartTrigger (Запустить триггер)	1451/TransducerManager/StartTrigger

12.1.2 Формат HTTP-ответа

Выходная спецификация или выходные аргументы, например массив аргументов ArgumentArray, могут быть включены в HTTP-запрос или в ответ на HTTP-запрос. Выходной формат может быть в форматах «XML», «HTML» или «text» («текст»).

Формат ответа определяется аргументом «format» («формат») HTTP-сообщения. Ответы должны быть отформатированы в соответствии с правилами, указанными в настоящем разделе.

12.1.2.1 Формат XML ответа

Если указан формат XML, то ответ должен быть отформатирован в соответствии с XML-схемой, связанной с данной командой. Форматирование, указанное в командах, опирается на библиотеку типов, основанную на типах данных, определенных в разделе 4. Полное описание XML-схемы доступно по адресу http://grouper.ieee.org/groups/1451/0/1451HTTPAPI/. Имя файла SmartTransducerHTTPResponse.xsd.

12.1.2.2 Формат HTML ответа

Если указан формат HTML, то ответ должен быть отформатирован как действительная веб-страница в формате HTTP 1.1. Форматирование веб-страницы и верстка не приводятся. Тем не менее все параметры, которые определяют ответы, должны использовать формат тегированных данных, тег которых совпадает с идентификатором возвращаемого ответа, указанным в возвращенном формате команды.

12.1.2.3 Текстовый ответ

Текстовый ответ должен быть представлен в формате, в котором возвращаются отдельные параметры, и должен быть ограничен управляющим символом CR/LF (возврат каретки/перевод строки) (ASCII 13,10). Значения должны быть возвращены в порядке, установленном семантикой команды.

Если в качестве ответа возвращается массив, то каждое значение должно быть отделено запятой и весь массив должен завершаться управляющим символом CR/LF (возврат каретки/перевод строки). Многомерный массив должен быть возвращен с крайним правым порядковым числительным, которое индексируется в первую очередь. Управляющий символ CR/LF (возврат каретки/перевод строки) должен быть возвращен после возвращения каждого полного набора крайних правых индексов. Данный формат обычно называется форматом «csv».

Целые числа должны быть возвращены в формате `<Sign><Value>`, где `<Sign>`= «+» или «–», а `<Value>` определяется как строка, содержащая один или более символов, значение которых от «0» до «9».

Числа с плавающей точкой должны быть возвращены в экспоненциальном представлении в формате

`<Sign><Leading digit>.<Mantissa>E<Sign><Exponent>`,

где `<Sign>` — определено выше;

`<Leadingdigit>` — одиночный символ, как определено для `<Value>` (см. выше);

`<Mantissa>` и `<Exponent>` — аналогично `<Value>` (см. выше).

Любое возвращаемое значение, которое представляет собой нумерацию, должно возвращать свое порядковое значение в виде целого числа.

Любое возвращаемое значение, которое представляет собой строку, должно возвращать данную строку, заключенную в кавычки (“<string>”). Встроенные кавычки должны быть возвращены в качестве удвоенных кавычек (“”).

12.2 API обнаружения

API обнаружения предназначен для выявления всех доступных ИМП и каналов преобразователей (преобразователей).

12.2.1 Интерфейс «TIMDiscovery» («Обнаружение ИМП»)

Данный API поддерживает выдачу отчетов обо всех «timIds» всех ИМП, подключенных к данному СПП (хосту). Данный API соответствует `Args::UInt16 reportTims()`, описанному в 10.1.2.

Путь «Path»: 1451/Discovery/TIMDiscovery.

Метод «GET» («Получить»): находит «timIds» всех ИМП, подключенных к СПП (хосту), и возвращает «timIds» в формате XML-документа или формате ASCII.

12.2.1.1 Входные параметры

Следующий параметр должен поставляться вместе с вызовом данного API:

`_String responseFormat` — указывает формат ответа, как это определено в 12.1.2.

12.2.1.2 HTTP-ответ обнаружения ИМП «TIMDiscoveryHTTPResponse»

Ответ на данный вызов API должен содержать следующие параметры:

- `UInt16 errorCode` — информация об ошибке, как определено в 9.3.1.2;
- `UInt16Array timeIds` — значения «timIds» всех ИМП, доступных для данного СПП.

12.2.1.3 XML-схема ответа «TIMDiscovery» («Обнаружение ИМП»)

Если формат ответа — «XML», то для ответа должна использоваться следующая схема:

```
<?xml version=»1.0» encoding=»UTF-8»?>
<xs:schema xmlns:xs=»http://www.w3.org/2001/XMLSchema»
xmlns:stml=http://grouper.ieee.org/groups/1451/0/1451HTTPAPI
<xs:complexType name="TIMDiscoveryHTTPResponse">
    <xs:sequence>
        <xs:element name="errorCode" type="stml:UInt16"/>
        <xs:element name="timIds" type="stml:UInt16Array"/>
    </xs:sequence>
</xs:complexType>
</xs:schema>
```

12.2.1.4 Интерфейс «TransducerDiscovery» («Обнаружение преобразователя»)

Данный API поддерживает выдачу отчетов обо всех «channelIds» всех преобразователей, доступных в заданном ИМП заданного СПП (хоста). Он получает список преобразователей и их имена для данного ИМП. Данная информация извлекается из ЭТДП, хранящихся в кэш-памяти СПП. Данный API соответствует интерфейсу `Args::Uint16 reportChannels()`, как описано в 10.1.3.

Путь «Path»: 1451/Discovery/TransducerDiscovery.

Метод «GET» («Получить»): получает значения «channelIds» всех преобразователей, имеющихся в заданном ИМП СПП (хоста), и сообщает о «channelIds» в заданном формате.

12.2.1.5 Входные параметры

Следующие параметры должны поставляться вместе с вызовом данного API:

- `UInt16 timId` — значение «timId» заданного ИМП;
- `_string ResponseFormat` — задает формат ответа, как определено в 12.1.2.

12.2.1.6 HTTP-ответ API обнаружения преобразователя «TransducerDiscoveryHTTPResponse»

Ответ на данный вызов API должен содержать следующие параметры:

- `UInt16 errorCode` — информация об ошибке;
- `UInt16 timId` — значение «timId» заданного ИМП;
- `UInt16Array channelIds` — список «channelIds» всех преобразователей (датчиков и исполнительных устройств), доступных в заданном ИМП;
- `StringArray transducerNames` — список имен всех преобразователей (датчиков и исполнительных устройств), доступных в заданном ИМП.

12.2.1.7 XML-схема ответа API «TransducerDiscovery» («Обнаружение преобразователя»)

Если в качестве формата ответа используется «XML», то для ответа должна быть использована следующая схема:

```
<?xml version=»1.0» encoding=»UTF-8»?>
<xs:schema xmlns:xs=»http://www.w3.org/2001/XMLSchema»
xmlns:stml=http://grouper.ieee.org/groups/1451/0/1451HTTPAPI
<xs:complexType=»TransducerDiscoveryHTTPResponse»>
    <xs:sequence>
        <xs:element name=»errorCode» type=»stml:UInt16»/>
        <xs:element name=»timId» type=»stml:UInt16»/>
        <xs:element name=»channelIds»type=»stml:UInt16Array»/>
        <xs:element name="transducerNames" type="stml:StringArray"/>
    </xs:sequence>
</xs:complexType>
</xs:schema>
```

12.3 API доступа к преобразователю

API доступа к преобразователю ориентирован на считывание и запись преобразователей (датчиков и исполнительных устройств) или каналов преобразователя.

12.3.1 API считывания данных преобразователя

API считывания преобразователя используется для считывания данных (значения) преобразователя.

12.3.1.1 Интерфейс «ReadData» («Считать данные»)

Данный API поддерживает получение данных преобразователей от канала преобразователя с заданным значением «channelID» на ИМП, заданном с помощью «timId» на заданном СПП (хосте).

Данный API соответствует `Args::UInt16 readData()`, как описано в 10.2.6.

Путь «Path»: 1451/TransducerAccess/ReadData.

Метод «GET» («Получить»): получает данные заданного канала преобразователя в заданном ИМП СПП (хоста) и возвращает данные преобразователя в заданном формате.

12.3.1.1.1 Входные параметры

Следующие параметры должны поставляться вместе с вызовом данного API:

- `UInt16 timId` — значение «timId» ИМП, содержащего канал преобразователя, данные которого будут считываться;
- `UInt16 channelID` — значение «channelID» считываемого канала преобразователя;
- `TimeDuration timeout` — данный аргумент определяет время ожидания для выполнения считывания без генерации ошибки тайм-аута в случае неполучения ответа. Значения «secs==0» и «nsecs==–1» указывают на непрерывное (бесконечное) время ожидания. Использование значений «непрерывное (бесконечное) время ожидания» является крайне опасным, поскольку при этом ресурс может быть заблокирован;
- `Uint8 SamplingMode` — данный аргумент задает триггерный механизм. Подробная информация представлена в 5.10.1 и 7.1.2.4;
- `_string ResponseFormat` — задает формат ответа согласно 12.1.2.

12.3.1.1.2 HTTP-ответ для API «ReadData» («Считать данные») «ReadDataHTTPResponse»

Ответ на вызов данного API должен содержать следующие параметры:

- `UInt16 errorCode` — информация об ошибках, как определено в 9.3.1.2;
- `UInt16 timId` — значение «timId» заданного ИМП;
- `UInt16 channelID` — значение «channelID» заданного канала преобразователя;
- `ArgumentArray transducerData` — данный массив содержит данные, считанные из заданного канала преобразователя выбранного ИМП.

12.3.1.1.3 Схема XML-ответа для API «ReadData» («Считать данные»)

Если формат ответа — «XML», то для ответа должна быть использована следующая схема:

```
<?xml version=»1.0» encoding=»UTF-8»?>
<xs:schema xmlns:xs=»http://www.w3.org/2001/XMLSchema»
xmlns:stml=http://grouper.ieee.org/groups/1451/0/1451HTTPAPI
<xs:complexType name="ReadDataHTTPResponse">
    <xs:sequence>
            <xs:element name="errorCode" type="stml:UInt16"/>
            <xs:element name="timId" type="stml:UInt16"/>
            <xs:element name="channelId" type="stml:UInt16"/>
            <xs:element name="transducerData" type="stml:ArgumentArrayType"/>
    </xs:sequence>
</xs:complexType>
</xs:schema>
```

12.3.1.2 Интерфейс «StartReadData» («Начать считывание данных»)

Данный API поддерживает начало получения данных заданного преобразователя заданного ИМП для заданного СПП (хоста). Данный метод запускает неблокирующее считывание заданных каналов преобразователя. Данный API соответствует `Args::UInt16 startReadData()`, как описано в 10.2.8. Данные преобразователя, которые должны быть переданы, завершаются вызовом API «MeasurementUpdate» («Обновить измерения»).

Путь «Path»: 1451/TransducerAccess/StartReadData.

Метод «GET» («Получить»): запускает извлечение данных преобразователя, доступного для заданного ИМП СПП (хоста), и возвращает код ошибки в заданном формате.

12.3.1.2.1 Входные параметры

Следующие параметры должны поставляться вместе с вызовом данного API:

- `UInt16 timId` — значение «timId» ИМП, содержащего канал преобразователя для считывания;

- UInt16 ChannelID — значение «ChannelID» заданного канала преобразователя;
- TimeInstance triggerTime — данный аргумент указывает время начала операции считывания;
- TimeDuration timeout — данный аргумент определяет время ожидания для выполнения считывания без генерации ошибки тайм-аута в случае неполучения ответа. Значения «secs==0» и «nsecs==–1» указывают на непрерывное (бесконечное) время ожидания. Использование значений «непрерывное (бесконечное) время ожидания» является крайне опасным, поскольку при этом ресурс может быть заблокирован;
- Uint8 SamplingMode — данный аргумент задает триггерный механизм. Подробная информация представлена в 5.10.1 и 7.1.2.4;
- _string ResponseFormat — задает формат ответа, как это определено в 12.1.2.

12.3.1.2.2 HTTP-ответ для API «StartReadData» («Начать считывание данных»)

Ответ на вызов данного API должен содержать следующие параметры:
- UInt16 errorCode — информация об ошибках, как определено в 9.3.1.2;
- UInt16 timId — значение «timId» заданного ИМП;
- UInt16 channelID — значение «channelID» заданного канала преобразователя.

12.3.1.2.3 Схема XML-ответа для API «StartReadData» («Начать считывание данных»)

Если формат ответа — «XML», то для ответа должна быть использована следующая схема:

```
<?xml version=»1.0» encoding=»UTF-8»?>
<xs:schema xmlns:xs=»http://www.w3.org/2001/XMLSchema»
xmlns:stml=http://grouper.ieee.org/groups/1451/0/1451HTTPAPI
<xs:complexType name="StartReadDataHTTPResponse">
    <xs:sequence>
        <xs:element name="errorCode" type="stml:UInt16"/>
        <xs:element name="timId" type="stml:UInt16"/>
        <xs:element name="channelId" type="stml:UInt16"/>
    </xs:sequence>
</xs:complexType>
</xs:schema>
```

12.3.1.3 Интерфейс «MeasurementUpdate» («Обновление измерения»)

В данном API неблокирующее считывание, инициируемое вызовом «StartReadData» («Начать считывание данных»), завершается получением данных заданного преобразователя для заданного ИМП заданного СПП (хоста). Данный API соответствует Args::UInt16 measurementUpdate(), как описано в 10.6.1.

Путь «Path»: 1451/TransducerAccess/MeasurementUpdate.

Метод «GET» («Получить»): извлекает данные преобразователя, доступного для заданного ИМП СПП (хоста), и возвращает данные преобразователя в заданном формате.

12.3.1.3.1 Входные параметры

Следующие параметры должны поставляться вместе с вызовом данного API:
- UInt16 timId — значение «timId» ИМП, содержащего канал преобразователя для считывания;
- UInt16 channelId — значение «ChannelID» заданного канала преобразователя;
- _String responseFormat — указывает формат ответа.

12.3.1.3.2 HTTP-ответ для API «Measurement Update» («Обновление измерения»)

Ответ на вызов данного API должен содержать следующие параметры:
- UInt16 errorCode — информация об ошибке, как определено в 9.3.1.2;
- UInt16 timId — значение «timId» заданного ИМП;
- UInt16 channelID — значение «channelID» заданного канала преобразователя;
- ArgumentArray transducerData — данный массив содержит данные, считанные из заданного канала преобразователя выбранного ИМП.

12.3.1.3.3 Схема XML-ответа для API «Measurement Update» («Обновление измерения»)

Если формат ответа — «XML», то для ответа должна быть использована следующая схема:

```
<?xml version=»1.0» encoding=»UTF-8»?>
<xs:schema xmlns:xs=»http://www.w3.org/2001/XMLSchema»
xmlns:stml=http://grouper.ieee.org/groups/1451/0/1451HTTPAPI
<xs:complexType name="MeasurementUpdateHTTPResponse">
    <xs:sequence>
        <xs:element name="errorCode" type="stml:UInt16"/>
```

```
            <xs:element name="timId" type="stml:UInt16"/>
            <xs:element name="transducerId" type="stml:UInt16"/>
            <xs:element name="transducerData" type="stml:ArgumentArrayType"/>
    </xs:sequence>
</xs:complexType>
</xs:schema>
```

12.3.2 Интерфейс «TransducerWriteDataAPI» («API записи данных преобразователя»)

API записи преобразователя используется для записи данных (значения) канала преобразователя.

12.3.2.1 Интерфейс «WriteData» («Записать данные»)

В данном API осуществляется запись данных преобразователя заданного преобразователя для заданного ИМП заданного СПП (хоста). Данный метод выполняет блокирующую запись заданных каналов преобразователя или преобразователей. Данный API соответствует `Args::UInt16 writeData()`, как описано в 10.2.7.

Путь «Path»: 1451/TransducerAccess/WriteData.

Метод «POST» («Отправить»): записывает данные преобразователя в назначенный канал преобразователя для заданного ИМП, подключенного к СПП (хосту).

12.3.2.1.1 Входные параметры

Следующие параметры должны поставляться вместе с вызовом данного API:

- `UInt16 timId` — значение «timId» ИМП, содержащего канал преобразователя для записи;
- `UInt16 channelId` — значение «ChannelID» заданного канала преобразователя;
- `TimeDuration timeout` — данный аргумент определяет время ожидания после записи данных без генерации ошибки тайм-аута в случае неполучения ответа. Значения «secs==0» и «nsecs==–1» указывают на непрерывное (бесконечное) время ожидания. Использование значений «непрерывное (бесконечное) время ожидания» является крайне опасным, поскольку при этом ресурс может быть заблокирован;
- `UInt8 SamplingMode` — данный аргумент задает механизм осуществления выборки. Подробная информация представлена в 5.10.1 и 7.1.2.4;
- `ArgumentArray transducerData` — данный массив содержит данные для записи в заданный канал преобразователя выбранного ИМП;
- `_String responseFormat` — задает формат ответа, как это определено в 12.1.2.

12.3.2.1.2 HTTP-ответ для API «WriteData» («Записать данные»)

Ответ на вызов данного API должен содержать следующие параметры:

- `UInt16 errorCode` — информация об ошибке, как определено в 9.3.1.2;
- `UInt16 timId` — значение «timId» заданного ИМП;
- `UInt16 channelId` — значение «channelID» заданного канала преобразователя.

12.3.2.1.3 Схема XML-ответа для API «WriteData» («Записать данные»)

Если формат ответа — «XML», то для ответа должна быть использована следующая схема:

```
<?xml version=»1.0» encoding=»UTF-8»?>
<xs:schema xmlns:xs=»http://www.w3.org/2001/XMLSchema»
xmlns:stml=http://grouper.ieee.org/groups/1451/0/1451HTTPAPI
<xs:complexType name="WriteDataHTTPResponse">
    <xs:sequence>
            <xs:element name="errorCode" type="stml:UInt16"/>
            <xs:element name="timId" type="stml:UInt16Array"/>
            <xs:element name="channelId" type="stml:UInt16"/>
    </xs:sequence>
</xs:complexType>
</xs:schema>
```

12.3.2.2 Интерфейс «StartWriteData» («Запустить запись данных»)

В данном API осуществляется запись данных преобразователя заданного преобразователя для заданного ИМП заданного СПП (хоста). Данный метод выполняет неблокирующую запись указанного канала преобразователя. Пользователь несет ответственность за определение завершения команды путем отправки вызова «SendCommand» («Отправить команду») (см. 12.5.1) с заданием команды «ReadStatusEventRegister» («Считать регистр статуса события») (см. 7.1.1.8) и проверки бита «DataProcessed» («Данные обработаны») (см. 5.13.10), который должен быть установлен. Данный API соответствует `Args::UInt16 startWriteData()`, как описано в 10.2.9.

Путь «Path»: 1451/TransducerAccess/StartWriteData.

Метод «POST» («Отправить»): записывает данные преобразователя в назначенный канал преобразователя в заданном ИМП, подключенном к СПП (хосту), и возвращает результат в заданном формате.

12.3.2.2.1 Входные параметры

Следующие параметры должны поставляться вместе с вызовом данного API:

- `UInt16 timId` — значение «timId» ИМП, содержащего канал преобразователя для считывания;
- `UInt16 channelId` — значение «ChannelID» заданного преобразователя;
- `TimeInstance triggerTime` — данный аргумент задает время, когда следует начать операцию записи;
- `TimeDuration timeout` — данный аргумент определяет время ожидания после записи данных без генерации ошибки тайм-аута в случае неполучения ответа. Значения «secs==0» и «nsecs==–1» указывают на непрерывное (бесконечное) время ожидания. Использование значений «непрерывное (бесконечное) время ожидания» является крайне опасным, поскольку при этом ресурс может быть заблокирован;
- `UInt8 SamplingMode` — данный аргумент задает триггерный механизм. Подробная информация представлена в 5.10.1 и 7.1.2.4;
- `ArgumentArray transducerData` — данный массив содержит данные, считанные из заданного канала преобразователя выбранного ИМП;
- `_String responseFormat` — задает формат ответа, как это определено в 12.1.2.

12.3.2.2.2 HTTP-ответ для API «StartWriteData» («Начать запись данных»)

Ответ на вызов данного API должен содержать следующие параметры:

- `UInt16 errorCode` — информация об ошибке, как определено в 9.3.1.2;
- `UInt16 timId` — значение «timId» заданного ИМП;
- `UInt16 channelId` — значение «channelID» заданного канала преобразователя.

12.3.2.2.3 Схема XML-ответа для API «StartWriteData» («Начать запись данных»)

Если формат ответа — «XML», то для ответа должна быть использована следующая схема:

```
<?xml version=»1.0» encoding=»UTF-8»?>
<xs:schema xmlns:xs=»http://www.w3.org/2001/XMLSchema»
xmlns:stml=http://grouper.ieee.org/groups/1451/0/1451HTTPAPI
<xs:complexType name="StartWriteDataHTTPResponse">
    <xs:sequence>
           <xs:element name="errorCode" type="stml:UInt16"/>
           <xs:element name="timId" type="stml:UInt16Array"/>
           <xs:element name="channelId" type="stml:UInt16"/>
    </xs:sequence>
</xs:complexType>
</xs:schema>
```

12.4 API управления ЭТДП

API управления ЭТДП включают в себя интерфейсы: «ReadTEDS» («Считать ЭТДП») для считывания ЭТДП и «WriteTEDS» («Записать ЭТДП») для записи ЭТДП.

12.4.1 Интерфейс «ReadTEDS» («Считать ЭТДП»)

Данный API поддерживает получение данных ЭТДП, связанных с указанным каналом преобразователя или ИМП от заданного СПП (хоста). Данный метод осуществляет считывание требуемого блока ЭТДП из кэш-памяти ЭТДП. Если считывание ЭТДП из кэш-памяти недоступно, то метод будет считывать ЭТДП из ИМП. Данный API соответствует `Args::UInt16 readTeds()`, как описано в 10.4.1.

Путь «Path»: 1451/TEDSManager/ReadTEDS.

Метод «GET» («Получить»): извлекает ЭТДП из заданного канала преобразователя или ИМП на СПП (хосте) и возвращает результат в заданном формате.

12.4.1.1 Входные параметры

Следующие параметры должны поставляться вместе с вызовом данного API:

- `UInt16 timId` — значение «timId» ИМП, содержащего канал преобразователя для считывания;
- `UInt16 channelId` — значение «channelID» заданного канала преобразователя. Данный аргумент равен «0», если осуществляется доступ к ЭТДП, связанной со всем ИМП;

- TimeDuration timeout — данный аргумент определяет время ожидания считывания данных без генерации ошибки тайм-аута в случае неполучения ответа. Значения «secs==0» и «nsecs==–1» указывают на непрерывное (бесконечное) время ожидания. Использование значений «непрерывное (бесконечное) время ожидания» является крайне опасным, поскольку при этом ресурс может быть заблокирован;

- UInt8 TEDSType — данный аргумент задает «TEDSType» («тип ЭТДП»), как указано в таблице 17, где он называется «TEDS Access Code» («Код доступа к ЭТДП»);

- _string ResponseFormat — задает формат ответа, как это определено в 12.1.2.

12.4.1.2 HTTP-ответ для API «ReadTEDS» («Считать ЭТДП»)

Ответ на вызов данного API должен содержать следующие параметры:

- UInt16 errorCode — информация об ошибке, как определено в 9.3.1.2;

- UInt16 timId — значение «timId» заданного ИМП;

- UInt16 channelId — значение «channelID» заданного канала преобразователя;

- UInt8 TEDSType — данный аргумент задает «TEDSType» («тип ЭТДП»), как указано в таблице 17, где он называется «TEDS Access Code» («Код доступа к ЭТДП»);

- ArgumentArray teds — данный массив содержит данные, считанные из заданной ЭТДП.

12.4.1.3 Схема XML-ответа для API «ReadTEDS» («Считать ЭТДП»)

Если формат ответа — «XML», то для ответа должна быть использована следующая схема:

```
<?xml version=»1.0» encoding=»UTF-8»?>
<xs:schema xmlns:xs=»http://www.w3.org/2001/XMLSchema»
xmlns:stml=http://grouper.ieee.org/groups/1451/0/1451HTTPAPI
<xs:complexType name="ReadTEDSHTTPResponse">
    <xs:sequence>
            <xs:element name="errorCode" type="stml:UInt16"/>
            <xs:element name="timId" type="stml:UInt16Array"/>
            <xs:element name="channelId" type="stml:UInt16"/>
            <xs:element name="tedsType" type="stml:UInt8"/>
            <xs:element name="teds" type="stml:ArgumentArrayType"/>
    </xs:sequence>
</xs:complexType>
</xs:schema>
```

12.4.2 Интерфейс «ReadRawTEDS» («Считать предварительную ЭТДП»)

Данный API поддерживает извлечение данных предварительной ЭТДП из выбранного канала преобразователя или из ИМП на заданном СПП (хосте). Данный метод считывает требуемый блок ЭТДП из ИМП в обход кэш-памяти ЭТДП в СПП. Кэш ЭТДП не обновляется. Данный API соответствует Args::UInt16readRawTeds(), как описано в 10.4.3.

Для целей настоящего API все ЭТДП имеют бинарные структуры. Для того чтобы кодировать эти структуры в заданный формат, необходимо кодировать их в виде текста. Для достижения этой цели все содержимое ЭТДП должно быть закодировано с использованием кодировки Base64, описанной в RFC 2045 (подраздел 6.8) [B8].

Путь «Path»: 1451/TEDSManager/ReadRawTEDS.

Метод «GET» («Получить»): извлекает предварительную ЭТДП канала преобразователя, доступного для заданного ИМП СПП (хоста), и отображает результат в заданном формате.

12.4.2.1 Входные параметры

Следующие параметры должны поставляться вместе с вызовом данного API:

- UInt16 timId — значение «timId» ИМП, содержащего канал преобразователя для считывания;

- UInt16 channelId — значение «channelID» заданного канала преобразователя. Данный аргумент равен «0», если доступ осуществляется к ЭТДП, связанной со всем ИМП;

- TimeDuration timeout — данный аргумент определяет время ожидания считывания данных без генерации ошибки тайм-аута в случае неполучения ответа. Значения «secs==0» и «nsecs==–1» указывают на непрерывное (бесконечное) время ожидания. Использование значений «непрерывное (бесконечное) время ожидания» является крайне опасным, поскольку при этом ресурс может быть заблокирован;

- UInt8 TEDSType — данный аргумент задает «TEDSType» («тип ЭТДП»), как указано в таблице 17, где он называется «TEDS Access Code» («Код доступа к ЭТДП»);

- _string ResponseFormat — задает формат ответа.

12.4.2.2 HTTP-ответ для API «ReadRawTEDS» («Считать предварительную ЭТДП»)

Ответ на вызов данного API должен содержать следующие параметры:

- `UInt16 errorCode` — информация об ошибке, как определено в 9.3.1.2;
- `UInt16 timId` — значение «timId» заданного ИМП;
- `UInt16 channelId` — значение «channelID» заданного канала преобразователя;
- `UInt8 TEDSType` — данный аргумент задает «TEDSType» («тип ЭТДП»), как указано в таблице 17, где он называется «TEDS Access Code» («Код доступа к ЭТДП»);
- `OctetArray TEDS` — данные предварительной ЭТДП заданного канала преобразователя или ИМП, закодированные с использованием кодировки Base64.

12.4.2.3 Схема XML-ответа для API «ReadRawTEDS» («Считать предварительную ЭТДП»)

Если формат ответа — «XML», то для ответа должна быть использована следующая схема:

```
<?xml version=»1.0» encoding=»UTF-8»?>
<xs:schema xmlns:xs=»http://www.w3.org/2001/XMLSchema»
xmlns:stml=http://grouper.ieee.org/groups/1451/0/1451HTTPAPI
<xs:complexType name="ReadRawTEDSHTTPResponse">
    <xs:sequence>
            <xs:element name="errorCode" type="stml:UInt16"/>
            <xs:element name="timId" type="stml:UInt16Array"/>
            <xs:element name="channelId" type="stml:UInt16"/>
            <xs:element name="tedsType" type="stml:UInt8"/>
            <xs:element name="teds" type="stml:ArgumentArrayType"/>
    </xs:sequence>
</xs:complexType>
</xs:schema>
```

12.4.3 Интерфейс «WriteTEDS» («Записать ЭТДП»)

Данный API используется при записи ЭТДП в заданный канал преобразователя или ИМП заданного СПП (хоста). Данный метод записывает требуемый блок ЭТДП в ИМП. Кэш ЭТДП также обновляется, в случае если запись произведена успешно. Представленная информация ЭТДП кодируется в массив аргументов ArgumentArray. Данный API соответствует `Args::UInt16 writeTeds()`, как описано в 10.4.2.

Путь «Path»: 1451/TEDSManager/WriteTEDS.

Метод «POST» («Отправить»): записывает данные ЭТДП на заданный канал преобразователя или ИМП заданного СПП (хоста) и отображает результат в заданном формате.

12.4.3.1 Входные параметры

Следующие параметры должны поставляться вместе с вызовом данного API:

- `UInt16 timId` — значение «timId» ИМП, содержащего ЭТДП для записи;
- `UInt16 channelId` — значение «channelID» заданного канала преобразователя. Данный аргумент равен «0», если доступ осуществляется к ЭТДП, связанной со всем ИМП;
- `TimeDuration timeout` — данный аргумент определяет время ожидания после записи данных без генерации ошибки тайм-аута в случае неполучения ответа. Значения «secs==0» и «nsecs==–1» указывают на непрерывное (бесконечное) время ожидания. Использование значений «непрерывное (бесконечное) время ожидания» является крайне опасным, поскольку при этом ресурс может быть заблокирован;
- `UInt8 TEDSType` — данный аргумент задает «TEDSType» («тип ЭТДП»), как указано в таблице 17, где он называется «TEDS Access Code» («Код доступа к ЭТДП»);
- `ArgumentArray TEDS` — данные ЭТДП заданного канала преобразователя или ИМП;
- `_string ResponseFormat` — задает формат ответа, как это определено в 12.1.2.

12.4.3.2 HTTP-ответ для API «WriteTEDS» («Записать ЭТДП»)

Ответ на вызов данного API должен содержать следующие параметры:

- `UInt16 errorCode` — информация об ошибке, как определено в 9.3.1.2;
- `UInt16 timId` — значение «timId» ИМП, содержащего ЭТДП для записи;
- `UInt16 channelId` — значение «channelID» заданного канала преобразователя. Данный аргумент равен «0», если доступ осуществляется к ЭТДП, связанной со всем ИМП;
- `UInt8 tedsType` — данный аргумент задает «tedsType» («тип ЭТДП»), как указано в таблице 17, где он называется «TEDS Access Code» («Код доступа к ЭТДП»).

12.4.3.3 Схема XML-ответа для API «WriteTEDS» («Записать ЭТДП»)

Если формат ответа — «XML», то для ответа должна быть использована следующая схема:

```
<?xml version=»1.0» encoding=»UTF-8»?>
<xs:schema xmlns:xs=»http://www.w3.org/2001/XMLSchema»
xmlns:stml=http://grouper.ieee.org/groups/1451/0/1451HTTPAPI
<xs:complexType name="WriteTEDSHTTPResponse">
    <xs:sequence>
          <xs:element name="errorCode" type="stml:UInt16"/>
          <xs:element name="timId" type="stml:UInt16Array"/>
          <xs:element name="tedsId" type="stml:UInt16"/>
          <xs:element name="tedsType" type="stml:UInt8"/>
    </xs:sequence>
</xs:complexType>
</xs:schema>
```

12.4.4 Интерфейс «WriteRawTEDS» («Записать предварительную ЭТДП»)

В данном API осуществляется запись данных «rawTEDS» («Предварительной ЭТДП») для заданного ИМП заданного СПП (хоста). Данный метод записывает нужный блок ЭТДП в ИМП в обход кэш-памяти ЭТДП. Предоставляемая информация ЭТДП кодируется в форму «кортежа» в байтовый массив OctetArray. Данный API соответствует `Args::UInt16 writeRawTeds()`, как описано в 10.4.4.

Для целей настоящего API все ЭТДП имеют бинарные структуры. Для того чтобы кодировать эти структуры в заданный формат, необходимо кодировать их в виде текста. Для достижения этой цели все содержимое ЭТДП должно быть закодировано с использованием кодировки Base64, описанной в RFC 2045 (подраздел 6.8).

Путь «Path»: 1451/TEDSManager/WriteRawTEDS.

Метод «POST» («Отправить»): записывает данные предварительной ЭТДП (rawTEDS) преобразователя, доступного в заданном ИМП СПП (хоста), и отображает результат в заданном формате.

12.4.4.1 Входные параметры

Следующие параметры должны поставляться вместе с вызовом данного API:

- `UInt16 timId` — значение «timId» заданного ИМП;
- `UInt16 channelId` — значение «channelID» заданного канала преобразователя или значение «0», если ЭТДП применяется ко всему ИМП;
- `TimeDuration timeout` — данный аргумент определяет время ожидания после записи данных без генерации ошибки тайм-аута в случае неполучения ответа. Значения «secs==0» и «nsecs==–1» указывают на непрерывное (бесконечное) время ожидания. Использование значения «непрерывное (бесконечное) время ожидания» является крайне опасным, поскольку при этом ресурс может быть заблокирован;
- `UInt8 tedsType` — данный аргумент задает «tedsType» («тип ЭТДП»), как указано в таблице 17, где он называется «TEDS Access Code» («Код доступа к ЭТДП»).
- `OctetArray rawTEDS` — данные предварительной ЭТДП заданного канала преобразователя или заданного ИМП;
- `_string ResponseFormat` — задает формат ответа, как это определено в 12.1.2.

12.4.4.2 HTTP-ответ для API «WriteRawTEDS» («Записать предварительную ЭТДП»)

Ответ на вызов данного API должен содержать следующие параметры:

- `UInt16 errorCode` — информация об ошибке, как определено в 9.3.1.2;
- `UInt16 timId` — значение «timId» заданного ИМП;
- `UInt16 channelId` — значение «channelID» заданного канала преобразователя;
- `UInt8 tedsType` — данный аргумент задает «tedsType» («тип ЭТДП»), как указано в таблице 17, где он называется «TEDS Access Code» («Код доступа к ЭТДП»).

12.4.4.3 Схема XML-ответа для API «WriteRawTEDS» («Записать предварительную ЭТДП»)

Если формат ответа — «XML», то для ответа должна быть использована следующая схема:

```
<?xml version=»1.0» encoding=»UTF-8»?>
<xs:schema xmlns:xs=»http://www.w3.org/2001/XMLSchema»
xmlns:stml=http://grouper.ieee.org/groups/1451/0/1451HTTPAPI
<xs:complexType name="WriteRawTEDSHTTPResponse">
    <xs:sequence>
          <xs:element name="errorCode" type="stml:UInt16"/>
```

```
            <xs:element name="timId" type="stml:UInt16Array"/>
            <xs:element name="channelId" type="stml:UInt16"/>
            <xs:element name="tedsType" type="stml:UInt8"/>
    </xs:sequence>
</xs:complexType>
</xs:schema>
```

12.4.5 Интерфейс «UpdateTEDSCache» («Обновить кэш ЭТДП»)

В данном API осуществляется обновление кэш-памяти ЭТДП заданного канала преобразователя или ИМП для заданного СПП (хоста). Данный API обновляет кэш-память ЭТДП. Контрольная сумма ЭТДП считывается из ИМП и сравнивается с контрольной суммой кэшированной ЭТДП. Если контрольные суммы различаются, то ЭТДП будет считана из ИМП и сохранена в кэш-память. Данный API соответствует `Args::Uint16 updateTedsCache`, как описано в 10.4.5.

Путь «Path»: 1451/TEDSManager/UpdateTEDSCache.

Метод «GET» («Получить»): обновляет кэш ЭТДП заданного преобразователя для заданного ИМП заданного СПП (хоста) и отображает результаты в заданном формате.

12.4.5.1 Входные параметры

Следующие параметры должны поставляться вместе с вызовом данного API:

- `UInt16 timId` — значение «timId» заданного ИМП;
- `UInt16 channelId` — значение «channelID» заданного преобразователя или значение «0», означающее считывание ЭТДП ИМП;
- `TimeDuration timeout` — задает время ожидания считывания без генерации ошибки таймаута;
- `UInt8 tedsType` — задает «tedsType» («тип ЭТДП»), как указано в таблице 17, где он называется «TEDS Access Code» («Код доступа к ЭТДП»);
- `_string ResponseFormat` — задает формат ответа, как это определено в 12.1.2.

12.4.5.2 HTTP-ответ для API «UpdateTEDSCache» («Обновить кэш ЭТДП»)

Ответ на вызов данного API должен содержать следующие параметры:

- `UInt16 errorCode` — информация об ошибке;
- `UInt16 timId` — значение «timId» заданного ИМП;
- `UInt16 channelId` — значение «channelID» заданного канала преобразователя;
- `UInt8 tedsType` — задает «tedsType» («тип ЭТДП»), как указано в таблице 17, где он называется «TEDS Access Code» («Код доступа к ЭТДП»).

12.4.5.3 Схема XML-ответа для API «UpdateTEDS» («Обновить ЭТДП»)

Если формат ответа — «XML», то для ответа должна быть использована следующая схема:

```
<?xml version=»1.0» encoding=»UTF-8»?>
<xs:schema xmlns:xs=»http://www.w3.org/2001/XMLSchema»
xmlns:stml=http://grouper.ieee.org/groups/1451/0/1451HTTPAPI
<xs:complexType name="UpdateTEDSHTTPResponse">
    <xs:sequence>
            <xs:element name="errorCode" type="stml:UInt16"/>
            <xs:element name="timId" type="stml:UInt16Array"/>
            <xs:element name="channelId" type="stml:UInt16"/>
            <xs:element name=”tedsType” type=”stml:UInt8”/>
    </xs:sequence>
</xs:complexType>
</xs:schema>
```

12.5 API управления преобразователем

Данный API содержит четыре API. API «SendCommand» («Отправить команду») и «StartCommand» («Запустить команду») позволяют системе отправлять команды непосредственно каналу преобразователя или ИМП. API «Trigger» («Триггер») и «StartTrigger» («Запустить триггер») используются для отправки триггерных сигналов каналу преобразователя или группе каналов преобразователей, относящимся к одному и тому же СПП.

12.5.1 Интерфейс «SendCommand» («Отправить команду»)

Данный метод осуществляет блокирующую операцию. Формат входных и выходных аргументов зависит от вида команды. Отправитель запроса должен убедиться, что использует правильные типы

данных для каждого входного аргумента. Данный API соответствует `Args::UInt16 sendCommand()`, как описано в 10.3.5.

Путь «Path»: 1451/TransducerManager/SendCommand.

Метод «POST» («Отправить»): отправляет команду каналу преобразователя или ИМП СПП (хоста) и возвращает результат в заданном формате.

12.5.1.1 Входные параметры

Следующие параметры должны поставляться вместе с вызовом данного API:

- `UInt16 timId` — значение «timId» заданного ИМП;
- `UInt16 channelId` — значение «channelID» заданного канала преобразователя;
- `TimeDuration timeout` — данный аргумент определяет время ожидания после записи данных без генерации ошибки тайм-аута в случае неполучения ответа. Значения «secs==0» и «nsecs==–1» указывают на непрерывное (бесконечное) время ожидания. Использование значений «непрерывное (бесконечное) время ожидания» является крайне опасным, поскольку при этом ресурс может быть заблокирован;
- `Uint8 cmdClassId` — задает требуемый код класса команды, как описано в 7.1 и таблице 15;
- `Uint8 cmdFunctionId` — задает требуемый код функции команды. Данные коды «cmdFunctionIds» определены при описании каждой команды в разделе 7 и перечисляются в командной ЭТДП для нестандартных команд;
- `ArgumentArray inArgs` — представляет собой входные аргументы в форме массива аргументов «ArgumentArray»;
- `_string ResponseFormat` — задает формат ответа, как это определено в 12.1.2.

12.5.1.2 HTTP-ответ для API «SendCommand» («Отправить команду»)

Ответ на вызов данного API должен содержать следующие параметры:

- `UInt16 errorCode` — информация об ошибке, как определено в 9.3.1.2;
- `UInt16 timId` — значение «timId» заданного ИМП;
- `UInt16 ChannelID` — значение «channelID» заданного канала преобразователя;
- `ArgumentArray outArgs` — возвращаемые выходные аргументы.

12.5.1.3 Схема XML-ответа для API «SendCommand» («Отправить команду»)

Если формат ответа — «XML», то для ответа должна быть использована следующая схема:

```
<?xml version=»1.0» encoding=»UTF-8»?>
<xs:schema xmlns:xs=»http://www.w3.org/2001/XMLSchema»
xmlns:stml=http://grouper.ieee.org/groups/1451/0/1451HTTPAPI
<xs:complexType name=»SendCommandHTTPResponse»>
    <xs:sequence>
            <xs:element name=»errorCode» type=»stml:UInt16»/>
            <xs:element name=»timId» type=»stml:UInt16Array»/>
            <xs:element name="channelId" type="stml:UInt16"/>
            <xs:element name="outArgs" type="stml:ArgumentArrayType"/>
    </xs:sequence>
</xs:complexType>
</xs:schema>
```

12.5.2 Интерфейс «StartCommand» («Запустить команду»)

Данный метод осуществляет неблокирующую операцию. Формат входных аргументов зависит от вида команды. Отправитель запроса должен убедиться, что использует правильные типы данных для каждого входного аргумента. Данный API соответствует `Args::UInt16 startCommand()`, как описано в 10.3.7. Возвращаемый массив аргументов «ArgumentArray» завершается вызовом API «CompleteCommand» («Завершить команду»).

Путь «Path»: 1451/TransducerManager/StartCommand.

Метод «GET» («Получить»): отправляет команду каналу преобразователя или ИМП СПП (хоста) и возвращает результат в заданном формате.

12.5.2.1 Входные параметры

Следующие параметры должны поставляться вместе с вызовом данного API:

- `UInt16 timId` — значение «timId» заданного ИМП;
- `UInt16 channelId` — значение «channelID» заданного канала преобразователя или значение «0», адресованное ИМП;
- `TimeInstance triggerTime` — задает время начала операции;

- TimeDuration timeout — максимальное время ожидания до генерации ошибки тайм-аута;
- Uint8 cmdClassId — задает требуемый код класса команды, как описано в 7.1 и таблице 15;
- Uint8 cmdFunctionId — задает требуемый код функции команды. Коды «cmdFunctionIds» определены при описании каждой команды в разделе 7 и перечисляются в командной ЭТДП для нестандартных команд;
- ArgumentArray inArgs — входные аргументы в форме массива аргументов «ArgumentArray». Входные аргументы зависят от вида команды;
- _string ResponseFormat — задает формат ответа, как это определено в 12.1.2.

12.5.2.2 HTTP-ответ для API «StartCommand» («Запустить команду»)

Ответ на вызов данного API должен содержать следующие параметры:

- UInt16 errorCode — информация об ошибке, как определено в 9.3.1.2, от неблокирующей операции завершения команды;
- UInt16 timId — значение «timId» заданного ИМП;
- UInt16 ChannelID — значение «channelID» заданного канала преобразователя.

12.5.2.3 Схема XML-ответа для API «StartCommand» («Запустить команду»)

Если формат ответа — «XML», то для ответа должна быть использована следующая схема:

```
<?xml version=»1.0» encoding=»UTF-8»?>
<xs:schema xmlns:xs=»http://www.w3.org/2001/XMLSchema»
xmlns:stml=http://grouper.ieee.org/groups/1451/0/1451HTTPAPI
<xs:complexType name=»StartCommandHTTPResponse»>
    <xs:sequence>
            <xs:element name=»errorCode» type=»stml:UInt16»/>
            <xs:element name=»timId» type=»stml:UInt16Array»/>
            <xs:element name="channelId" type="stml:UInt16"/>
    </xs:sequence>
</xs:complexType>
</xs:schema>
```

12.5.3 Интерфейс «CommandComplete» («Завершить команду»)

Данный метод завершает неблокирующую операцию «StartCommand» («Запустить команду»). Формат входных аргументов зависит от вида команды. Отправитель запроса должен убедиться, что использует правильные типы данных для каждого входного аргумента. Данный API соответствует Args::UInt16 commandComplete(), как описано в 11.6.4.

Путь «Path»: 1451/TransducerManager/CommandComplete.

Метод «GET» («Получить»): получает результат от преобразователя, доступного для заданного ИМП заданного СПП (хоста), и возвращает результат команды «StartCommand» («Запустить команду»), рассмотренной в 12.5.2, в заданном формате.

12.5.3.1 Входные параметры

Следующие параметры должны поставляться вместе с вызовом данного API:

- UInt16 timId — значение «timId» заданного ИМП;
- UInt16 channelId — значение «channelID» заданного канала преобразователя или значение «0», адресованное ИМП;
- _string ResponseFormat — задает формат ответа;

12.5.3.2 HTTP-ответ для API «CommandComplete» («Завершить команду»)

Ответ на вызов данного API должен содержать следующие параметры:

- UInt16 errorCode — информация об ошибке, как определено в 9.3.1.2, от неблокирующей операции завершения команды;
- UInt16 timId — значение «timId» заданного ИМП;
- UInt16 ChannelID — значение «channelID» заданного канала преобразователя;
- ArgumentArray outArgs — возвращаемый массив аргументов «ArgumentArray». Данная информация зависит от конкретной команды.

12.5.3.3 Схема XML-ответа для API «CommandComplete» («Завершить команду»)

Если формат ответа — «XML», то для ответа должна быть использована следующая схема:

```
<?xml version=»1.0» encoding=»UTF-8»?>
<xs:schema xmlns:xs=»http://www.w3.org/2001/XMLSchema»
xmlns:stml=http://grouper.ieee.org/groups/1451/0/1451HTTPAPI
<xs:complexType name=»CommandCompleteHTTPResponse»>
```

```
    <xs:sequence>
        <xs:element name=»errorCode» type=»stml:UInt16»/>
        <xs:element name=»timId» type=»stml:UInt16Array»/>
        <xs:element name="transducerId" type="stml:UInt16"/>
        <xs:element name="outArgs" type="stml:ArgumentArrayType"/>
    </xs:sequence>
</xs:complexType>
</xs:schema>
```

12.5.4 Интерфейс «Trigger» («Триггер»)

Данный метод осуществляет блокирующий триггер на заданном канале преобразователя или группе каналов преобразователей. Данный API соответствует `Args::UInt16 trigger()`, как описано в 10.3.9.

Путь «Path»: 1451/TransducerManager/Trigger.

Метод «POST» («Отправить»): выполняет триггер для канала преобразователя или группы каналов преобразователей, относящихся к определенному СПП (хосту).

12.5.4.1 Входные параметры

Следующие параметры должны поставляться вместе с вызовом данного API:

- `UInt16 timId` — значение «timId» заданного ИМП;
- `UInt16 channelId` — значение «channelID» заданного канала преобразователя или значение «0», если требуется считать ЭТДП всего ИМП;
- `TimeInstance triggerTime` — задает время начала операции;
- `TimeDuration timeout` — задает максимальное время ожидания до генерации ошибки таймаута;
- `UInt8 SamplingMode` — задает режим выборки для канала преобразователя или каналов преобразователя. Подробная информация представлена в 5.11;
- `_string ResponseFormat` — задает формат ответа, как это определено в 12.1.2.

12.5.4.2 HTTP-ответ для API «Trigger» («Триггер»)

Ответ на вызов данного API должен содержать следующие параметры:

- `UInt16 errorCode` — информация об ошибке, как определено в 9.3.1.2;
- `UInt16 timId` — значение «timId» заданного ИМП;
- `UInt16 channelId` — значение «channelID» заданного канала преобразователя или значение «0», если требуется считать ЭТДП всего ИМП.

12.5.4.3 Схема XML-ответа для API «Trigger» («Триггер»)

Если формат ответа — «XML», то для ответа должна быть использована следующая схема:

```
<?xml version=»1.0» encoding=»UTF-8»?>
<xs:schema xmlns:xs=»http://www.w3.org/2001/XMLSchema»
xmlns:stml=http://grouper.ieee.org/groups/1451/0/1451HTTPAPI
<xs:complexType name="TriggerHTTPResponse">
    <xs:sequence>
        <xs:element name="errorCode" type="stml:UInt16"/>
        <xs:element name="timId" type="stml:UInt16Array"/>
        <xs:element name="channelId" type="stml:UInt16"/>
    </xs:sequence>
</xs:complexType>
</xs:schema>
```

12.5.5 Интерфейс «StartTrigger» («Запустить триггер»)

Данный метод выполняет неблокирующий триггер на заданном канале преобразователя или группе каналов преобразователя. Пользователь несет ответственность за определение времени завершения команды путем отправки вызова «SendCommand» («Отправить команду») (см. 12.5.1) с указанием команды «ReadStatusEventRegister» («Считать регистр статуса события») (см. 7.1.1.8) и проверки бита «DataProcessed» («Данные обработаны») (см. 5.13.10), который должен быть установлен. Данный API соответствует `Args::UInt16startTrigger()`, как описано в 10.3.10.

Путь «Path»: 1451/TransducerManager/StartTrigger.

Метод «POST» («Отправить»): запускает триггер преобразователя, доступного для заданного ИМП заданного СПП (хоста), и отображает результат в заданном формате.

12.5.5.1 Входные параметры

Следующие параметры должны поставляться вместе с вызовом данного API:

- `UInt16 timId` — значение «timId» заданного ИМП;
- `UInt16 channelId` — значение «channelID» заданного канала преобразователя или значение «0», если требуется считать ЭТДП всего ИМП;
- `TimeInstance triggerTime` — задает время начала операции;
- `TimeDuration timeout` — задает максимальное время ожидания до генерации ошибки таймаута;
- `UInt16 SamplingMode` — задает режим триггера. Подробная информация представлена в 5.11;
- `_string ResponseFormat` — задает формат ответа, как это определено в 12.1.2.

12.5.5.2 HTTP-ответ для API «StartTrigger» («Запустить триггер»)

Ответ на вызов данного API должен содержать следующие параметры:

- `UInt16 errorCode` — информация об ошибке, как определено в 9.3.1.2;
- `UInt16 timId` — значение «timid» заданного ИМП;
- `UInt16 channelId` — значение «channelID» заданного канала преобразователя или значение «0», если требуется считать ЭТДП всего ИМП.

12.5.5.3 Схема XML-ответа для API «StartTrigger» («Запустить триггер»)

Если формат ответа — «XML», то для ответа должна быть использована следующая схема:

```
<?xml version=»1.0» encoding=»UTF-8»?>
<xs:schema xmlns:xs=»http://www.w3.org/2001/XMLSchema»
xmlns:stml=http://grouper.ieee.org/groups/1451/0/1451HTTPAPI
<xs:complexType name="StartTriggerHTTPResponse">
    <xs:sequence>
          <xs:element name="errorCode" type="stml:UInt16"/>
          <xs:element name="timId" type="stml:UInt16Array"/>
          <xs:element name="channelId" type="stml:UInt16"/>
    </xs:sequence>
</xs:complexType>
</xs:schema>
```

Приложение А
(справочное)

Библиография

[B1] HPL-96-61, 1995[1)] Hamilton B. «A compact representation of physical units», Hewlett-Packard Company, Palo Alto, CA, Hewlett-Packard Laboratories Technical Report (Гамильтон Б. «Компактное представление физической единицы». Компания «Хьюлетт паккард», Пало-Альто, Калифорния, технический отчет лаборатории «Хьюлетт паккард»)

[B2] IEEE 100 The Authoritative Dictionary of IEEE Standards Terms, Seventh Edition, New York, Institute of Electrical and Electronics Engineers, Inc. (Авторитетный словарь терминов стандартов ИИЭР. Седьмое издание, Нью-Йорк, Институт инженеров по электротехнике и радиоэлектронике)

[B3] IEEE P1451.6 Draft Standard for a Smart Transducer Interface for Sensors and Actuators — A High-Speed CAN open-Based Transducer Network Interface for Intrinsically Safe and Non-Intrinsically Safe Applications[2)] [Стандарт для интрефейса интеллектуального преобразователя для датчиков и исполнительных устройств. Высокоскоростной сетевой интрефейс преобразователя, основанный на открытых стандартах CAN для конструктивно безопасных и небезопасных приложений (проект)]

[B4] IEEE Std 1451.5—2007 IEEE Standard for a Smart Transducer Interface for Sensors and Actuators — Wireless Communication Protocols and Transducer Electronic Data Sheet (TEDS) Formats[3), 4)] (Стандарт ИИЭР для интерфейса интеллектуального преобразователя для датчиков и исполнительных устройств. Протоколы радиосвязи и форматы ЭТДП)

[B5] ISO 8601 Data Elements and Interchange Formats — Information Interchange — Representation of Dates and Times[5)] (Элементы данных и форматы обменов. Информационный обмен. Представление даты и времени)

[B6] ISO/IEC 7498-1 Information Technology — Open Systems Interconnection — Basic Reference Model — Part 1: The Basic Model[6)] (Информационные технологии. Взаимосвязь открытых систем. Базовая эталонная модель. Часть 1. Базовая модель)

[B7] OGC document 05-010, 2005 OGC™ Recommendation Paper, Version: 1.0, «URNs of definitions in ogc namespace» [Рекоммендации OGC, версия 1.0, «Определение унифицированных имен ресурсов (УИР) в именах Открытого консорциума геопространственных данных (ОКГД)»]

---

1) Документ доступен в отделе технических публикаций компании «Aligent», 1501 Page Mill Road, Mail Stop 2L, Palo Alto, CA 94304, США.

2) Данный проект стандарта ИИЭР не был одобрен Советом по стандартам ИИЭР (IEEE-SA) на момент публикации настоящего стандарта. Для получения проекта стандарта необходимо связаться с ИИЭР.

3) Публикации ИИЭР доступны в Институте инженеров по электротехнике и радиоэлектронике, Inc., 445 Hoes Lane, Piscataway, NJ08854, США (http://standards.ieee.org/).

4) Стандарты ИИЭР или продукты, упомянутые в настоящем пункте, являются товарными знаками Института инженеров по электротехнике и радиоэлектронике, Inc.

5) Публикации ИСО доступны в Центральном секретариате ИСО, Case Postale 56, 1 rue de Varembé, CH-1211, Genève 20, Швейцария (http://www.iso.ch/). Публикации ИСО также доступны в Соединенных Штатах в отделе продаж Американского национального института стандартов, 25 West 43rd Street, 4th Floor, New York, NY 10036, США (http://www.ansi.org/).

6) Публикации ИСО/МЭК доступны в Центральном секретариате ИСО, Case Postale 56, 1 rue de Varembé, CH-1211, Genève 20, Швейцария (http://www.iso.ch/). Публикации ИСО/МЭК также доступны в Соединенных Штатах в компании Global Engineering Documents, InvernessWayEast, Englewood, CO 80112, США (http://global.ihs.com/). Электронные копии доступны в Соединенных Штатах в Американском национальном институте стандартов, 25 West 43rd Street, 4th Floor, New York, NY 10036, США (http:// www.ansi.org/).

[B8] RFC 2045 — Multipurpose Internet Mail Extensions (MIME) — Part One: Format of Internet Message Bodies[1] (Многоцелевые расширения почтового стандарта. Часть 1. Формат тела сообщений)

[B9] — Taylor B. N., Ed., The International System of Units (SI), National Institute of Standards and Technology, Special Publication 330. Washington, D.C.: U.S. Government Printing Office, August 1991 (Тейлор Б. Н. «Международная система единиц (СИ)». Национальный институт стандартов и технологий США, специальное издание 330. Вашингтон, государственная типография США, август 1991)

[B10] — Taylor B.N. and Kuyatt C.E., «Guidelines for evaluating and expressing the uncertainty of NIST measurement results», NIST Technical Note 1297, National Institute of Standards Technology, Gaithersburg, MD, 1994 edition (Тейлор Б.Н. и Куятт К. Е. Техническое указание 1297 «Руководство по оценке и представлению неопределенности результатов измерений NIST». Национальный институт стандартов и технологий США, Гейтерсберг, издание 1994 г.)

[1] Публикации, предлагаемые к обсуждению, доступны в компании Global Engineering Documents, 15 Inverness Way East, Englewood, CO 80112, США (http://global.ihs.com/).

Приложение В
(справочное)

Руководящие указания для интерфейса сервисов преобразователя

Приложения, предназначенные для измерения и контроля, взаимодействуют с уровнем ИИЭР 1451.0 при помощи интерфейса сервисов преобразователя. В настоящем приложении приведены примеры использования данного интерфейса.

Простейшее и наиболее распространенное приложение измерения осуществляет считывание значения канала преобразователя. Если предположить, что канал преобразователя эксплуатируется в режиме выборки данных «Immediate operation» («Немедленное выполнение»), то канал преобразователя автоматически выполнит измерение и возвратит результат. Таким образом, данное измерение является «опрошенным».

На диаграмме последовательности действий (рисунок В.1) показан поток. Приложение находится слева, а временная шкала направлена вниз. Детали обработки уровня ИИЭР 1451.0 подробнее публичного API не представлены.

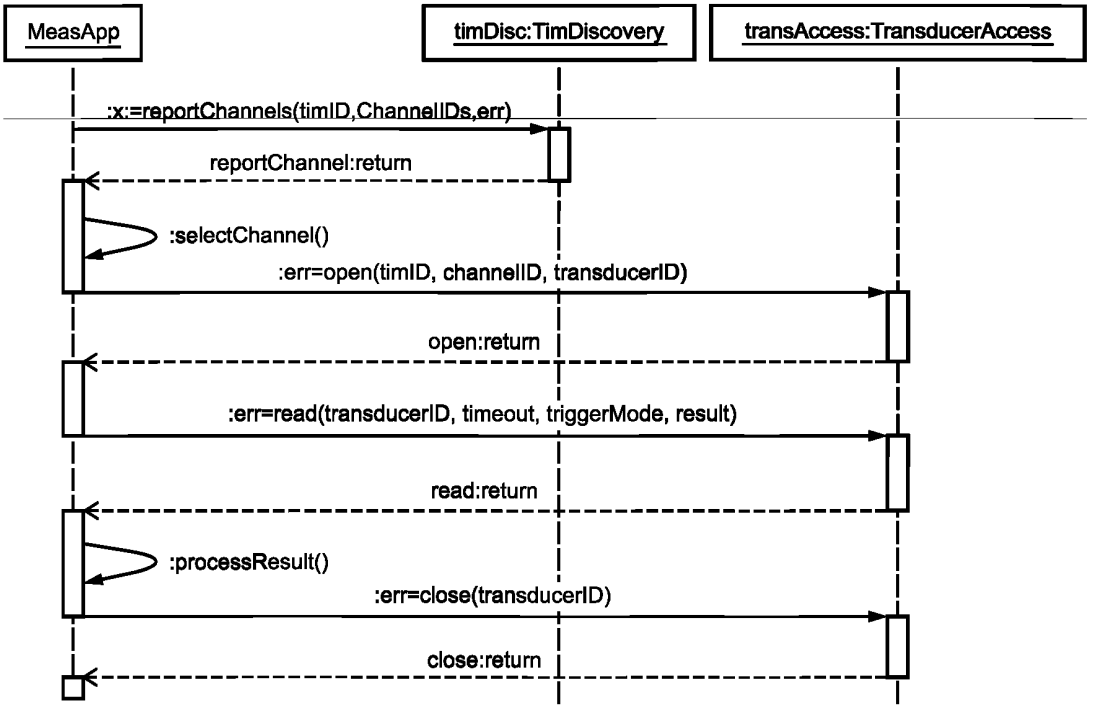


Рисунок В.1 — Простой пример «опрошенного» измерения

Приложение (объект, обозначенный как «MeasApp») получает доступ к методам на объекте «TIMDiscovery» («Обнаружение ИМП») для обнаружения доступных ИМП и каналов преобразователя.

После выбора канала осуществляется вызов метода «open( )» («Открыть») на объекте «TransducerAccess» доступа к преобразователю. В результате данной операции возвращается действительный идентификатор «transducerID» преобразователя, который будет использоваться при последующих вызовах.

Когда приложение готово выполнить измерение, оно осуществляет вызов метода «read()» («Считать»). Новое значение измерения будет возвращено в качестве выходного параметра «result» («Результат»). Данный параметр представляет собой объект «ArgumentArray» («Массив аргументов»), и приложение будет использовать метод «get ()» («Получить») для извлечения интересующих атрибутов измерения. Например, атрибут «value» («Значение») представляет собой значение измерения, а атрибут «timestamp» («Временная метка») содержит время проведения измерения.

Приложение может контролировать атрибуты, которые возвращаются в массиве аргументов, с помощью вызова метода «configureAttributes()» («Конфигурировать атрибуты») на объекте «TransducerManager» («Управление преобразователем») (на рисунке не показан). Например, могут быть включены атрибуты «units» («Единицы измерения»), «accuracy» («Точность»), «name» («Имя») и «ID» («Идентификатор»). Это приводит к тому, что при вызове метода «read()» («Считать») возвращается массив аргументов с данными атрибутами для более полного «самоописания» данных.

Приложение может осуществлять вызов метода «read()» («Считать») так часто, как это требуется для выполнения множественных считываний. Когда приложение выполнено, осуществляется вызов метода «close()» («Закрыть») для освобождения ресурсов.

Приведенный выше пример описывает простейший случай, когда выполнены следующие условия:

- осуществляется доступ к одному каналу преобразователя;
- приложение блокируется до момента возвращения результата. Максимальное количество времени, в течение которого приложение готово ждать, задается параметром «timeout» («Тайм-аут» или «Временная задержка»);
- приложение задает требуемый режим выборки данных. В этом примере используется режим выборки «Immediateoperation» («Немедленное выполнение»), который запускает выполнение измерения каналом преобразователя.

Существуют также методы для выполнения более сложных последовательностей измерений.

B.1 Пример неблокирующего считывания

Во многих приложениях для измерения и контроля измерения запрашиваются или планируются при помощи одного вызова. Позже, когда процесс измерения завершен, результаты доставляются обратно в приложение с помощью обратного вызова. Пример такого неблокирующего считывания показан на рисунке B.2.

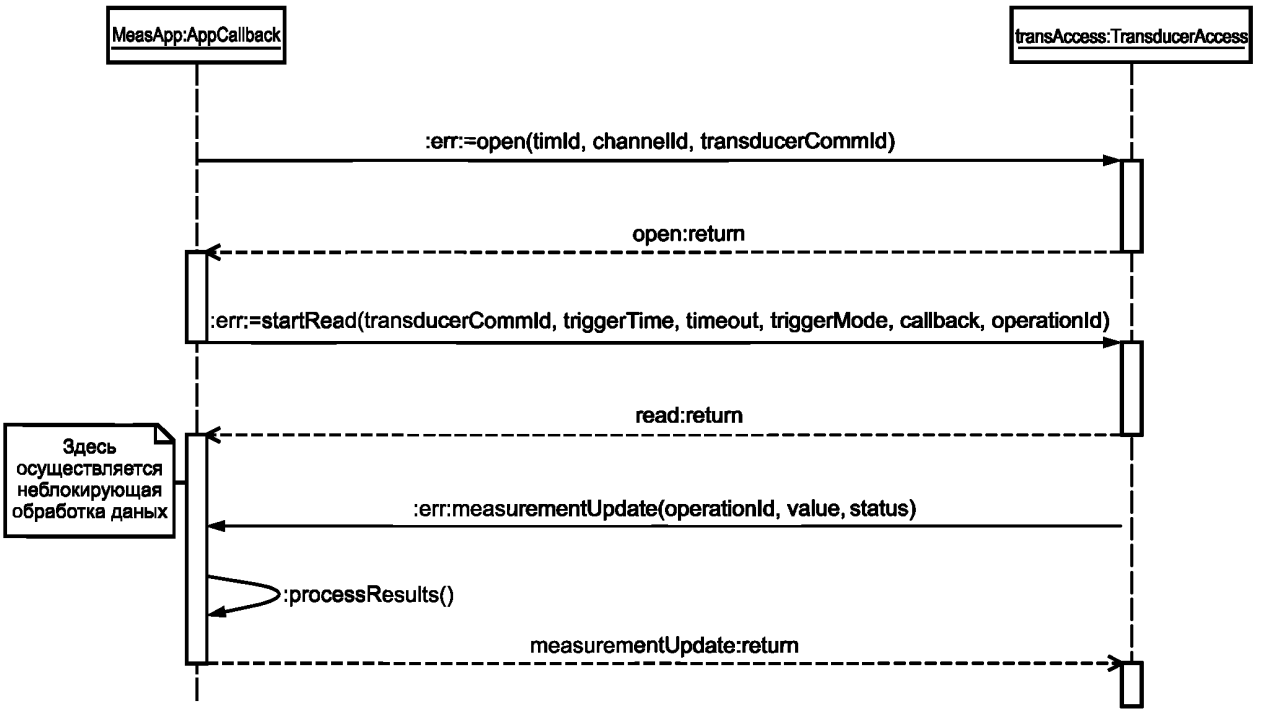


Рисунок B.2 — Пример неблокирующего считывания

При неблокирующем считывании приложение применяет интерфейс «AppCallback». Во время вызова «startRead()» («Начать считывание») данный интерфейс передается в качестве параметра «callback» («Обратный вызов»). Также задаются требуемые время запуска триггера и время ожидания (тайм-аут).

Когда измерение завершено, применяется метод обратного вызова «measurementUpdate()» («Обновить измерения») для возврата результатов измерения приложению.

B.2 Общий механизм «SendCommand()» («Отправить команду»)

Как показано в разделе 7, многие команды определяются на уровне ИИЭР 1451.0. Во многих случаях через API предоставляются конкретные «удобные» методы, чтобы облегчить приложению выполнение полезной работы, например, считывание канала преобразователя или ЭТДП. После применения «удобных» методов используется механизм «SendCommand()». Чтобы получить доступ к функциям, которые не предоставляются через «удобные» методы, метод «SendCommand()» используют следующим образом:

- осуществляется вызов метода «open()» для получения параметра «transducerId» требуемого ИМП и канала преобразователя;
- «commandId» выбирают из всех возможных команд, нумерация которых представлена в разделе 7. Данное значение формата UInt16 содержит «класс» команды в старших 8 битах и «функцию» команды в младших 8 битах;

- все конкретные входные аргументы команды содержатся в массиве аргументов «inArgs». Положение и типы аргументов в массиве аргументов должны соответствовать тому, что запрашивается для команды в вопросе;
- осуществляется вызов метода «SendCommand()»;
- выходные параметры возвращаются к отправителю запроса посредством массива аргументов «outArgs». Положение и типы аргументов в данном массиве аргументов будут соответствовать тому, что возвращается командой в вопросе.

На рисунке В.3 показан процесс использования механизма «SendCommand()».

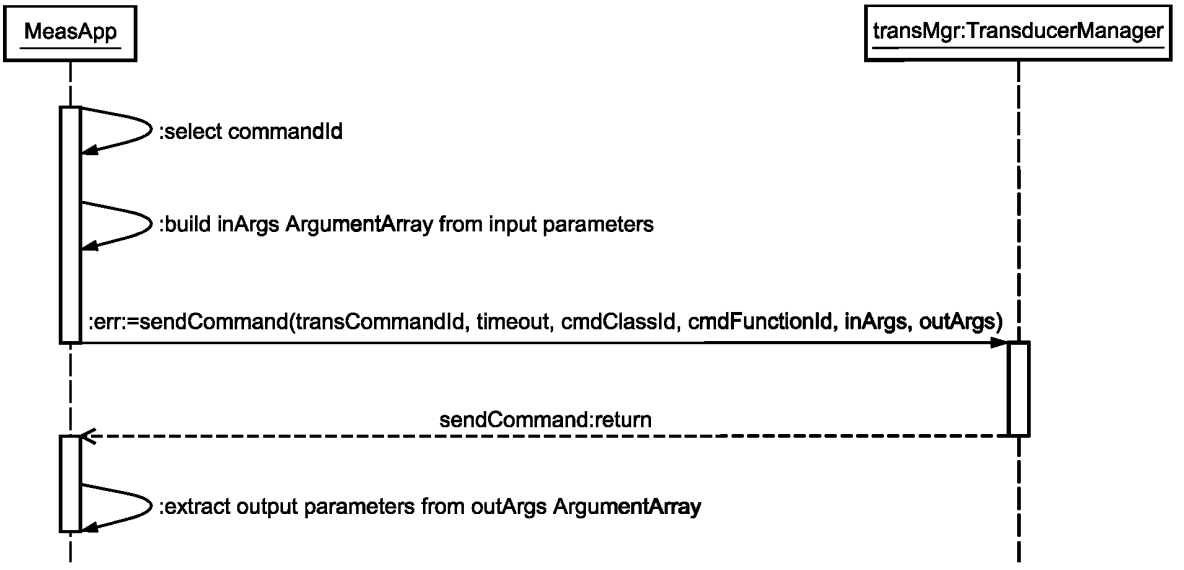


Рисунок В.3 — Диаграмма последовательности действий «SendCommand»

Также доступна неблокирующая версия данной команды (см. обратные вызовы «startCommand( )» и «commandComplete( )»).

В.3 Детали обработки данных на уровне ИИЭР 1451.0

Для описания типа обработки данных, которая осуществляется вслед за вызовом метода «read( )» при осуществлении измерения, на рисунке В.4 приведена диаграмма последовательности действий. Поскольку внутренние детали уровня ИИЭР 1451.0 не рассматриваются в настоящем стандарте, данная диаграмма является вспомогательным средством для демонстрации одного из способов структурирования обработки данных на уровне ИИЭР 1451.0.

Приложение представлено слева, а уровни ИИЭР 1451.X находятся в центре. Вызов «open( )» и обработка со стороны ИМП над уровнем ИИЭР 1451.0 на данной диаграмме не показаны. Данная диаграмма предполагает, что приложение выполняет немедленное считывание канала преобразователя 1 из требуемого ИМП. После того как приложение вызывает метод «read( )», происходит следующая обработка данных:

a) формируется соответствующая команда для ИМП о считывании требуемого канала преобразователя с заданным «samplingmode» («Режим выборки данных») (см. 7.1.2.4) [режим «immediate mode» («Немедленное выполнение») в данном случае]. При этом все входные аргументы помещаются в массив аргументов «inArg»;

b) метод «Codec's encode( )» («Кодирование кодеком») используется для построения байтового массива «payload» («Полезная нагрузка») для данной команды. Так как в данном случае рассматривается команда считывания, то «payload» содержит следующие байты («payload» на выходе содержит 10 байтов):

1) идентификатор канала преобразователя 0x0001 в виде Uint16, который представляет собой номер канала преобразователя для считывания;

2) класс команды 0x02 в виде Uint8, который показывает рабочий класс команды;

3) код функции команды в виде Uint8. В данном случае код представляет собой код команды «Считать сегмент набора данных канала преобразователя» (значение 1);

4) длина переменной части 0x0004 в виде Uint16;

5) аргумент «DataSetOffset», который является единственным аргументом команды и содержит 0x00000000 в виде UInt32, так как набор данных содержит одно значение;

c) команда «write( )» («Записать») вызывается на уровне ИИЭР 1451.X для инициации передачи байтового массива данных в ИМП. Теперь данная цепочка будет заблокирована, так как происходит ожидание, пока ИМП осуществит измерение. Максимальное значение времени ожидания задается приложением;

d) когда обработка со стороны ИМП завершена, уровень ИИЭР 1451.X передает результат в виде байтового массива обратно в СПП. Это приводит к вызову метода «notify( )» («Уведомить»);

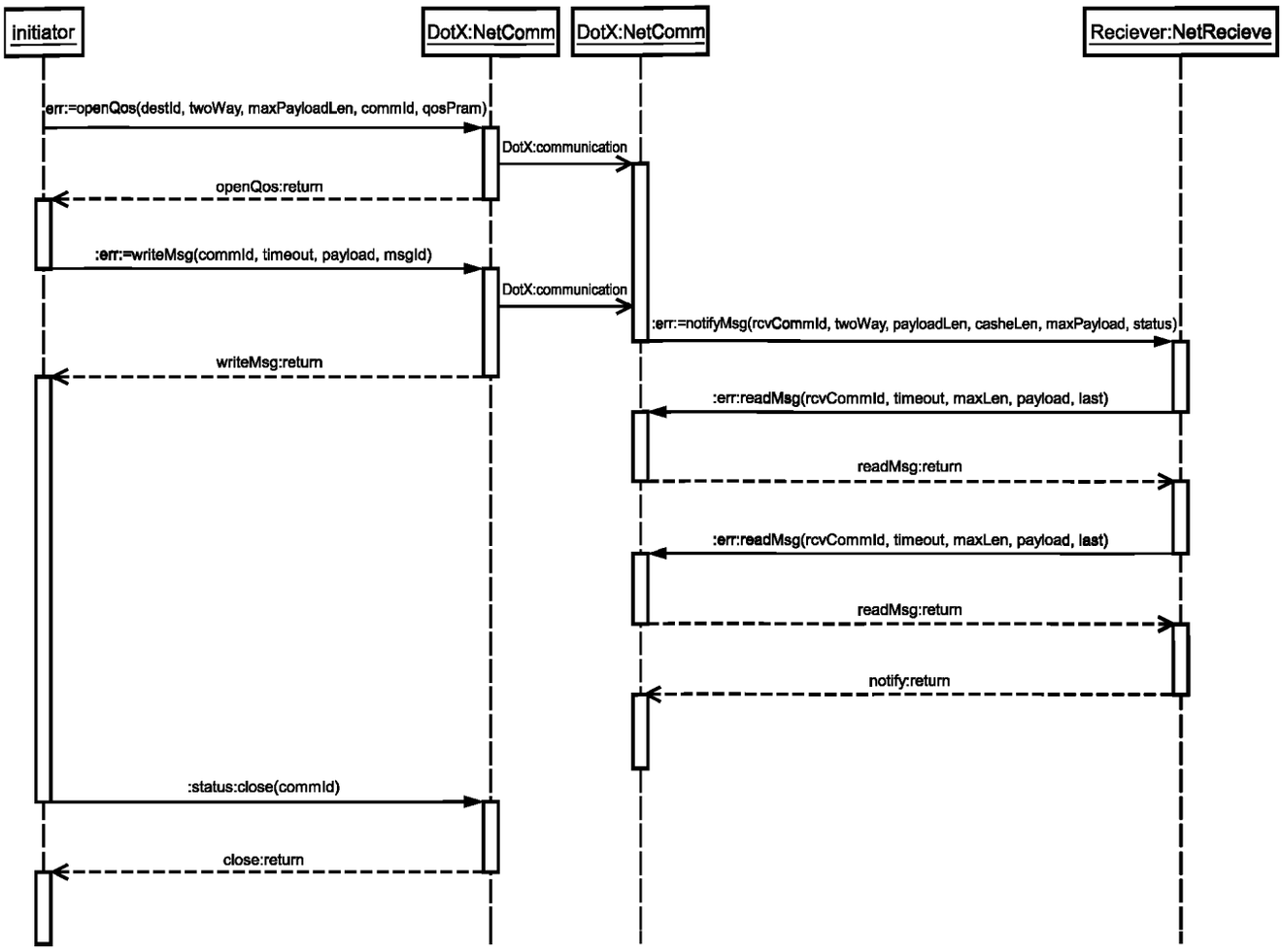


Рисунок В.4 — Диаграмма последовательности действий для детального описания операции считывания

e) обработка «notify( )» переводит в рабочее состояние заблокированную цепочку приложения;

f) вызывается метод «read( )» для получения байтового массива ответного сообщения;

g) метод «Codec's encode( )» используется для обратного преобразования байтового массива в массив аргументов;

h) метод «CorrectionEngine's convert( )» используется для вычисления скорректированных значений (то есть преобразования единиц измерения со стороны ИМП в единицы измерения со стороны СПП);

i) массив аргументов дополнительно форматируется для возврата требуемых атрибутов измерений. Данный процесс может включать в себя извлечение отдельной информации из кэшированных ЭТДП (например, единиц измерения);

j) массив аргументов возвращается в виде выходного параметра обратно в приложение.

Приложение С
(справочное)

Руководящие указания для интерфейса модульных связей

С.1 API модульных связей

Логическая связь между СПП и модулями ИМП или между различными ИМП задается посредством API модульных связей. Существуют четыре аспекта данного API:

а) обмен сообщениями может быть как «односторонним», от отправителя запроса к получателю, так и «двусторонним», когда отправитель посылает команду получателю, а получатель посылает ответ;

б) обмен сообщениями может осуществляться либо в режиме «один-на-один», когда участвуют только два устройства, либо в режиме «один-многим», когда отправитель запроса взаимодействует с группой устройств. Режим «один-многим» всегда является «односторонним»;

в) параметр «Quality of Service» («Качество сервиса») связи может принимать значение «по умолчанию» или «специальные» значения. Значение качества сервиса «по умолчанию» предполагает обеспечение оптимальной доставки данных для данных условий без использования особых механизмов связи;

г) физический интерфейс может представлять собой простой канал связи «точка-точка» («узел-узел») или являться «сетью», когда несколько устройств совместно используют средства связи.

API модульных связей подразделяется на три основных интерфейса: «Comm» («Общий»), «Registration» («Регистрация») и «Receive» («Получение»). Интерфейс «Comm» реализуется на уровне ИИЭР1451.X и вызывается уровнем ИИЭР 1451.0 для осуществления контроля связи между СПП и ИМП. Интерфейс «Registration» обеспечивается уровнем ИИЭР 1451.0 и вызывается уровнем ИИЭР 1451.X для саморегистрации и регистрации известных адресатов. Интерфейс «Receive» обеспечивается уровнем ИИЭР 1451.0 и вызывается уровнем ИИЭР 1451.X, когда входящие сообщения связи получены уровнем ИИЭР 1451.X в виде ссылки.

Каждый из трех данных интерфейсов, в свою очередь, подразделяется на два субинтерфейса: «point-to-point» («точка-точка» или «узел-узел»), когда СПП и ИМП непосредственно связываются друг с другом, и «network» («сеть»), когда вместе связываются СПП и несколько ИМП. Для обозначения данных двух случаев используются соответственно приставки «P2P» и «Net». На рисунке С.1 данные случаи показаны для интерфейсов типа «Comm». На рисунке С.2 данные случаи показаны для интерфейсов типа «Registration». На рисунке С.3 данные случаи показаны для интерфейсов типа «Receive».

Название «point-to-point» («точка-точка» или «узел-узел») дано с точки зрения связи. Это не означает, что физический канал связи реализован как «точка-точка».

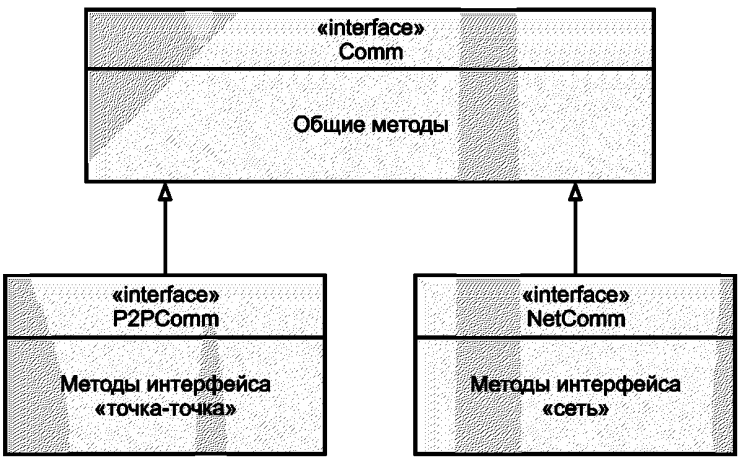


Рисунок С.1 — Методы связи

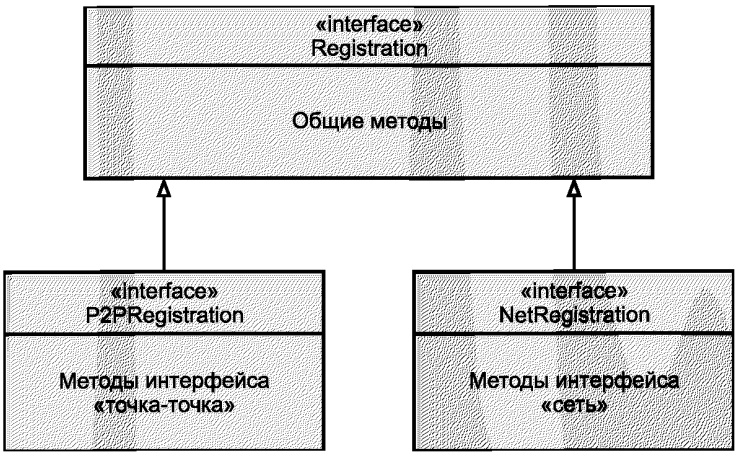


Рисунок С.2 — Интерфейсы «Registration» («Регистрация»)

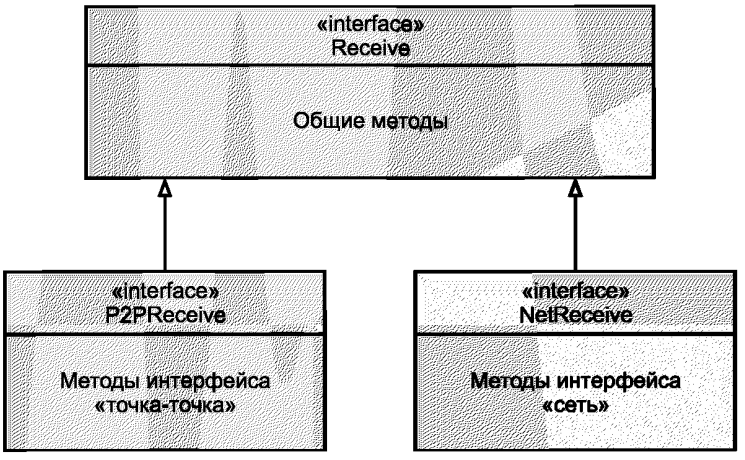


Рисунок С.3 — Интерфейсы «Receive» («Получение»)

С.2 Симметричные API

Данные API являются симметричными и поддерживают связи «СПП-ИМП» и «ИМП-СПП» и опционально «ИМП-ИМП». Каждая операция связи начинается с инициирующего узла, передающего полезную нагрузку в один или более принимающих узлов. Принимающий узел может опционально возвращать ответ инициирующему узлу.

Поддерживаются следующие модели связи:

- «двусторонняя»/«один-на-один». Например, СПП выдает команду «Read TEDS» («Считать ЭТДП») заданному ИМП. ИМП посылает ответ с содержанием ЭТДП;
- «односторонняя»/«один-на-один». Например, ИМП генерирует периодические измерения, которые отправляются обратно в СПП;
- «односторонняя»/«один-многим». Например, СПП выдает команду «group trigger» («Триггер группы»), которая передается нескольким ИМП, участвующим в измерении.

С.3 Выборы реализации

СПП и ИМП могут выбирать различные интерфейсы реализации и при этом быть в состоянии связываться друг с другом. Ключевым является вопрос о том, необходимо ли устройству инициировать связь со многими устройствами или же он связывается с одним адресатом. В первом случае необходимо реализовать интерфейс «NetComm». Во втором случае необходимо реализовать интерфейс «P2PComm». Устройство, которое только получает команду и выдает ответное сообщение (то есть никогда не инициирует команду), должно также реализовывать интерфейс «P2PComm».

Второй вопрос, который необходимо рассмотреть, заключается в том, какое число различных линий связи ИИЭР 1451.X поддерживается на устройстве и носят ли данные связи статический или динамический характер. В большинстве случаев доступен только один канал связи ИИЭР 451.X. Следовательно, для подключения реализации ИИЭР 1451.X к реализации ИИЭР 1451.0 может использоваться процесс сборки программного обеспече-

ния (например, редактор связей), что устраняет необходимость использования некоторых методов интерфейса «Registration» («Регистрация»).

Несколько более сложное устройство может обеспечить фиксированный набор каналов связи ИИЭР 1451.X. Аналогично, так как данные линии связи известны при сборке, процесс сборки программного обеспечения (например, редактор связей) может использоваться для управления регистрацией. При этом за использование корректного устройства ИИЭР 1451.X при попытке связи через данный канал отвечает уровень ИИЭР 1451.0.

Чтобы иметь возможность динамически управлять доступными каналами связи ИИЭР 1451.X, может потребоваться устройство высокого класса. В данном случае обеспечивается динамический механизм регистрации, позволяющий уровню ИИЭР 1451.X уведомить уровень ИИЭР 1451.0 о том, что он доступен. Данный механизм подходит как для интерфейса «P2PComm», так и для интерфейса «NetComm».

Устройство, которое подключено к многоточечной сети ИИЭР 1451.X и которому необходимо инициировать связь с более чем одним адресатом, реализовывает интерфейс «NetComm». Данный интерфейс обеспечивает дополнительные методы и параметры для обработки одновременных перекрывающихся операций связи с несколькими адресатами.

С.4 Примеры реализаций

Простейший случай представляет собой основной ИМП, отвечающий исключительно на запросы СПП. Данный процесс может осуществляться посредством канала «point-to-point» («точка-точка» или «узел-узел») (см., например, ИИЭР 1451.2-RS232) или посредством «сетевого» физического канала (см., например, ИИЭР 1451.5—2007 (раздел 7) [B4]). Тем не менее с точки зрения данного ИМП он исключительно отвечает на запросы и никогда не инициирует операции связи. Поскольку требования к связи являются простыми, реализация ИМП ИИЭР 1451.X обеспечивает интерфейс «P2PComm». Так как имеется только одна реализация «Comm» ИИЭР 1451.X, то механизм регистрации будет осуществляться через статическое соединение, устанавливаемое в процессе сборки программного обеспечения.

Следующий простейший случай представляет собой простой СПП с единственным физическим соединением типа «point-to-point» («точка-точка» или «узел-узел») с одним ИМП (например, ИИЭР 1451.2-RS232). Реализация СПП ИИЭР 1451.X будет также обеспечивать интерфейс «P2PComm». Соединение между уровнем ИИЭР 1451.X и уровнем ИИЭР 1451.0 устанавливается в процессе сборки программного обеспечения.

Следующий по сложности случай представляет собой СПП с несколькими физическими соединениями «point-to-point» («точка-точка» или «узел-узел») (например, несколько портов ИИЭР 1451.2-RS232). При этом необходимо наличие нескольких устройств ИИЭР 1451.X, по одному для каждого физического соединения. Для каждого такого соединения будет обеспечиваться интерфейс «P2PComm». Использование соответствующего интерфейса ИИЭР 1451.X для связи с каждым ИМП управляется реализацией ИИЭР 1451.0 на СПП.

Самый сложный случай представляет собой СПП, которому требуется поддерживать нескольких физических интерфейсов различных типов и в случае динамического характера конфигурации. Каждому физическому интерфейсу требуется либо интерфейс «P2PComm», либо интерфейс «NetComm». Данные устройства должны также динамически регистрировать себя, чтобы сообщить уровню ИИЭР 1451.0 о том, что они доступны.

С.5 Параметры связи узлов

В конфигурациях «NetComm» каждый узел в сети однозначно идентифицируется с помощью параметров связи узла 1451.X. Например, в случае с ИИЭР 1451.5-802.11 каждый узел имеет уникальный IP-адрес, а для связей уровня ИИЭР 1451.0 выделяются сетевые порты. Каждая технология ИИЭР 1451.X может иметь различный набор обязательных параметров.

Несмотря на то что природа данных параметров зависит от конкретной технологии уровня ИИЭР 1451.X, уровень ИИЭР 1451.0 должен иметь возможность запрашивать и передавать их по системе в общей форме. Вызов метода «getNodeParams()» («Получить параметры узла») используется для извлечения данных параметров в виде массива аргументов. Данный вызов возвращает параметры с локального уровня ИИЭР 1451.X.

С.6 Параметр «destId» для идентификации адресата

На уровне ИИЭР 1451.0 единственный СПП и связанные с ним ИМП образуют логическую группу связи. Уровень ИИЭР 1451.0 со стороны СПП присваивает уникальный идентификатор Uint16 каждому узлу (СПП или ИМП). Значение 0x0000 зарезервировано как широковещательный адрес, а значение 0x0001 зарезервировано как адрес управляющего СПП. В оставшейся части настоящего стандарта данный «идентификатор адресата ИИЭР 1451.0» будет сокращенно обозначаться как «destId».

Со стороны СПП уровень ИИЭР 1451.X отвечает за обнаружение всех ИМП в логической группе связи и их регистрацию на уровне ИИЭР 1451.0 со стороны СПП путем вызова метода «registerDest()» («Зарегистрировать адресата»). Уровень ИИЭР 1451.0 назначает уникальный «destId», а уровень ИИЭР 1451.X должен кэшировать информацию соответствующего сетевого «узла» и сопоставить ее с данным «destId». Например, в случае реализации ИИЭР 1451.5-802.11 потребуется сопоставить «destId» с IP-адресом удаленного узла и номером порта.

Со стороны ИМП уровень ИИЭР 1451.X должен знать, как связываться с управляющим СПП с помощью значения 0x0001 для «destId». При этом регистрация уровня ИИЭР 1451.0 со стороны ИМП не требуется.

В качестве дополнительной функции в случаях, когда ИМП инициирует связь с различными адресатами (например, сгенерированная ИМП триггерная команда для другого ИМП или группы), для обработки необходимой конфигурации обеспечивается команда ИИЭР 1451.0 со стороны ИМП. Массив аргументов, извлеченный с помощью метода «getNodeParams()» на адресуемом ИМП (или группе ИМП), будет передан уровню ИИЭР 1451.0 инициирующего ИМП. Данная информация будет спущена на уровень ИИЭР 1451.X инициирующего ИМП с помощью вызова метода «addDestination()», где она может кэшировать необходимую частную сетевую информацию адресата. В данном случае уровень ИИЭР 1451.X на инициирующем ИМП не вызывает метод «registerDest()» уровня ИИЭР 1451.0.

«DestId» 0x0000 зарезервирован как широковещательный адрес для логической группы связи. Он может быть использован только для отправки односторонних сообщений всем узлам в пределах группы.

С.7 Параметр «commId» сеанса связи

При инициировании операции связи уровень ИИЭР 1451.0 осуществляет вызовы «open( )» или «openQoS( )» и задает «destId» и уникальный «commId». Если данные вызовы осуществлены успешно, то ИИЭР 1451.0 будет обращаться к данной сессии при помощи параметра «commId». При закрытии уровня ИИЭР 1451.0 он осуществляет вызов «close( )», чтобы уведомить уровень ИИЭР 1451.X о завершении сеанса и о том, что ресурсы ИИЭР 1451.X могут быть безопасно восстановлены.

В качестве дополнительной опции уровень ИИЭР 1451.X может поддерживать несколько вызовов «open( )» для одного и того же или разных адресатов. В случае если ИИЭР 1451.X достиг пределов сети или ресурсов памяти, он должен генерировать соответствующие коды сбоя. Кроме того, уровню ИИЭР 1451.0 следует попытаться осуществить вызов «close( )», чтобы освободить ресурсы уровня ИИЭР 1451.X до осуществления последующих вызовов «open( )».

ИИЭР 1451.X, поддерживающий несколько вызовов «open( )», позволяет осуществлять перекрывающиеся связи, которые, как правило, приводят к повышению эффективности. Данная функция является дополнительной.

В случаях, когда узел выступает в роли получателя, ИИЭР 1451.X вызывает метод «notifyMsg()» на уровне ИИЭР 1451.0 и обеспечивает уникальный «commId». В данном случае на стороне получателя метод «open( )» не вызывается. Уровень ИИЭР 1451.0 использует данный параметр «commId» для считывания входящих сетевых данных с помощью вызова «readMsg()». В качестве дополнительной опции уровень ИИЭР 1451.0 будет использовать «commId» для отправки ответа обратно инициирующему узлу. Уровень ИИЭР 1451.0 не будет осуществлять вызов «close( )» для данного «commId», так как он обрабатывается внутри уровнем ИИЭР 1451.X.

С.8 Параметр «MsgID» идентификатора транзакции сообщения

При инициировании связи после успешного осуществления вызовов «open( )» или «openQoS( )» уровень ИИЭР 1451.0 начнет связь путем вызова метода «writeMsg()». Уровень ИИЭР 1451.0 обеспечивает соответствующий параметр «commId» для информирования уровня ИИЭР 1451.X о том, какой сеанс связи необходимо использовать. Уровень ИИЭР 1451.0 также задает уникальный параметр «msgID» для информирования уровня ИИЭР 1451.X о том, как соотнести ответ и исходящее сообщение. Уровень ИИЭР 1451.X отвечает за кэширование обоих идентификаторов для использования в обратном вызове «notifyRsp()» к уровню ИИЭР 1451.0, когда ответ получен.

В качестве дополнительной функции уровень ИИЭР 1451.X может поддерживать несколько вызовов «writeMsg()» для одного и того же «commId», но с разными «msgID». Это позволяет осуществлять перекрывающиеся связи в пределах одного сеанса связи и может привести к значительному повышению производительности. При достижении пределов памяти или сети уровень ИИЭР 1451.X должен давать сбой при осуществлении последующих вызовов «writeMsg()». Пока не получены недостающие ответы, уровень ИИЭР 1451.0 прекращает работу до последующих вызовов «writeMsg()».

С.9 Реализации с ограничением памяти

Проектирование данных API нацелено на работу в устройствах, которые имеют серьезные ограничения оперативной памяти, как, например, 8-битный программируемый контроллер интерфейса микропроцессора. Когда уровень ИИЭР 1451.0 вызывает методы считывания или записи байтового массива из уровня ИИЭР 1451.X или в уровень ИИЭР 1451.X (например, «readMsg()», «readRsp()», «writeMsg()» и «writeRsp()»), уровень ИИЭР 1451.0 всегда задает максимальное число байтов, которые он предоставляет или может принять. Для передачи байтовых массивов больших размеров могут потребоваться повторные вызовы данных методов. Чтобы сигнализировать о полной передаче байтового массива, используется сигнальный параметр «last» («последний»).

В случаях использования устройств с ограниченной памятью крайне важно, чтобы уровень ИИЭР 1451.X обеспечивал механизм регулирования передачи потока байтов по сети. Типичным примером является ИМП с ограниченными ресурсами памяти, который выполняет операцию записи ЭТДП. Если размер блока ЭТДП превышает размер доступной памяти ИМП, то уровень ИИЭР 1451.0 со стороны ИМП будет вынужден считывать байтовый массив по частям с использованием вызова метода «readRsp()». С каждой частью он будет записывать данные в соответствующее постоянное хранилище (например, флэш-память). Может потребоваться очень много времени в случае, если флэш-память должна быть стерта. В любом случае уровень ИИЭР 1451.X отвечает за ожидание, пока

уровень ИИЭР 1451.0 осуществляет последующий вызов «readMsg( )» без переполнения каких-либо локальных сетевых буферов. В большинстве случаев потребуются сообщения для контроля потока между СПП и ИМП уровня ИИЭР 1451.X.

С.10 Механизмы состояний связи ИИЭР 1451.X

На рисунках С.4 и С.5 показаны переходы состояний для инициирующего и принимающего узлов. В случае «NetComm», когда поддерживаются одновременные транзакции, применение каждой пары <«commId», «MsgID»> приводит к новому вызову данных механизмов состояний. На рисунке С.4 приведена диаграмма состояний для инициирующего узла, а на рисунке С.5 — диаграмма состояний для принимающего узла.

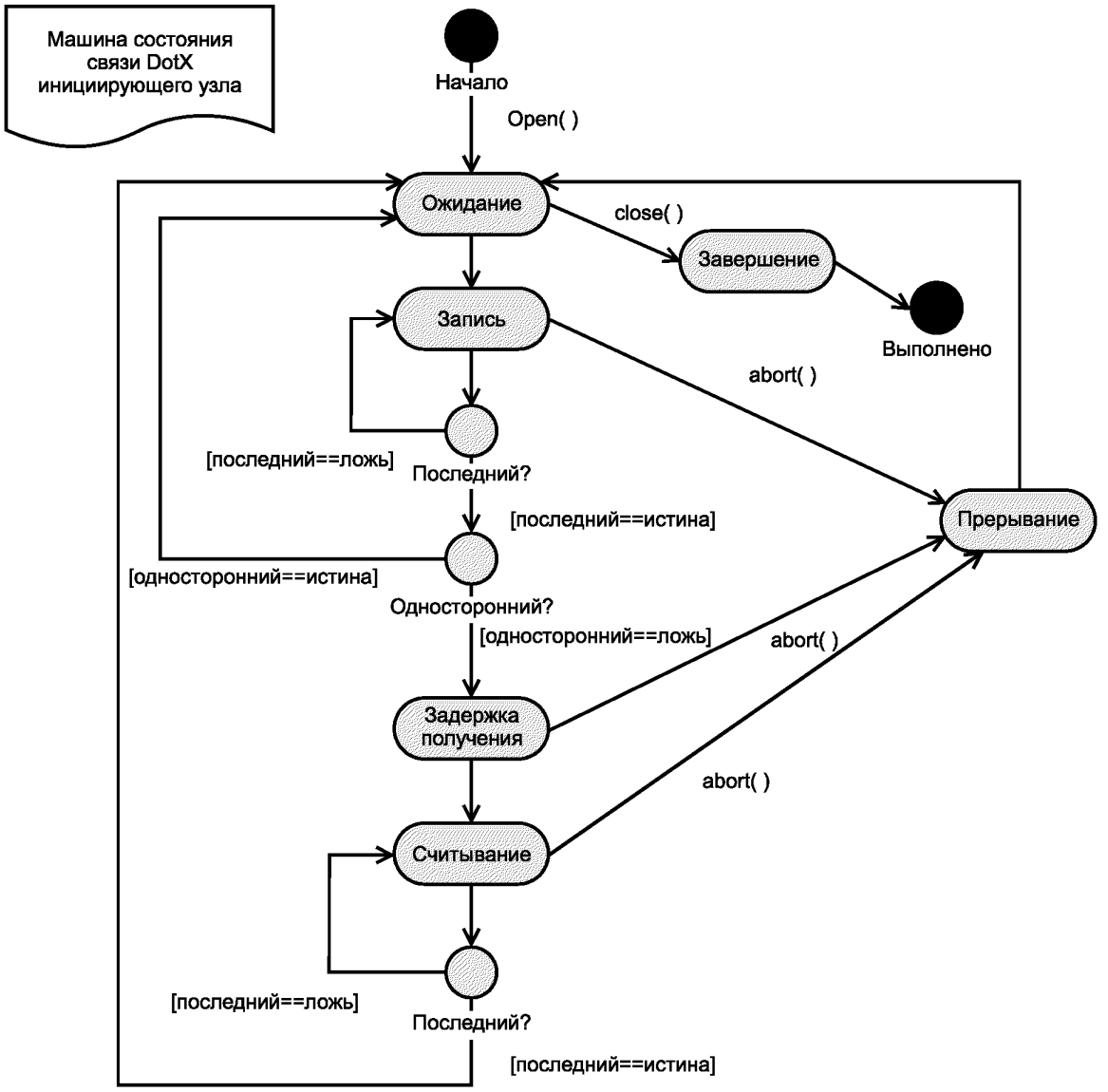


Рисунок С.4 — Диаграмма состояний связи для инициирующего узла

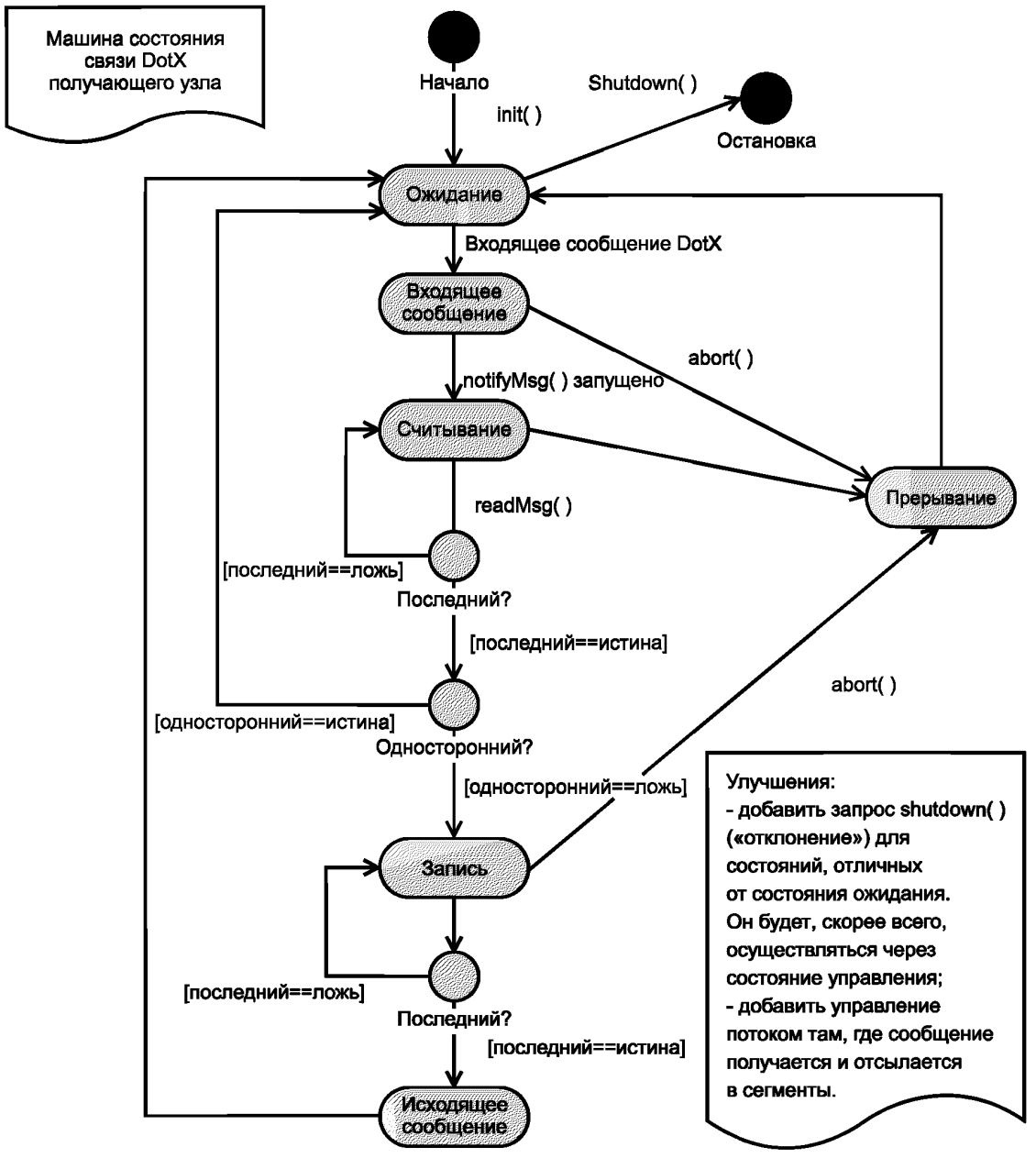


Рисунок С.5 — Диаграмма состояний связи для принимающего узла

С.11 Последовательность связи

На диаграмме последовательности[1)] (рисунок С. 6) приведена типичная последовательность вызовов между инициирующим и принимающим узлами для односторонней операции. Информация передается от уровня ИИЭР 1451.0 на инициирующем узле к уровню ИИЭР 1451.0 на принимающем узле.

В данном примере предполагается, что получающая сторона имеет существенные ограничения памяти, которые требуют осуществления многократных вызовов «readMsg()». Кроме того, показано использование потока ИИЭР 1451.X при обработке данных на стороне получателя. Вызов «notifyMsg( )» может возвращаться только

[1)] На диаграмме последовательности шкала времени направлена вниз. Вызов стандартной блокирующей функции или метода обозначен сплошной стрелкой с закрашенным наконечником. Обратное действие представлено в виде пунктирной стрелки с незакрашенным наконечником. Для обозначения асинхронных вызовов методов используется сплошная стрелка с незакрашенным наконечником.

после завершения обработки данных на стороне получателя. Кроме того, вызов «openQoS( )» использован для обеспечения требований «качества сервиса» связи для уровня ИИЭР 1451.X.

На инициирующем узле уровень ИИЭР 1451.0 вызывает методы «openQoS()», «writeMsg( )» и «close()». На принимающем узле уровень ИИЭР 1451.X вызывает метод «receiveMsg()» для информирования уровня ИИЭР 1451.0 о начале новой операции связи. Уровень ИИЭР 1451.0 осуществляет два обратных вызова «readMsg( )» для уровня ИИЭР 1451.X для получения информации по этапам. Следует отметить, что вызовы «Open( )», «openQoS( )» и «close( )» осуществляются только на инициирующем узле.

Несмотря на то что представлено только одно асинхронное сообщение между двумя уровнями ИИЭР 1451.X на инициирующем и принимающем узлах, фактическое число сетевых сообщений выходит за рамки рассмотрения настоящего стандарта. Уровню ИИЭР 1451.X может потребоваться отправка многочисленных сетевых сообщений в обоих направлениях для завершения операции связи. Такие вопросы, как управление потоком, сегментация, повторная сборка, повторные попытки и предотвращение перегрузок, могут также нуждаться в проработке.

На диаграмме последовательности (рисунок С.7) показан «двусторонний» поток информации от уровня ИИЭР 1451.0 на инициирующем узле к уровню ИИЭР 1451.0 на принимающем узле в случае использования интерфейса «Network» («Сетевой»). Принимающий узел генерирует ответ, который передается обратно на инициирующий узел. Для наглядности используется вызов «open( )», так как «качество сервиса» связи по умолчанию является приемлемым для инициирующего узла.

В случаях, когда требуется повышенная производительность, рекомендуется использовать многопоточную архитектуру для обработки перекрывающихся связей. Следует отметить, что вызовы «notifyMsg()» и «notifyRsp()» не блокируют в течение срока обработки. Практическая реализация устройства должна предусматривать некоторый механизм «пробуждения» или запуска связи обработки, которая завершит необходимый процесс обработки. Хотя на рисунке показана работа в одном и том же объекте, ситуация может отличаться и зависит от выбора устройства реализации.

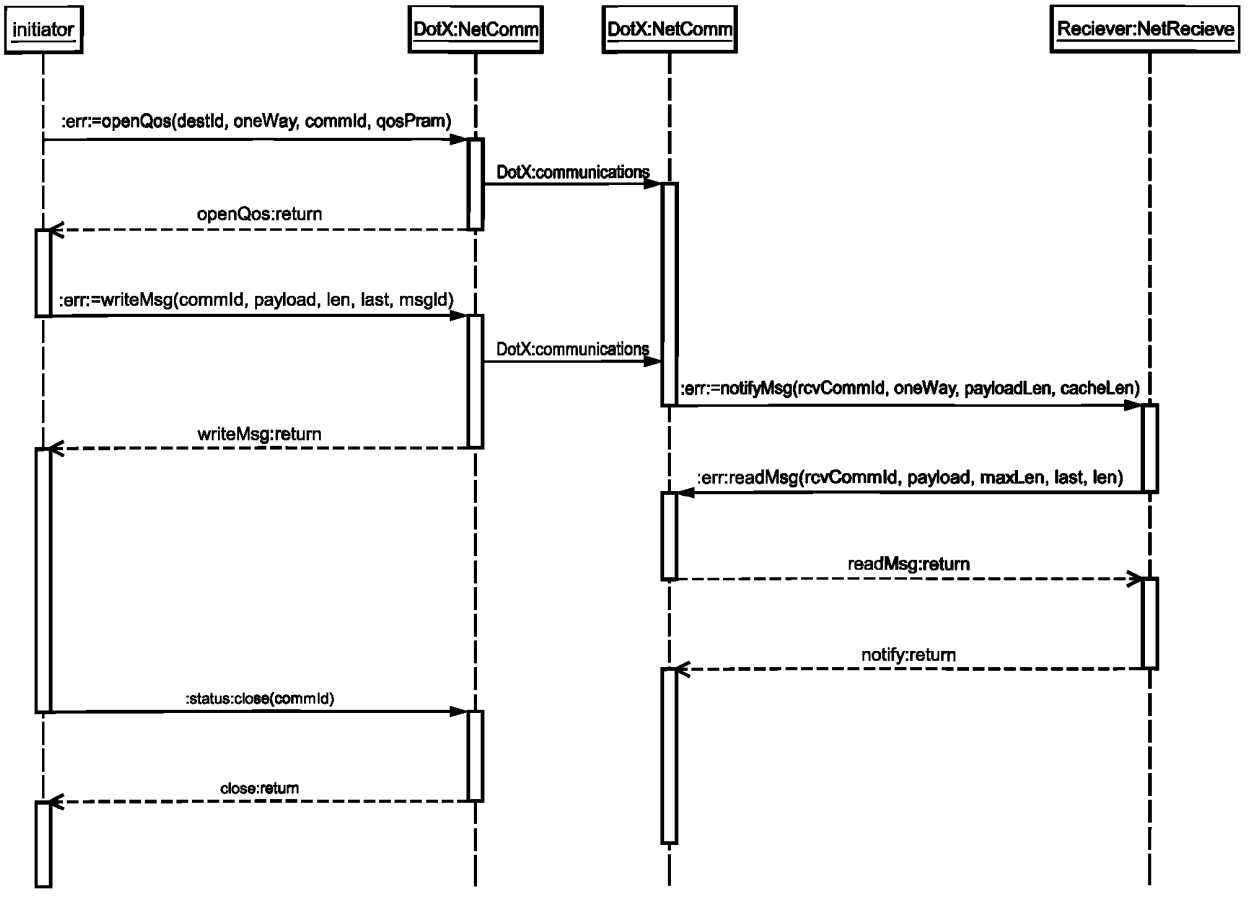


Рисунок С.6 — Простая последовательность связей

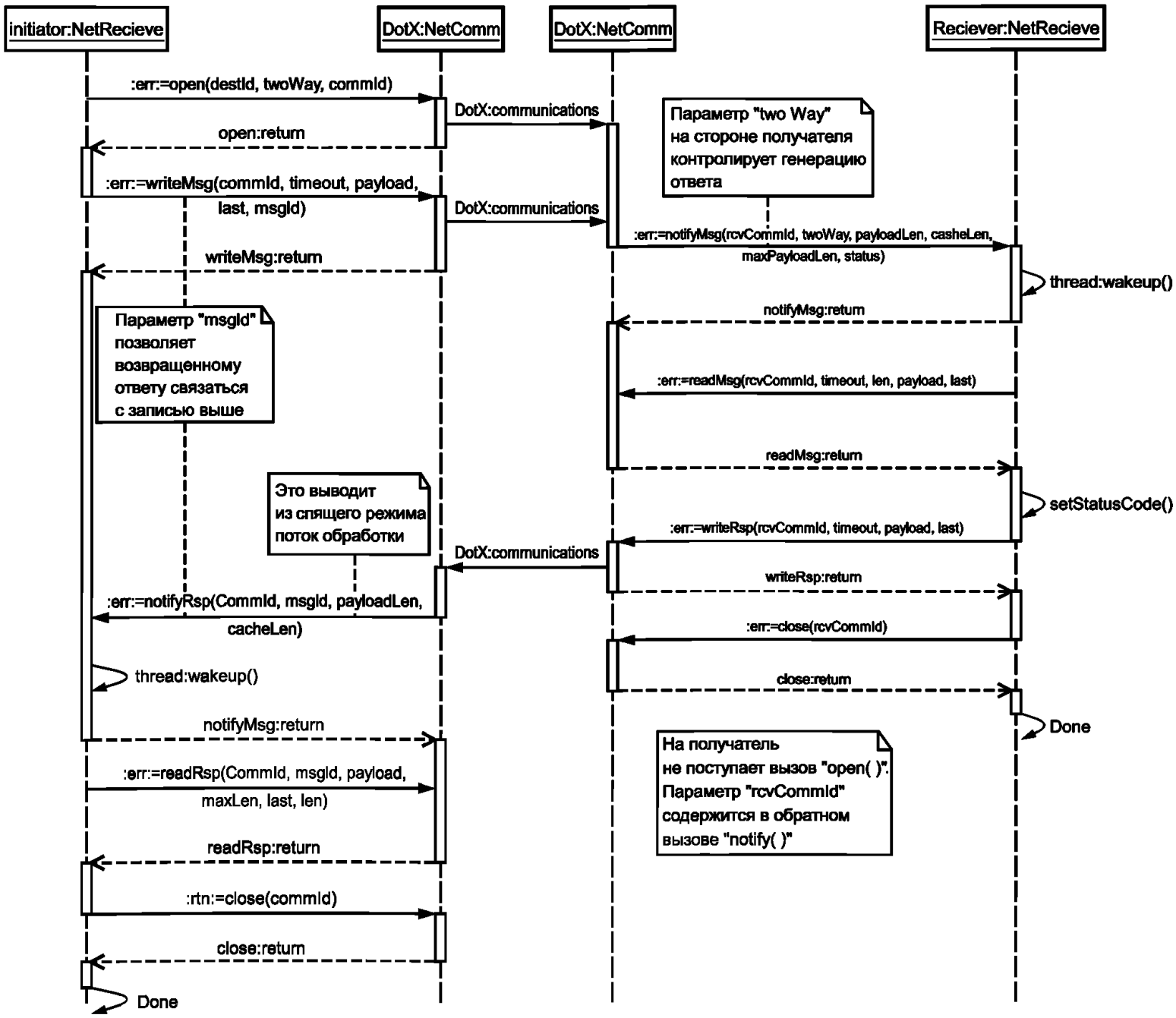


Рисунок С.7 — Двусторонняя последовательность связей

Приложение D
(справочное)

XML-схема для текстовых ЭТДП

D.1 Введение в текстовые ЭТДП

Текстовые ЭТДП предоставляют изготовителю механизм встраивания текстовой информации в интеллектуальный преобразователь. В настоящем стандарте определены следующие текстовые ЭТДП:

- ЭТДП мета-идентификации (справочная схема);
- ЭТДП идентификации канала преобразователя (справочная схема);
- ЭТДП идентификации калибровки (справочная схема);
- командная ЭТДП (обязательная схема);
- ЭТДП места нахождения и заголовка (справочная схема);
- ЭТДП географического места нахождения (обязательная схема, предоставляемая другой организацией).

Любая текстовая ЭТДП имеет структуру, которая инкапсулирует текстовую информацию как «блок данных» произвольной длины. Структура ЭТДП совмещает многочисленные блоки для поддержки содержания на разных языках. Единственным ограничением, налагаемым настоящим стандартом, является предоставление каждого блока данных в виде «образца документа» XML.

XML является гибким мета-языком разметки, в котором тэги разметки могут быть определены так, как требует контекст того информационного домена, в котором используются данные тэги. Такие тэги должны быть определены и организованы в соответствии с набором правил. Существуют два доступных метода описания тэгов: «Document Type Definition (DTD)» («Определение типа документа») или схема XML, которая при дальнейшем упоминании в настоящем приложении будет называться «схема».

Схема может быть встроенной (полностью содержаться внутри образца документа), а также может быть внешней либо внешней с внутренними дополнениями или расширениями. В настоящем приложении представлены схемы, которые могут служить в качестве внешних схем для всех текстовых ЭТДП, определенных в настоящем стандарте. Данная схема имеет два преимущества:

- применение внешней схемы минимизирует размер образца документа;
- применение общей схемы способствует стандартизации семантики домена.

Образец документа внутри текстовой ЭТДП не обязательно должен ссылаться на данную схему. Изготовитель может принять для себя, что любой обработчик образца документа может получить данную схему без явной ссылки, содержащейся в образце документа.

Схемы предоставлены на английском языке. Пользователь вправе перевести их на любой необходимый ему язык.

D.2 Схема

Схемы для мета-идентификационной ЭТДП, ЭТДП канала преобразователя, идентификационной ЭТДП калибровки предоставляются и базируются на информации о соответствующей ЭТДП, определенной в стандарте ИИЭР 1451.2—1997. Уровень ИИЭР 1451.0 не требует применения данных схем. Предложенная схема также предоставлена для ЭТДП места нахождения и заголовка. Схема для командной ЭТДП требуется в случае, если предоставлена командная ЭТДП. Схема для ЭТДП географического места нахождения, которая предоставляется в блоке данных, должна быть составлена на языке географической разметки «Geography Markup Language (GML)», описанном в проекте ИСО 19136 «Географическая информация. Язык географической разметки» 2005 г. или более поздней версии. Схема для ЭТДП с расширенным набором единиц измерения требуется, если предоставлена ЭТДП с расширенным набором единиц измерения. Более детальная информация представлена в 5.5.2.8. Электронные копии данных схем могут быть найдены по следующему URL: http://grouper.ieee.org/groups/1451/0/1451HTTPAPI/.

D.3 Включаемый файл «SmartTransducerDataModel.xsd»

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" xmlns:SOAPENC="
http://schemas.xmlsoap.org/soap/encoding"
xmlns:corba="http://www.omg.org/IDL-WSDL/1.0/"
xmlns:stml="http://grouper.ieee.org/groups/1451/0/1451HTTPAPI"
targetNamespace="http://grouper.ieee.org/groups/1451/0/1451HTTPAPI"
elementFormDefault="qualified" attributeFormDefault="unqualified">
   <!--This is IEEE 1451.0 data types-->
   <xs:simpleType name="Int8">
      <xs:restriction base="xs:byte"/>
   </xs:simpleType>
```

```
<xs:simpleType name="Int16">
   <xs:restriction base="xs:short"/>
</xs:simpleType>
<xs:simpleType name="Int32">
   <xs:restriction base="xs:int"/>
</xs:simpleType>
<xs:simpleType name="UInt8">
   <xs:restriction base="xs:unsignedByte"/>
</xs:simpleType>
<xs:simpleType name="UInt16">
   <xs:restriction base="xs:unsignedShort"/>
</xs:simpleType>
<xs:simpleType name="UInt32">
   <xs:restriction base="xs:unsignedInt"/>
</xs:simpleType>
<xs:simpleType name="Float32">
   <xs:restriction base="xs:float"/>
</xs:simpleType>
<xs:simpleType name="Float64">
   <xs:restriction base="xs:double"/>
</xs:simpleType>
<xs:simpleType name="_Boolean">
   <xs:annotation>
      <xs:documentation source="boolean «/>
   </xs:annotation>
   <xs:restriction base="xs:boolean"/>
</xs:simpleType>
<xs:simpleType name="Octet">
   <xs:restriction base="xs:unsignedByte"/>
</xs:simpleType>
<xs:simpleType name="_String">
   <xs:annotation>
      <xs:documentation source="string"/>
   </xs:annotation>
   <xs:restriction base="xs:string"/>
</xs:simpleType>
<xs:simpleType name="Int8Array">
   <xs:annotation>
      <xs:documentation source="Array of Int8"/>
   </xs:annotation>
   <xs:list itemType="stml:Int8"/>
</xs:simpleType>
<xs:simpleType name="Int16Array">
   <xs:annotation>
      <xs:documentation source="Array of Int16"/>
   </xs:annotation>
   <xs:list itemType="stml:Int16"/>
</xs:simpleType>
<xs:simpleType name="Int32Array">
   <xs:annotation>
      <xs:documentation source="Array of Int32"/>
   </xs:annotation>
   <xs:list itemType="stml:Int32"/>
</xs:simpleType>
<xs:simpleType name="UInt8Array">
   <xs:annotation>
      <xs:documentation source="Array of UInt8"/>
   </xs:annotation>
   <xs:list itemType="stml:UInt8"/>
</xs:simpleType>
<xs:simpleType name="UInt16Array">
```

```
   <xs:annotation>
      <xs:documentation source="Array of UInt16"/>
   </xs:annotation>
   <xs:list itemType="stml:UInt16"/>
</xs:simpleType>
<xs:simpleType name="UInt32Array">
   <xs:annotation>
      <xs:documentation source="Array of UInt32"/>
   </xs:annotation>
   <xs:list itemType="stml:UInt32"/>
</xs:simpleType>
<xs:simpleType name="Float32Array">
   <xs:annotation>
      <xs:documentation source="Array of Float32"/>
   </xs:annotation>
   <xs:list itemType="stml:Float32"/>
</xs:simpleType>
<xs:simpleType name="Float64Array">
   <xs:annotation>
      <xs:documentation source="Array of Float64"/>
   </xs:annotation>
   <xs:list itemType="stml:Float64"/>
</xs:simpleType>
<xs:simpleType name="OctetArray">
   <xs:annotation>
      <xs:documentation source="Array of Octet"/>
   </xs:annotation>
   <xs:list itemType="stml:Octet"/>
</xs:simpleType>
<xs:simpleType name="BooleanArray">
   <xs:annotation>
      <xs:documentation source="Array of _Boolean"/>
   </xs:annotation>
   <xs:list itemType="stml:_Boolean"/>
</xs:simpleType>
<xs:simpleType name="StringArray">
   <xs:annotation>
      <xs:documentation source="Array of _String"/>
   </xs:annotation>
   <xs:list itemType="stml:_String"/>
</xs:simpleType>
<xs:simpleType name="ErrorCode" final="restriction">
   <xs:restriction base="xs:string">
      <xs:enumeration value="NO_ERROR"/>
      <xs:enumeration value="INVALID_COMMID"/>
      <xs:enumeration value="UNKNOWN_DESTID"/>
      <xs:enumeration value="TIMEOUT"/>
      <xs:enumeration value="NETWORK_FAILURE"/>
      <xs:enumeration value="NETWORK_CORRUPTION"/>
      <xs:enumeration value="MEMORY"/>
      <xs:enumeration value="QOS_FAILURE"/>
      <xs:enumeration value="MCAST_NOT_SUPPORTED"/>
      <xs:enumeration value="UNKNOWN_GROUPID"/>
      <xs:enumeration value="UNKNOWN_MODULEID"/>
      <xs:enumeration value="UNKNOWN_MSGID"/>
      <xs:enumeration value="NOT_GROUP_MEMBER"/>
      <xs:enumeration value="ILLEGAL_MODE"/>
      <xs:enumeration value="LOCKED_RESOURCE"/>
      <xs:enumeration value="FATAL_TEDS_ERROR"/>
      <xs:enumeration value="NON_FATAL_TEDS_ERROR"/>
      <xs:enumeration value="CLOSE_ON_LOCKED_RESOURCE"/>
```

```
            <xs:enumeration value="LOCK_BROKEN"/>
            <xs:enumeration value="NETWORK_RESOURCE_EXCEEDED"/>
            <xs:enumeration value="MEMORY_RESOURCE_EXCEEDED"/>
         </xs:restriction>
      </xs:simpleType>
      <xs:simpleType name="TypeCode">
         <xs:annotation>
            <xs:documentation>Each valid type of 1451.0 Argument has a unique typeCode.
</xs:documentation>
         </xs:annotation>
         <xs:restriction base="xs:string">
            <xs:enumeration value="UNKNOWN_TC"/>
            <xs:enumeration value="INT8_TC"/>
            <xs:enumeration value="INT16_TC"/>
            <xs:enumeration value="INT32_TC"/>
            <xs:enumeration value="UINT8_TC"/>
            <xs:enumeration value="UINT16_TC"/>
            <xs:enumeration value="UINT32_TC"/>
            <xs:enumeration value="FLOAT32_TC"/>
            <xs:enumeration value="FLOAT64_TC"/>
            <xs:enumeration value="STRING_TC"/>
            <xs:enumeration value="OCTET_TC"/>
            <xs:enumeration value="BOOLEAN_TC"/>
            <xs:enumeration value="TIME_INSTANCE_TC"/>
            <xs:enumeration value="TIME_DURATION_TC"/>
            <xs:enumeration value="QOS_PARAMS_TC"/>
            <xs:enumeration value="INT8_ARRAY_TC"/>
            <xs:enumeration value="INT16_ARRAY_TC"/>
            <xs:enumeration value="INT32_ARRAY_TC"/>
            <xs:enumeration value="UINT8_ARRAY_TC"/>
            <xs:enumeration value="UINT16_ARRAY_TC"/>
            <xs:enumeration value="UINT32_ARRAY_TC"/>
            <xs:enumeration value="FLOAT16_ARRAY_TC"/>
            <xs:enumeration value="FLOAT32_ARRAY_TC"/>
            <xs:enumeration value="STRING_ARRAY_TC"/>
            <xs:enumeration value="OCTET_ARRAY_TC"/>
            <xs:enumeration value="BOOLEAN_ARRAY_TC"/>
            <xs:enumeration value="TIME_INSTANCE_ARRAY_TC"/>
            <xs:enumeration value="TIME_DURATION_ARRAY_TC"/>
         </xs:restriction>
      </xs:simpleType>
      <xs:simpleType name="UUID">
         <xs:list itemType="xs:short"/>
      </xs:simpleType>
      <xs:simpleType name="ErrorCodeSource">
         <xs:restriction base="xs:string">
            <xs:enumeration value="LOCAL_1451_0_LAYER"/>
            <xs:enumeration value="LOCAL_1451_X_LAYER"/>
            <xs:enumeration value="REMOTE_1451_0_LAYER"/>
            <xs:enumeration value="REMOTE_1451_X_LAYER"/>
            <xs:enumeration value="REMOTE_APPLICATION_LAYER"/>
         </xs:restriction>
      </xs:simpleType>
      <xs:element name="Argument" type="stml:ArgumentType"/>
      <xs:element name="ArgumentArray" type="stml:ArgumentArrayType"/>
      <xs:element name="QoSParam" type="stml:QoSParamType"/>
      <xs:element name="TimeDurationArray" type="stml:TimeInstanceArrayType"/>
      <xs:element name="TimeDuration" type="stml:TimeDurationType"/>
      <xs:element name="TimeInstanceArray" type="stml:TimeInstanceArrayType"/>
      <xs:element name="TimeInstance" type="stml:TimeInstanceType"/>
      <xs:complexType name="TimeDurationType">
```

```
      <xs:sequence>
         <xs:element name="secs" type="stml:UInt32"/>
         <xs:element name="nsecs" type="stml:UInt32"/>
      </xs:sequence>
   </xs:complexType>
   <xs:complexType name="QoSParamType">
      <xs:annotation>
         <xs:documentation source="QoSParams structure S"/>
      </xs:annotation>
      <xs:sequence>
         <xs:element name="service" type="stml:_Boolean"/>
         <xs:element name="period" type="stml:TimeDurationType"/>
         <xs:element name="transmitSize" type="stml:UInt32"/>
         <xs:element name="accesLatency" type="stml:TimeDurationType"/>
         <xs:element name="transmitLatency" type="stml:TimeDurationType"/>
      </xs:sequence>
   </xs:complexType>
   <xs:complexType name="TimeInstanceArrayType">
      <xs:sequence>
         <xs:element name="TimeInstance" type="stml:TimeDurationType" minOccurs="0"
maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="size" type="xs:short"/>
   </xs:complexType>
   <xs:complexType name="TimeInstanceType">
      <xs:sequence>
         <xs:element name="secs" type="stml:UInt32"/>
         <xs:element name="nsecs" type="stml:UInt32"/>
         <xs:element name="epoch" type="stml:UInt8"/>
      </xs:sequence>
   </xs:complexType>
   <xs:complexType name="TimeDurationArrayType">
      <xs:sequence>
         <xs:element name="timeDuration" type="stml:TimeDurationType" minOccurs="0"
maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="size" type="xs:short"/>
   </xs:complexType>
   <xs:complexType name="ArgumentType">
      <xs:annotation>
         <xs:documentation source="A generic data container"/>
      </xs:annotation>
      <xs:sequence>
         <xs:element name="discriminator" type="stml:TypeCode"/>
      <xs:choice>
            <xs:element name="ValueError" type="stml:_Boolean"
minOccurs="0"/>
            <xs:element name="valueInt8" type="stml:UInt8"
minOccurs="0"/>
            <xs:element name="valueInt16" type="stml:Int16"
minOccurs="0"/>
            <xs:element name="valueInt32" type="stml:Int32"
minOccurs="0"/>
            <xs:element name="valueUInt8" type="stml:UInt8"
minOccurs="0"/>
            <xs:element name="valueUInt16" type="stml:UInt16"
minOccurs="0"/>
            <xs:element name="valueUIn32" type="stml:UInt32"
minOccurs="0"/>
            <xs:element name="valueFloat32" type="stml:Float32"
minOccurs="0"/>
```

```
            <xs:element name="valueFloat64" type="stml:Float64"
minOccurs="0"/>
            <xs:element name="valueString" type="stml:_String"
minOccurs="0"/>
            <xs:element name="valueOctet" type="stml:Octet"
minOccurs="0"/>
            <xs:element name="valueBoolean" type="stml:_Boolean"
minOccurs="0"/>
            <xs:element name="valueTimeInstance" type="stml:TimeDurationType"
minOccurs="0"/>
            <xs:element name="valueTimeDuration" type="stml:TimeDurationType"
minOccurs="0"/>
            <xs:element name="valueQosParams" type="stml:QoSParamType" minOccurs="0"/>
            <xs:element name="valueUInt8Array" type="stml:UInt8Array" minOccurs="0"/>
            <xs:element name="valueUInt16Array" type="stml:UInt16Array" minOccurs="0"/>
            <xs:element name="valueUInt32Array" type="stml:UInt32Array" minOccurs="0"/>
            <xs:element name="valueFloat32Array"
type="stml:Float32Array" minOccurs="0"/>
            <xs:element name="valueFloat64Array"
type="stml:Float64Array" minOccurs="0"/>
            <xs:element name="valueStringArray" type="stml:StringArray" minOccurs="0"/>
            <xs:element name="valueOctetArray" type="stml:OctetArray" minOccurs="0"/>
            <xs:element name="valueBooleanArray"
type="stml:BooleanArray" minOccurs="0"/>
            <xs:element name="valueTimeInstanceArray"
type="stml:TimeInstanceArrayType" minOccurs="0"/>
            <xs:element name="valueTimeDurationArray"
type="stml:TimeDurationArrayType" minOccurs="0"/>
         </xs:choice>
      </xs:sequence>
   </xs:complexType>
   <xs:complexType name="Units">
      <xs:annotation>
         <xs:documentation source="Definitions of the SI base units are given in The
International System of Units (SI), «/>
      </xs:annotation>
      <xs:sequence>
         <xs:element name="interpret" type="stml:UInt8"/>
         <xs:element name="radians" type="stml:UInt8"/>
         <xs:element name="steradians" type="stml:UInt8"/>
         <xs:element name="meters" type="stml:UInt8"/>
         <xs:element name="kilograms" type="stml:UInt8"/>
         <xs:element name="seconds" type="stml:UInt8"/>
         <xs:element name="amperes" type="stml:UInt8"/>
         <xs:element name="kelvins" type="stml:UInt8"/>
         <xs:element name="moles" type="stml:UInt8"/>
         <xs:element name="candelas" type="stml:UInt8"/>
      </xs:sequence>
   </xs:complexType>
   <xs:complexType name="ArgumentArrayType">
      <xs:sequence>
      <xs:element name="argument" type="stml:ArgumentType"
minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="size" type="xs:short"/>
   </xs:complexType>
</xs:schema>
```

D.4 Включаемый файл «TextTEDS.xsd»

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
```

```
xmlns:stml="http://grouper.ieee.org/groups/1451/0/1451HTTPAPI"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://grouper.ieee.org/groups/1451/0/1451HTTPAPI"
elementFormDefault="qualified" attributeFormDefault="unqualified">
   <xs:include schemaLocation="SmartTransducerDataModel.xsd"/>
   <xs:include schemaLocation="MetaIdentificationTEDS.xsd"/>
   <xs:include
schemaLocation="Transducer ChannelIdentificationTEDS.xsd"/>
   <xs:include schemaLocation="CalibrationIdentificationTEDS.xsd"/>
   <xs:include schemaLocation="CommandsTEDS.xsd"/>
   <xs:include schemaLocation="LocationAndTitleTEDS.xsd"/>
   <xs:include schemaLocation="GeographicLocationTEDS.xsd"/>
   <xs:element name="TextTEDS" type="stml:TextTEDSType"
abstract="true"/>
   <xs:complexType name="TextTEDSDataBlockType">
      <xs:annotation>
         <xs:documentation>Structure of a Text-based TEDS data
block</xs:documentation>
      </xs:annotation>
      <xs:sequence>
         <xs:element name="TEDSID">
            <xs:annotation>
               <xs:documentation>TEDS Identification
Header</xs:documentation>
            </xs:annotation>
            <xs:complexType>
               <xs:sequence>
                  <xs:element name="Type" type="xs:short" default="3"/>
                  <xs:element name="Length" type="xs:short" default="4"/>
                  <xs:element name="Value" type="stml:UInt8Array"/>
               </xs:sequence>
            </xs:complexType>
         </xs:element>
         <xs:element name="NumLang">
            <xs:annotation>
               <xs:documentation>The number N of different language blocks in this TEDS</
xs:documentation>
            </xs:annotation>
            <xs:complexType>
               <xs:sequence>
                  <xs:element name="Type" type="xs:short" default="10"/>
                  <xs:element name="Length" type="xs:short"
default="1"/>
                  <xs:element name="Value" type="stml:UInt8"/>
               </xs:sequence>
            </xs:complexType>
         </xs:element>
         <xs:element name="DirBlock">
            <xs:annotation>
               <xs:documentation>Language block description This block is repeated N
times</xs:documentation>
            </xs:annotation>
            <xs:complexType>
               <xs:sequence>
                  <xs:element name="Type" type="xs:short" default="11"/>
                  <xs:element name="Length" type="xs:short"/>
                  <xs:sequence>
                     <xs:element name="LangCode">
                        <xs:annotation>
                           <xs:documentation>Language code from ISO 639 (two letters in
USASCII)</xs:documentation>
```

```
                        </xs:annotation>
                        <xs:complexType>
                           <xs:sequence>
                              <xs:element name="Type" type="xs:short" default="20"/>
                              <xs:element name="Length" type="xs:short" default="2"/>
                              <xs:element name="Value"
type="stml:UInt8"/>
                           </xs:sequence>
                        </xs:complexType>
                     </xs:element>
                     <xs:element name="LangOffset">
                        <xs:annotation>
                           <xs:documentation>Language offset</xs:documentation>
                        </xs:annotation>
                        <xs:complexType>
                           <xs:sequence>
                              <xs:element name="Type" type="xs:short" default="21"/>
                              <xs:element name="Length" type="xs:short" default="4"/>
                              <xs:element name="Value" type="stml:UInt32"/>
                           </xs:sequence>
                        </xs:complexType>
                     </xs:element>
                     <xs:element name="LangLength">
                        <xs:annotation>
                           <xs:documentation>Language length =
LL</xs:documentation>
                        </xs:annotation>
                        <xs:complexType>
                           <xs:sequence>
                              <xs:element name="Type" type="xs:short" default="22"/>
                              <xs:element name="Length" type="xs:short" default="4"/>
                              <xs:element name="Value"
type="stml:UInt32"/>
                           </xs:sequence>
                        </xs:complexType>
                     </xs:element>
                     <xs:element name="Compress">
                        <xs:annotation>
                           <xs:documentation>Enumeration identifying the compression
technique used.</xs:documentation>
                        </xs:annotation>
                        <xs:complexType>
                           <xs:sequence>
                              <xs:element name="Type" type="xs:short" default="23"/>
                              <xs:element name="Length" type="xs:short" default="1"/>
                              <xs:element name="Value"
type="stml:UInt8"/>
                           </xs:sequence>
                        </xs:complexType>
                     </xs:element>
                  </xs:sequence>
               </xs:sequence>
            </xs:complexType>
         </xs:element>
         <xs:element name="SubSum">
            <xs:annotation>
               <xs:documentation>Non-displayable data
checksum</xs:documentation>
            </xs:annotation>
            <xs:complexType>
               <xs:sequence>
```

```
                  <xs:element name="Type" type="xs:short" default="12"/>
                  <xs:element name="Length" type="xs:short"
default="2"/>
                  <xs:element name="Value" type="stml:UInt16"/>
               </xs:sequence>
            </xs:complexType>
         </xs:element>
      </xs:sequence>
   </xs:complexType>
   <xs:simpleType name="LanguageCodes">
      <xs:restriction base="xs:string">
         <xs:enumeration value="aa"/>
         <xs:enumeration value="da"/>
         <xs:enumeration value="de"/>
         <xs:enumeration value="en"/>
         <xs:enumeration value="es"/>
         <xs:enumeration value="eu"/>
         <xs:enumeration value="fi"/>
         <xs:enumeration value="fr"/>
         <xs:enumeration value="ga"/>
         <xs:enumeration value="it"/>
         <xs:enumeration value="nl"/>
         <xs:enumeration value="no"/>
         <xs:enumeration value="pl"/>
         <xs:enumeration value="pt"/>
         <xs:enumeration value="ru"/>
         <xs:enumeration value="sv"/>
         <xs:enumeration value="vi"/>
         <xs:enumeration value="zu"/>
      </xs:restriction>
   </xs:simpleType>
   <xs:complexType name="TextTEDSType">
      <xs:annotation>
         <xs:documentation>This is a class of optional TEDS. The
function of these TEDS is to provide information for display to an
operator. There are six TEDS listed in Table 17 that fall into this category. They
are the Meta-Identification TEDS, Transducer Channel Identification TEDS, Calibration-
Identification TEDS, Commands TEDS and the Location and Title TEDS and the Geographic
Location TEDS.
</xs:documentation>
      </xs:annotation>
      <xs:sequence>
         <xs:element name="TEDSLength" type="stml:UInt32"/>
         <xs:element name="TextTEDSDataBlock"
type="stml:TextTEDSDataBlockType"/>
         <xs:element name="Checksum" type="stml:UInt16"/>
      </xs:sequence>
   </xs:complexType>
</xs:schema>
```

D.5 ЭТДП мета-идентификации (MetaIdentificationTEDS.xsd)

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:stml="http://grouper.ieee.org/groups/1451/0/1451HTTPAPI"
targetNamespace="http://grouper.ieee.org/groups/1451/0/1451HTTPAPI"
elementFormDefault="qualified" attributeFormDefault="unqualified">
   <xs:include schemaLocation="SmartTransducerDataModel.xsd"/>
   <xs:include schemaLocation="TextTEDS.xsd"/>
   <xs:element name="Meta-IdentificationTEDS" type="stml:Meta-
IdentificationTEDSType" substitutionGroup="stml:TextTEDS">
      <xs:annotation>
```

```
         <xs:documentation>1451 Smart Sensor Meta-Identification TEDS schema</
xs:documentation>
      </xs:annotation>
   </xs:element>
   <xs:complexType name="Meta-IdentificationTEDSDataBlockType">
      <xs:sequence>
         <xs:element name="ManufacturerId" type="stml:_String"
minOccurs="0" maxOccurs="255"/>
         <xs:element name="ModelNo" type="stml:_String" minOccurs="0" maxOccurs="255"/>
         <xs:element name="VersionCode" type="stml:_String"
minOccurs="0" maxOccurs="255"/>
         <xs:element name="SerialNo" type="stml:_String" minOccurs="0" maxOccurs="255"/>
         <xs:element name="DateCode" type="stml:_String" minOccurs="0" maxOccurs="255"/>
         <xs:element name="NumOfChannelGrouping" type="stml:UInt8"/>
         <xs:element name="GroupName" type="stml:_String" minOccurs="0" maxOccurs="255"/>
         <xs:element name="ProductDescription" type="stml:_String"
minOccurs="0" maxOccurs="65535"/>
      </xs:sequence>
   </xs:complexType>
   <xs:complexType name="Meta-IdentificationTEDSType">
      <xs:complexContent>
         <xs:extension base="stml:TextTEDSType">
            <xs:sequence>
               <xs:element name="XMLText" type="stml:Meta-
IdentificationTEDSDataBlockType"/>
               <xs:element name="XMLSum" type="stml:UInt16"/>
            </xs:sequence>
         </xs:extension>
      </xs:complexContent>
   </xs:complexType>
</xs:schema>
```

D.6 ЭТДП идентификации канала преобразователя (Transducer ChannelIdentificationTEDS.xsd)

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:stml="http://grouper.ieee.org/groups/1451/0/1451HTTPAPI"
targetNamespace="http://grouper.ieee.org/groups/1451/0/1451HTTPAPI"
elementFormDefault="qualified" attributeFormDefault="unqualified">
   <xs:include schemaLocation="SmartTransducerDataModel.xsd"/>
   <xs:include schemaLocation="TextTEDS.xsd"/>
   <xs:element name="Transducer ChannelIdentificationTEDS"
type="stml:Transducer ChannelIdentificationTEDSType"
substitutionGroup="stml:TextTEDS">
      <xs:annotation>
         <xs:documentation>1451 Smart Sensor
Transducer ChannelIdentification TEDS schema</xs:documentation>
      </xs:annotation>
   </xs:element>
   <xs:complexType
name="Transducer ChannelIdentificationTEDSDataBlockType">
      <xs:sequence>
         <xs:element name="ManufactruereID" type="stml:_String"/>
         <xs:element name="ModelNo" type="stml:_String" minOccurs="0" maxOccurs="255"/>
         <xs:element name="VersionCode" type="stml:_String"
minOccurs="0" maxOccurs="255"/>
         <xs:element name="SerialNo" type="stml:_String" minOccurs="0" maxOccurs="255"/>
         <xs:element name="ChannelDescription" type="stml:_String"
minOccurs="0" maxOccurs="65535"/>
      </xs:sequence>
   </xs:complexType>
   <xs:complexType name="Transducer ChannelIdentificationTEDSType">
      <xs:complexContent>
```

```
         <xs:extension base="stml:TextTEDSType">
            <xs:sequence>
               <xs:element name="XMLText"
type="stml:Transducer ChannelIdentificationTEDSDataBlockType"/>
               <xs:element name="XMLSum" type="stml:UInt16"/>
            </xs:sequence>
         </xs:extension>
      </xs:complexContent>
   </xs:complexType>
</xs:schema>
```

D.7 ЭТДП идентификации калибровки (CalibrationIdentificationTEDS.xsd)

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:stml="http://grouper.ieee.org/groups/1451/0/1451HTTPAPI"
targetNamespace="http://grouper.ieee.org/groups/1451/0/1451HTTPAPI"
elementFormDefault="qualified" attributeFormDefault="unqualified">
   <xs:include schemaLocation="SmartTransducerDataModel.xsd"/>
   <xs:include schemaLocation="SmartTransducerTEDS.xsd"/>
   <xs:include schemaLocation="TextTEDS.xsd"/>
   <xs:element name="CalibrationIdentificationTEDS"
type="stml:CalibrationIdentificationTEDSType"
substitutionGroup="stml:TextTEDS">
      <xs:annotation>
         <xs:documentation>1451 Smart Sensor
CalibrationIdentificationTEDS schema</xs:documentation>
      </xs:annotation>
   </xs:element>
   <xs:complexType name="CalibrationIdentificationTEDSDataBlockType">
      <xs:sequence>
         <xs:element name="CalLabNo" type="stml:_String"/>
         <xs:element name="CalTech" type="stml:_String" minOccurs="0" maxOccurs="255"/>
         <xs:element name="CalibrationDescrption" type="stml:_String" minOccurs="0"
maxOccurs="65535"/>
         <xs:element name="CalDate" type="stml:_String"/>
         <xs:element name="CalProc" type="stml:_String" minOccurs="0" maxOccurs="255"/>
         <xs:element name="CalStd" type="stml:_String" minOccurs="0" maxOccurs="255"/>
      </xs:sequence>
   </xs:complexType>
   <xs:complexType name="CalibrationIdentificationTEDSType">
      <xs:complexContent>
         <xs:extension base="stml:TextTEDSType">
            <xs:sequence>
               <xs:element name="XMLText">
                  <xs:complexType>
                     <xs:complexContent>
                        <xs:extension
base="stml:CalibrationIdentificationTEDSDataBlockType"/>
                     </xs:complexContent>
                  </xs:complexType>
               </xs:element>
               <xs:element name="XMLSum" type="stml:UInt16"/>
            </xs:sequence>
         </xs:extension>
      </xs:complexContent>
   </xs:complexType>
</xs:schema>
```

D.8 Командная ЭТДП (CommandsTEDS.xsd)

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
xmlns:stml="http://grouper.ieee.org/groups/1451/0/1451HTTPAPI"
```

```
xmlns:xs="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://grouper.ieee.org/groups/1451/0/1451HTTPAPI"
elementFormDefault="qualified" attributeFormDefault="unqualified">
   <xs:include schemaLocation="SmartTransducerDataModel.xsd"/>
   <xs:include schemaLocation="TextTEDS.xsd"/>
   <xs:element name="CommandTEDS" type="stml:CommandTEDSType"
substitutionGroup="stml:TextTEDS">
      <xs:annotation>
         <xs:documentation>1451 Smart Sensor CommandTEDS
schema</xs:documentation>
      </xs:annotation>
   </xs:element>
   <xs:complexType name="CommandTEDSDataBlockType">
      <xs:sequence>
         <xs:element name="Name" type="stml:_String"/>
         <xs:element name="CmdClass" type="stml:UInt8"/>
         <xs:element name="CmdFunction" type="stml:UInt8"/>
         <xs:element name="Scope" type="stml:_String"/>
         <xs:element name="arg" minOccurs="0" maxOccurs="16535">
         <xs:annotation>
         <xs:documentation>Each arg element is one element in the argumentArray that will
be sent as part of the command.</xs:documentation>
            </xs:annotation>
            <xs:complexType>
               <xs:choice>
                  <xs:element name="TagValuePair" minOccurs="0"
maxOccurs="unbounded">
                     <xs:complexType mixed="true">
                        <xs:choice>
                           <xs:element name="Int8Value" type="stml:Int8" minOccurs="0"/>
                           <xs:element name="UInt8Value"
type="stml:UInt8" minOccurs="0"/>
                           <xs:element name="Int16Value"
type="stml:Int16" minOccurs="0"/>
                           <xs:element name="UInt16Value"
type="stml:UInt16" minOccurs="0"/>
                           <xs:element name="Int32Value"
type="stml:Int32" minOccurs="0"/>
                           <xs:element name="UInt32Value"
type="stml:UInt32" minOccurs="0"/>
                           <xs:element name="floatValue"
type="stml:Float32" minOccurs="0"/>
                           <xs:element name="doubleValue"
type="stml:Float64" minOccurs="0"/>
                           <xs:element name="timeValue" minOccurs="0">
                              <xs:complexType>
                                 <xs:simpleContent>
                                    <xs:extension base="xs:dateTime"/>
                                 </xs:simpleContent>
                              </xs:complexType>
                           </xs:element>
                           <xs:element name="booleanValue"
type="stml:_Boolean" minOccurs="0"/>
                           <xs:element name="StringArg"
type="stml:_String" minOccurs="0"/>
                        </xs:choice>
                        <xs:attribute name="Tag" type="xs:string"
use="required"/>
                     </xs:complexType>
                  </xs:element>
                  <xs:element name="listOfValues" minOccurs="0"
maxOccurs="unbounded">
```

```
                        <xs:complexType>
                           <xs:choice>
                              <xs:element name="int8ArrayValue"
type="stml:Int8Array" minOccurs="0"/>
                              <xs:element name="uInt8ArrayValue"
type="stml:UInt8Array" minOccurs="0"/>
                              <xs:element name="int16ArrayValue"
type="stml:Int16Array" minOccurs="0"/>
                              <xs:element name="uInt16ArrayValue"
type="stml:UInt16Array" minOccurs="0"/>
                              <xs:element name="int32ArrayValue"
type="stml:Int32Array" minOccurs="0"/>
                              <xs:element name="uInt32ArrayValue"
type="stml:UInt32Array" minOccurs="0"/>
                              <xs:element name="float32ArrayValue"
type="stml:Float32Array" minOccurs="0"/>
                              <xs:element name="float64ArrayValue"
type="stml:Float64Array" minOccurs="0"/>
                              <xs:element name="stringArrayValue"
type="stml:StringArray" minOccurs="0"/>
                              <xs:element name="octetArrayValue"
type="stml:OctetArray" minOccurs="0"/>
                              <xs:element name="booleanArrayValue"
type="stml:BooleanArray" minOccurs="0"/>
                              <xs:element name="timeInstanceArrayValue" type="stml:TimeInsta
nceArrayType" minOccurs="0"/>
                              <xs:element name="timeDurationArrayValue" type="stml:TimeDurat
ionArrayType" minOccurs="0"/>
                           </xs:choice>
                           <xs:attribute name="Tag" type="xs:string"
use="required"/>
                        </xs:complexType>
                     </xs:element>
                     <xs:element name="rangeOfValues" minOccurs="0">
                        <xs:complexType mixed="true">
                           <xs:choice>
                              <xs:element name="Int8Step" type="stml:Int8" minOccurs="0"/>
                              <xs:element name="UInt8Step"
type="stml:UInt8" minOccurs="0"/>
                              <xs:element name="Int16Step"
type="stml:Int16" minOccurs="0"/>
                              <xs:element name="UInt16Step"
type="stml:UInt16" minOccurs="0"/>
                              <xs:element name="Int32Step"
type="stml:Int32" minOccurs="0"/>
                              <xs:element name="UInt32Step"
type="stml:UInt32" minOccurs="0"/>
                              <xs:element name="floatStep"
type="stml:Float32" minOccurs="0"/>
                              <xs:element name="doubleStep"
type="stml:Float64" minOccurs="0"/>
                              <xs:element name="timeStep"
type="xs:dateTime" minOccurs="0"/>
                              <xs:element name="logical"
type="stml:_Boolean" minOccurs="0"/>
                           </xs:choice>
                           <xs:attribute name="Tag" type="xs:string"
use="required"/>
                        </xs:complexType>
                     </xs:element>
                  </xs:choice>
                  <xs:attribute name="argName" type="xs:string"
```

```
use="required"/>
                <xs:attribute name="argNumber" type="xs:unsignedByte" use="required"/>
                <xs:attribute name="dataModel" type="stml:DataModelTypes" use="required"/>
                <xs:attribute name="desc" type="xs:string"/>
             </xs:complexType> </xs:element>
          <xs:element name="reply" minOccurs="0" maxOccurs="65535">
             <xs:annotation>
                <xs:documentation>Each reply element is one element in the argumentArray
that will be returned as part of the reply to the command. The attribute replyArgOffset
gives the number of octets from the beginning of the reply message to the first octet in
this argument.</xs:documentation>
             </xs:annotation>
             <xs:complexType>
                <xs:annotation>
                   <xs:documentation>This field provides an offset in octets from the
beginning of the reply message to the first octet of this value.</xs:documentation>
                </xs:annotation>
                <xs:choice>
                   <xs:annotation>
                      <xs:documentation>This field provides an array of octets. The exact
structure and meaning of the elements of this array is defined in the desc attribute or by
reference to another document.</xs:documentation>
                   </xs:annotation>
                   <xs:element name="timeValue" type="xs:dateTime"/>
                   <xs:element name="BitMappedOctet">
                      <xs:annotation>
                         <xs:documentation>The mask allows a bit or bits to be selected
from the reply argument. This can be repeated as many times as required by specifying the
same replyArgOffset repeatedly. A one in a bit position in the mask selects the bit and a
zero rejects that bit.</xs:documentation>
                      </xs:annotation>
                      <xs:complexType>
                         <xs:simpleContent>
                            <xs:extension base="xs:unsignedByte">
                               <xs:attribute name="Mask"
type="stml:UInt8" use="required"/>
                            </xs:extension>
                         </xs:simpleContent>
                      </xs:complexType>
                   </xs:element>
                   <xs:element name="Int8Value" type="stml:Int8"/>
                   <xs:element name="UInt8Value" type="stml:UInt8"/>
                   <xs:element name="Int16Value" type="stml:Int16"/>
                   <xs:element name="UInt16Value" type="stml:UInt16"/>
                   <xs:element name="Int32Value" type="stml:Int32"/>
                   <xs:element name="UInt32Value" type="stml:UInt32"/>
                   <xs:element name="floatValue" type="stml:Float32"/>
                   <xs:element name="doubleValue" type="stml:Float64"/>
                   <xs:element name="textValue" type="stml:_String"/>
                   <xs:element name="octetArrayValue"
type="stml:OctetArray"/>
                   <xs:element name="logicalValue" type="stml:_Boolean"/>
                </xs:choice>
                <xs:attribute name="replyArgName" type="xs:string"
use="required"/>
                <xs:attribute name="replyArgOffset" type="xs:unsignedInt" use="required"/>
                <xs:attribute name="desc" type="xs:string"/>
             </xs:complexType>
          </xs:element>
       </xs:sequence>
    </xs:complexType>
    <xs:complexType name="ArgType">
```

```
      <xs:sequence>
         <xs:element name="DataModel">
            <xs:complexType>
               <xs:sequence>
                  <xs:element name="Type" type="xs:short"/>
                  <xs:element name="Length" type="xs:short"
default="1"/>
                  <xs:element name="Value" type="stml:UInt8"/>
               </xs:sequence>
            </xs:complexType>
         </xs:element>
         <xs:element name="description">
            <xs:complexType>
               <xs:sequence>
                  <xs:element name="Type" type="xs:short"/>
                  <xs:element name="Length" type="xs:short"/>
                  <xs:element name="Value" type="stml:_String"/>
               </xs:sequence>
            </xs:complexType>
         </xs:element>
      </xs:sequence>
   </xs:complexType>
   <xs:complexType name="CommandTEDSType">
      <xs:complexContent>
         <xs:extension base="stml:TextTEDSType">
            <xs:sequence>
               <xs:element name="XMLText"
type="stml:CommandTEDSDataBlockType"/>
               <xs:element name="XMLSum" type="stml:UInt16"/>
            </xs:sequence>
         </xs:extension>
      </xs:complexContent>
   </xs:complexType>
   <xs:simpleType name="DataModelType">
      <xs:restriction base="xs:string">
         <xs:enumeration value="Int8"/>
         <xs:enumeration value="UInt8"/>
         <xs:enumeration value="Int8Array"/>
         <xs:enumeration value="UInt8Array"/>
         <xs:enumeration value="Int16"/>
         <xs:enumeration value="UInt16"/>
         <xs:enumeration value="Int16Array"/>
         <xs:enumeration value="UInt16Array"/>
         <xs:enumeration value="Int32"/>
         <xs:enumeration value="UInt32"/>
         <xs:enumeration value="Int32Array"/>
         <xs:enumeration value="UInt32Array"/>
         <xs:enumeration value="Float32"/>
         <xs:enumeration value="Float32Array"/>
         <xs:enumeration value="Float64"/>
         <xs:enumeration value="Float64Array"/>
         <xs:enumeration value="_String"/>
         <xs:enumeration value="StringArray"/>
         <xs:enumeration value="_Octet"/>
         <xs:enumeration value="OctetArray"/>
         <xs:enumeration value="_Boolean"/>
         <xs:enumeration value="BooleanArray"/>
         <xs:enumeration value="TimeInstance"/>
         <xs:enumeration value="TimeInstanceArray"/>
         <xs:enumeration value="TimeDuration"/>
         <xs:enumeration value="TimeDurationArray"/>
      </xs:restriction>
```

```
   </xs:simpleType>
</xs:schema>
```

D.9 ЭТДП места нахождения и заголовка (LocationAndTitleTEDS.xsd)

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:stml="http://grouper.ieee.org/groups/1451/0/1451HTTPAPI"
targetNamespace="http://grouper.ieee.org/groups/1451/0/1451HTTPAPI"
elementFormDefault="qualified" attributeFormDefault="unqualified">
   <xs:include schemaLocation="SmartTransducerDataModel.xsd"/>
   <xs:include schemaLocation="TextTEDS.xsd"/>
   <xs:element name="LocationAndTitleTEDS"
type="stml:LocationAndTitleTEDSType" substitutionGroup="stml:TextTEDS">
      <xs:annotation>
         <xs:documentation>1451 Smart Sensor LocationAndTitle TEDS
schema</xs:documentation>
      </xs:annotation>
   </xs:element>
   <xs:complexType name="LocationAndTitleDataBlockType">
      <xs:sequence>
         <xs:element name="LocationAndTitle" maxOccurs="255">
            <xs:complexType>
               <xs:sequence>
                  <xs:element name="URL4TEDS" type="stml:_String"/>
                  <xs:element name="TEDSTitle" type="stml:_String"/>
               </xs:sequence>
            </xs:complexType>
         </xs:element>
      </xs:sequence>
   </xs:complexType>
   <xs:complexType name="LocationAndTitleTEDSType">
      <xs:complexContent>
         <xs:extension base="stml:TextTEDSType">
            <xs:sequence>
               <xs:element name="XMLText"
type="stml:LocationAndTitleDataBlockType"/>
               <xs:element name="XMLSum" type="stml:UInt16"/>
            </xs:sequence>
         </xs:extension>
      </xs:complexContent>
   </xs:complexType>
</xs:schema>
```

D.10 ЭТДП с расширенным набором единиц измерения (UnitsExtensionTEDS.xsd)

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:stml="http://grouper.ieee.org/groups/1451/0/1451HTTPAPI"
targetNamespace="http://grouper.ieee.org/groups/1451/0/1451HTTPAPI"
elementFormDefault="qualified" attributeFormDefault="unqualified">
   <xs:include schemaLocation="SmartTransducerDataModel.xsd"/>
   <xs:element name="UnitsExtensionDataBlock">
      <xs:annotation>
         <xs:documentation>This is the schema for the contents of the Text-based Units
Extension TEDS</xs:documentation>
      </xs:annotation>
      <xs:complexType>
         <xs:sequence>
            <xs:element name="UnitsExtensionText" type="stml:_String"/>
         </xs:sequence>
      </xs:complexType>
   </xs:element>
</xs:schema>
```

Приложение Е
(справочное)

Пример ЭТДП мета-идентификации

Е.1 Введение

Текстовые ЭТДП предоставляют изготовителю механизм встраивания текстовой информации в интеллектуальный преобразователь. Любая текстовая ЭТДП имеет структуру, которая инкапсулирует текстовую информацию как «блок данных» произвольной длины. Структура ЭТДП совмещает многочисленные блоки для поддержки содержания на разных языках. Единственным ограничением, налагаемым настоящим стандартом, является предоставление каждого блока данных в виде «образца документа» XML.

В данном приложении представлен простой пример образца документа на английском языке, подходящий для включения в ЭТДП мета-идентификации. В приложении D представлена схема для текстовой ЭТДП из данного примера. Пример, представленный ниже, не имеет цели расширения схемы.

Информационное наполнение любой текстовой ЭТДП оставлено на усмотрение изготовителя. Пример, приведенный здесь, является иллюстративной моделью для того вида информации, который изготовитель может пожелать включить в ЭТДП мета-идентификации.

Е.2 Пример схемы блока данных

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
xmlns:stml="http://grouper.ieee.org/groups/1451/0/1451HTTPAPI"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://grouper.ieee.org/groups/1451/0/1451HTTPAPI"
elementFormDefault="qualified" attributeFormDefault="unqualified">
   <xs:include schemaLocation="SmartTransducerDataModel.xsd"/>
   <xs:element name="MetaIdTEDSDataBlock">
      <xs:annotation>
          <xs:documentation>This is the schema for the contents of the data block for the
Meta Identification TEDS</xs:documentation>
      </xs:annotation>
      <xs:complexType>
         <xs:sequence>
            <xs:element name="manufacturerId" type="stml:_String"
minOccurs="0"/>
            <xs:element name="ModelNo" type="stml:_String"
minOccurs="0"/>
            <xs:element name="ProductDescription" type="stml:_String" minOccurs="0"/>
            <xs:element name="serialNo" type="stml:_String"
minOccurs="0"/>
            <xs:element name="dateCode" type="stml:_String"
minOccurs="0"/>
            <xs:element name="versionCode" type="stml:_String"
minOccurs="0"/>
         </xs:sequence>
      </xs:complexType>
   </xs:element>
</xs:schema>
```

Е.3 Пример образца документа

```
<?xml version="1.0" encoding="UTF-8"?>
< MetaIdentificationTEDSDataBlock
xmlns="http://grouper.ieee.org/groups/1451/0/1451HTTPAPI"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://grouper.ieee.org/groups/1451/0/1451HTTPAPI/Me
taIdentificationTEDSDataBlock.xsd">
   <manufacturerId>Acme Corp.</manufacturerId>
   <ModelNo>1036Z</ModelNo>
   <ProductDescription>4 Channnel Temperature TIM</ProductDescription>
```

```
   <serialNo>SNAB64</serialNo>
   <dateCode>9/05/2006</dateCode>
   <versionCode>V1.30</versionCode>
</MetaIdentificationTEDSDataBlock>
```

Все элементы данного образца документа не являются обязательными. Порядок расположения элементов схемы не ограничен. Описание (строка 4) может быть кратким или подробным.

Приложение F
(справочное)

Пример ЭТДП идентификации канала преобразователя

F.1 Введение

Текстовые ЭТДП предоставляют изготовителю механизм встраивания текстовой информации в интеллектуальный преобразователь. Любая текстовая ЭТДП имеет структуру, которая инкапсулирует текстовую информацию как «блок данных» произвольной длины. Структура ЭТДП совмещает многочисленные блоки для поддержки содержания на разных языках. Единственным ограничением, налагаемым настоящим стандартом, является предоставление каждого блока данных в виде «образца документа» XML.

В данном приложении представлен простой пример образца документа на английском языке, подходящий для включения идентификационной ЭТДП канала преобразователя. Пример, представленный ниже, не имеет цели расширения схемы.

Информационное наполнение любой текстовой ЭТДП оставлено на усмотрение изготовителя. Пример, приведенный здесь, является иллюстративной моделью для того вида информации, который изготовитель может пожелать включить в ЭТДП идентификации канала преобразователя.

F.2 Пример схемы блока данных

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
xmlns:stml="http://grouper.ieee.org/groups/1451/0/1451HTTPAPI"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://grouper.ieee.org/groups/1451/0/1451HTTPAPI"
elementFormDefault="qualified" attributeFormDefault="unqualified">
   <xs:include schemaLocation="SmartTransducerDataModel.xsd"/>
   <xs:element name="Transducer ChannelIdDataBlock">
      <xs:annotation>
         <xs:documentation>This is the schema for the contents of the data block for the
Transducer Channel Identification TEDS</xs:documentation>
      </xs:annotation>
      <xs:complexType>
         <xs:sequence>
            <xs:element name="manufacturerId" type="stml:_String"
minOccurs="0"/>
            <xs:element name="ModelNo" type="stml:_String"
minOccurs="0"/>
            <xs:element name="ChannelDescription" type="stml:_String" minOccurs="0"/>
            <xs:element name="serialNo" type="stml:_String"
minOccurs="0"/>
            <xs:element name="versionCode" type="stml:_String"
minOccurs="0"/>
         </xs:sequence>
      </xs:complexType>
   </xs:element>
</xs:schema>
```

F.3 Пример образца документа

```
<?xml version="1.0" encoding="UTF-8"?>
<Transducer ChannelIdDataBlock
xmlns="http://grouper.ieee.org/groups/1451/0/1451HTTPAPI"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://grouper.ieee.org/groups/1451/0/1451HTTPAPI/Tr
ansducerChannelIdDataBlock.xsd">
   <manufacturerId>Jones Pressure Products</manufacturerId>
   <ModelNo>1500AB</ModelNo>
   <ChannelDescription>Pressure channel</ChannelDescription>
   <serialNo>78529</serialNo>
   <versionCode>V7.00</versionCode>
</Transducer ChannelIdDataBlock>
```

Все элементы данного образца документа не являются обязательными. Порядок расположения элементов схемы не ограничен. Описание (строка 2) может быть кратким или подробным.

Приложение G
(справочное)

Пример ЭТДП идентификации калибровки

G.1 Введение

Текстовые ЭТДП предоставляют изготовителю механизм встраивания текстовой информации в интеллектуальный преобразователь. Любая текстовая ЭТДП имеет структуру, которая инкапсулирует текстовую информацию как «блок данных» произвольной длины. Структура ЭТДП совмещает многочисленные блоки для поддержки содержания на разных языках. Единственным ограничением, налагаемым настоящим стандартом, является предоставление каждого блока данных в виде «образца документа» XML.

В данном приложении представлен простой пример образца документа на английском языке, подходящий для включения идентификационной ЭТДП калибровки. В приложении D представлена схема для текстовой ЭТДП в данном примере. Пример, представленный ниже, не имеет цели расширения схемы.

В отличие от прочих текстовых ЭТДП информационное наполнение идентификационной ЭТДП калибровки определяется персоналом калибровочной лаборатории, которая может быть внешним поставщиком услуг. Тем не менее решение о включении необходимых ресурсов для поддержания данной ЭТДП остается на усмотрение изготовителя.

Представленный здесь пример является иллюстративной моделью для того вида информации, который изготовитель может пожелать включить в ЭТДП идентификации калибровки.

G.2 Пример схемы блока данных

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
xmlns:stml="http://grouper.ieee.org/groups/1451/0/1451HTTPAPI"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://grouper.ieee.org/groups/1451/0/1451HTTPAPI"
elementFormDefault="qualified" attributeFormDefault="unqualified">
   <xs:include schemaLocation="SmartTransducerDataModel.xsd"/>
   <xs:element name="CalibrationIdentificationTEDSDataBlock">
      <xs:annotation>
          <xs:documentation>This is the schema for the contents of the data block for the
Calibration Identification TEDS</xs:documentation>
      </xs:annotation>
      <xs:complexType>
         <xs:sequence>
            <xs:element name="CalLabNo" type="stml:_String"
minOccurs="0"/>
            <xs:element name="CalTech" type="stml:_String"
minOccurs="0"/>
            <xs:element name="CalibrationlDescription"
type="stml:_String" minOccurs="0"/>
            <xs:element name="CalDate" type="stml:_String"
minOccurs="0"/>
            <xs:element name="CalProc" type="stml:_String"
minOccurs="0"/>
            <xs:element name="CalStd" type="stml:_String"
minOccurs="0"/>
         </xs:sequence>
      </xs:complexType>
   </xs:element>
</xs:schema>
```

G.3 Пример образца документа

```
<?xml version="1.0" encoding="UTF-8"?>
<CalibrationIdentificationTEDSDataBlock
xmlns="http://grouper.ieee.org/groups/1451/0/1451HTTPAPI"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://grouper.ieee.org/groups/1451/0/1451HTTPAPI/Ca
```

```
librationIdentificationTEDSDataBlock.xsd">
   <CalLabNo>Acurate Calibrations Inc</CalLabNo>
   <CalTech>C.P. Lee</CalTech>
   <CalibrationlDescription>I had difficulty setting the gain on IA 3. Watch for excessive
drift on next cal cycle</CalibrationlDescription>
   <CalDate>08/01/2002</CalDate>
   <CalProc>Calibrated per the procedures in D6-46289</CalProc>
   <CalStd>Ektron 8200, SN 24698</CalStd>
</CalibrationIdentificationTEDSDataBlock>
```

Приложение Н
(справочное)

Пример командной ЭТДП

Н.1 Введение

Текстовые ЭТДП предоставляют изготовителю механизм встраивания текстовой информации в интеллектуальный преобразователь. Любая текстовая ЭТДП имеет структуру, которая инкапсулирует текстовую информацию как «блок данных» произвольной длины. Структура ЭТДП совмещает многочисленные блоки для поддержки содержания на разных языках. Единственным ограничением, налагаемым настоящим стандартом, является предоставление каждого блока данных в виде «образца документа» XML.

В данном приложении представлен простой пример образца документа на английском языке, подходящий для включения командной ЭТДП. В приложении D представлена схема для текстовой ЭТДП в данном примере. Пример, представленный ниже, не имеет цели расширения схемы.

Информационное наполнение любой текстовой ЭТДП оставлено на усмотрение изготовителя. Представленный здесь пример является иллюстративной моделью для того вида информации, который изготовитель может пожелать включить в командную ЭТДП. Соответствие данным рекомендациям дает изготовителю преимущество использования обобщенных интерактивных инструментов, которые, как ожидается, будут развиваться по мере работы. Пример подобного инструмента приведен ниже. Несоответствие данным рекомендациям потребует от изготовителя разработки собственного набора пользовательских инструментов для запуска команд, определенных изготовителем.

Н.2 Пример ситуации

Предположим, что изготовитель ИМП имеет собственное производство блоков формирования сигналов, отличающее данный продукт от всех конкурентов. Фронтальная часть таких блоков имеет многочисленные ступени усиления с программируемыми коэффициентами усиления и смещениями (поправками). Для поддержки функции калибровки и решения возникающих проблем поставщик желает определить следующие команды:

SetGain	Set gain, PreAmp	(Установить коэффициент усиления)
AlternateSetGain	Set gain, PreAmp	(Установить альтернативный коэффициент усиления)
SetOffset1	Set offset, PreAmp	(Установить смещение)

Данный искусственный пример имеет перечень намеренно сокращенных команд, но может быть легко расширен для охвата «многочисленных ступеней усиления», упомянутых выше.

Описанный выше сценарий был специально выбран, чтобы проиллюстрировать факт того, что определенные изготовителем команды не планируются к использованию в рабочем режиме преобразователя. Такие команды служат скорее инструментами решения проблем калибровки или пусконаладки.

Н.3 Схема блока данных

```
<?xml version="1.0" encoding="UTF-8"?>
<mdCmd xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
<xs:schema
xmlns:stml="http://grouper.ieee.org/groups/1451/0/1451HTTPAPI"
targetNamesace="http://grouper.ieee.org/groups/1451/0/1451HTTPAPI"
elementFormDefault="qualified" attributeFormDefault="unqualified">
xsi:noNamespaceSchemaLocation="http://grouper.ieee.org/groups/1451/0/1451HTTPAPI/
IEEE1451CommandsTEDSSchema.xsd">
   <name desc="This command is a manufacturer unique command and is used set the gain of
an amplifier.">setGain</name>
   <CmdClass>128</CmdClass>
   <CmdFunction>2</CmdFunction>
  <arg argName="GainValue" argNumber="0" dataModel="Int8" desc="This amplifier has allowable
gains of 1, 2, 4, 8, 16 or 32. Select the value for the gain that you want to use and enter
it into the argument array for the command.">
      <TagValuePair Tag="gain=1">
         <Int8Value>0</Int8Value>
      </TagValuePair>
      <TagValuePair Tag="gain=2">
         <Int8Value>1</Int8Value>
```

```
      </TagValuePair>
      <TagValuePair Tag="gain=4">
         <Int8Value>2</Int8Value>
      </TagValuePair>
      <TagValuePair Tag="gain=8">
         <Int8Value>3</Int8Value>
      </TagValuePair>
      <TagValuePair Tag="gain=16">
         <Int8Value>4</Int8Value>
      </TagValuePair>
      <TagValuePair Tag="gain=32">
         <Int8Value>5</Int8Value>
      </TagValuePair>
   </arg>
   <reply replyArgName="Success/Fail" desc="The reply to this command should contain the
Success Fail flag and a length of one" replyArgOffset="0">
      <Int8Value>0</Int8Value>
   </reply>
</mdCmd>
```

Н.4 Пример образцов документа

```
<?xml version="1.0" encoding="UTF-8"?>
<mdCmd xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
<xs:schema
xmlns:stml="http://grouper.ieee.org/groups/1451/0/1451HTTPAPI"
targetNamespace="http://grouper.ieee.org/groups/1451/0/1451HTTPAPI"
elementFormDefault="qualified" attributeFormDefault="unqualified">
xsi:noNamespaceSchemaLocation="http://grouper.ieee.org/groups/1451/0/1451HTTPAPI/
IEEE1451CommandsTEDSSchema.xsd">
   <name desc="This is a manufacturer unique command is used to set gain from a list of
allowable gains.">SetGain</name>
   <CmdClass>128</CmdClass>
   <CmdFunction>2</CmdFunction>
    <arg argName="Select Gain" argNumber="0" dataModel="Int8" desc="This command allows
setting gains from a list of allowable gains. Decide what gain you want and enter a value
from the list in the argument for the command">
      <listOfValues Tag="List of Gain Values">
         <UInt8Array>1,2,4,8,16,32</UInt8Array>
      </listOfValues>
   </arg>
   <reply replyArgName="Success/Fail" desc="The reply to this command should contain the
Success Fail flag and a length of one"
replyArgOffset="0">
      <Int8Value>0</Int8Value>
   </reply>
</mdCmd>

 <?xml version="1.0" encoding="UTF-8"?>
<mdCmd xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
<xs:schema xmlns:stml=http://grouper.ieee.org/groups/1451/0/1451HTTPAPI
targetNamespace="http://grouper.ieee.org/groups/1451/0/1451HTTPAPI"
elementFormDefault="qualified" attributeFormDefault="unqualified">
xsi:noNamespaceSchemaLocation="http://grouper.ieee.org/groups/1451/0/1451HTTPAPI/
IEEE1451CommandsTEDSSchema.xsd">
   <name desc="This is a manufacturer unique command is used to set an offset in a signal
conditioner.">SetOffset</name>
   <CmdClass>128</CmdClass>
   <CmdFunction>3</CmdFunction>
   <arg argName="Tag" argNumber="0" dataModel="Int8" desc="This command allows setting an
offset between -5v and + 5 V in 250 steps of 0.04 volts each. Decide what offset you want.
Divide it by 0.04 and enter that for the value in the argument for the command">
```

```
      <rangeOfValues Tag="Range of Offset Values">
         <Int8Step hiRange="125" loRange="-125"
stepSize="1">0</Int8Step>
      </rangeOfValues>
   </arg>
   <reply replyArgName="Success/Fail" desc="The reply to this command should contain the
Success Fail flag and a length of one"
replyArgOffset="0">
      <Int8Value>0</Int8Value>
   </reply>
</mdCmd>
```

Таким образом, может быть разработан инструмент многофункционального программного обеспечения для отображения набора выполняемых команд с запросом их выбора у оператора. На основе выбранной команды инструмент может запрашивать у оператора каждый аргумент. При накоплении информации инструмент может сконструировать соответствующую командную строку, передать команду устройству, считать ее и правильно отобразить результат(ы).

Приложение I
(справочное)

Пример ЭТДП места нахождения и заголовка

I.1 Введение

Текстовые ЭТДП предоставляют изготовителю механизм встраивания текстовой информации в интеллектуальный преобразователь. Любая текстовая ЭТДП имеет структуру, которая инкапсулирует текстовую информацию как «блок данных» произвольной длины. Структура ЭТДП совмещает многочисленные блоки для поддержки содержания на разных языках. Единственным ограничением, налагаемым настоящим стандартом, является предоставление каждого блока данных в виде «образца документа» XML.

В данном приложении представлен простой пример образца документа на английском языке, подходящий для включения ЭТДП места нахождения и заголовка. В приложении D представлена схема для текстовой ЭТДП из данного примера.

Информационное наполнение любой текстовой ЭТДП оставлено на усмотрение изготовителя. Представленный здесь пример является иллюстративной моделью для того вида информации, который изготовитель может пожелать включить в ЭТДП места нахождения и заголовка.

I.2 Пример ситуации

Предположим, что компания — изготовитель интеллектуальных преобразователей произвела ИМП модели М2260 с обработкой сигнала, получаемого на фиксированной частоте 5 кГц. Предположим также, что данное устройство имеет цифровой фильтр децимации (DDF) (с кратным 10 уменьшением частоты), обеспечивающий эффективный прием данных на частотах 1 кГц, 100, 10 и 1 Гц. Изготовитель понимает, что для продвижения его продукции (то есть каналов преобразователя) покупателю, имеющему потребность в наличии альтернативных частот приема данных, например 10 Гц вместо 100 Гц, проблема решается простой заменой коэффициентов цифрового фильтра. Таким образом, для обеспечения такого механизма замены коэффициентов применяются четыре ЭТДП, определенные изготовителем.

I.3 Обязательная схема для блока данных

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
xmlns:stml="http://grouper.ieee.org/groups/1451/0/1451HTTPAPI"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://grouper.ieee.org/groups/1451/0/1451
HTTPAPI"
elementFormDefault="qualified" attributeFormDefault="unqualified">
   <xs:include schemaLocation="SmartTransducerDataModel.xsd"/>
   <xs:element name="LocationAndTitleDataBlock">
      <xs:annotation>
          <xs:documentation>This is the schema for the contents of the data block for the
LocationAndTitleTEDS</xs:documentation>
      </xs:annotation>
      <xs:complexType>
         <xs:sequence>
            <xs:element name="URL4TEDS" type="stml:_String"
minOccurs="0"/>
            <xs:element name="TEDSTitle" maxOccurs="255">
               <xs:complexType>
                  <xs:simpleContent>
                     <xs:extension base="stml:UInt8">
                        <xs:attribute name="TEDSAccessCode"/>
                     </xs:extension>
                  </xs:simpleContent>
               </xs:complexType>
            </xs:element>
         </xs:sequence>
      </xs:complexType>
   </xs:element>
</xs:schema>
```

I.4 Пример образца документа

```
<?xml version="1.0" encoding="UTF-8"?>
<LocationAndTitleDataBlock
xmlns="http://grouper.ieee.org/groups/1451/0/1451HTTPAPI"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://grouper.ieee.org/groups/1451/0/1451HTTPAPI/
LocationAndTitleTEDS.xsd">
   <URL4TEDS>http://www.AcmeCorp.com/1036Z/TEDS</URL4TEDS>
   <TEDSTitle TEDSAccessCode="129">5kHzDigitalFilterTEDS</TEDSTitle>
   <TEDSTitle TEDSAccessCode="130">1kHzDigitalFilterTEDS</TEDSTitle>
   <TEDSTitle TEDSAccessCode="131">100HzDigitalFilterTEDS</TEDSTitle>
   <TEDSTitle TEDSAccessCode="132">10HzDigitalFilterTEDS</TEDSTitle>
   <TEDSTitle TEDSAccessCode="133">1HzDigitalFilterTEDS</TEDSTitle>
</LocationAndTitleDataBlock>
```

Элемент «URL4TEDS» сообщает место нахождения сети в форме URL, где пользователь может найти электронные копии всех виртуальных ЭТДП, если таковые имеются.

Элемент «TEDSTitle» предоставляет смысловой заголовок для ЭТДП. Он имеет единственный аргумент, который определяет коды доступа к данной ЭТДП. Данный элемент повторяется пять раз, по одному разу для каждой ЭТДП.

Приложение J
(справочное)

Пример ЭТДП с расширенным набором единиц измерения

J.1 Введение

Как показано в 5.5.2.8, существуют условия, когда для понимания того, измерения каких величин проводятся, необходимо добавить дополнительную информацию к набору физических единиц измерения. Данная ЭТДП предоставлена для размещения такой информации.

J.2 Пример ситуации

Преобразователь разработан для измерения количества хлора в молях. Физические единицы измерения адекватно отражают единицы в молях, но не сообщают, к какому веществу они относятся. Данный пример ЭТДП предоставляет текстовую информацию, отражающую такие единицы измерения, как моли хлора.

J.3 Пример схемы для блока данных

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:stml="http://grouper.ieee.org/groups/1451/0/1451HTTPAPI"
targetNamespace="http://grouper.ieee.org/groups/1451/0/ 1451HTTPAPI"
elementFormDefault="qualified" attributeFormDefault="unqualified">
   <xs:include schemaLocation="SmartTransducerDataModel.xsd"/>
   <xs:element name="UnitsExtensionDataBlock">
      <xs:annotation>
          <xs:documentation>This is the schema for the contents of the Text-based Units
Extension TEDS</xs:documentation>
      </xs:annotation>
      <xs:complexType>
         <xs:sequence>
            <xs:element name="UnitsExtensionText" type="stml:_String"/>
         </xs:sequence>
      </xs:complexType>
   </xs:element>
</xs:schema>
```

J.4 Пример образца документа

```
<?xml version="1.0" encoding="UTF-8"?>
<UnitsExtensionDataBlock
xmlns="http://grouper.ieee.org/groups/1451/0/1451HTTPAPI"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://grouper.ieee.org/groups/1451/0/1451HTTPAPIUnitsExtensionDataBl
ock.xsd">
   <UnitsExtensionText>of Chlorine</UnitsExtensionText>
</UnitsExtensionDataBlock>
```

Приложение К
(справочное)

Примеры физических единиц

Рабочая группа ИИЭР 1451.2 разработала простой и удобный для хранения метод определения физических единиц во время работы интеллектуальных преобразователей, позволяющий преобразователям предоставить на выходе информацию, понятную любому пользователю. Настоящий стандарт принял данный метод к сведению и дополнил его. Данный метод крайне необходим, если преобразователь должен быть подключен в любую систему, при этом оставаясь работоспособным без написания дополнительного программного обеспечения. Выбранный подход содержит компромиссы, но он применим практически в любом приложении.

Для физических единиц модуля преобразователя существуют два назначения. Первая задача заключается в определении величин, измерение которых проводит датчик, или тех величин, которые подаются на выход исполнительного устройства. При считывании ЭТДП канала преобразователя можно определить, что ИМП разработан для измерения давления, ускорения и т. д. Вторая задача заключается в идентификации масштаба величин, связанного с соответствующим входом или выходом ИМП, что позволит изготовителю определить калибровочные константы в ЭТДП калибровки и иметь уже известные масштабы. Тем не менее данная изготовителю возможность назначать калибровочные константы в устройстве не позволяет преобразователю работать в любой системе без разработки дополнительного программного обеспечения. Например, какие единицы следует использовать для калибровки преобразователя для измерения давления? Это могут быть паскали, килопаскали, или мегапаскали. Кроме паскалей, давление может измеряться в «psi» («фунты на квадратный дюйм») или в миллиметрах ртутного столба. Возможно, у пользователя возникнет необходимость использования миллиметров водного или даже спиртового столба. Все эти единицы регулярно применяются для измерения давления, таким образом, список возможных единиц измерения, нужных пользователю, бесконечен. Чтобы дать изготовителю возможность произвести преобразователь со встроенными калибровочными константами, должны быть определены физические единицы измерения. В стандарте ИИЭР 1451.2—1997 данные две функции были объединены в поле единиц измерения ЭТДП канала преобразователя. В настоящем стандарте также используется только одно поле единиц измерения ЭТДП канала преобразователя, но введены дополнительные поля в ЭТДП калибровки, позволяющие за счет калибровочных констант предоставлять различный набор величин. Описание данных полей представлено в конце данного приложения в пункте «Системные ограничения».

Существуют два основных подхода для определения единиц измерения, которые могли быть использованы рабочей группой. Первый вариант предполагает попытку представления таблицы всех возможных единиц измерения с использованием нумерации для выбора тех, которые используются преобразователем. Трудность данного подхода заключается в том, что число всех возможных единиц измерения практически не ограничено. Таблица может быть слишком большой и все равно не включать все возможные единицы измерения. Второй подход заключается в определении небольшого набора основных единиц измерения, которые могут быть скомбинированы для получения требуемых единиц. Таким образом, задача стандарта заключается в нахождении способа комбинирования таких основных единиц для получения единиц измерения для определенного преобразователя. Не все единицы измерения, которые могут оказаться необходимыми пользователю, могут быть получены таким путем, но может быть создан набор, адекватный для большинства преобразователей. Преобразование единиц, оговоренных в настоящем стандарте, в единицы, необходимые пользователю, обычно достигается простой операцией умножения, а системе остается только отобразить данные.

Задача рабочей группы заключалась в определении постоянного набора единиц измерения и нахождении практического пути для их встраивания в преобразователь. Другие стандарты также рассматривают вопрос определения постоянного набора единиц измерения. Подходящий набор выбран рабочей группой ИИЭР 1451.2, ссылка на него размещена в стандарте. Данный набор представлен в Международной системе единиц (СИ), установленной в 1960 г. на XI Генеральной конференции по мерам и весам. Сокращение СИ (SI) происходит от французского наименования «Le System International d'Units», которое обычно используется для идентификации данного набора единиц измерения. Данный набор основан на Международной метрической системе и используется во всем мире.

В таблице К.1 представлены семь основных взаимно независимых величин, на которых основана система СИ, с наименованиями и обозначениями их единиц измерения в системе СИ — основных единиц СИ. Определения основных единиц СИ приведены в документе «Международная система единиц (СИ)» [В9][1)].

Таблица К.1 — Основные единицы измерения системы СИ

Основная величина	Единица измерения	Обозначение
Длина	метр	м (m)

[1)] В оригинале ISO/IEC/IEEE 21450:2010 приведена ошибочная ссылка на [В7].

Окончание таблицы К.1

Основная величина	Единица измерения	Обозначение
Масса	килограмм	кг (kg)
Время	секунда	с (s)
Электрический ток	ампер	А (A)
Термодинамическая температура	кельвин	К (K)
Количество вещества	моль	моль (mol)
Сила света	кандела	кд (cd)

Производные единицы измерения выражаются алгебраически через основные единицы. Некоторые единицы в графе «Выражение через другие единицы СИ» из таблицы К.2 выражены через другие производные единицы. Тем не менее, до того как они будут представлены в соответствии с настоящим стандартом, они должны быть преобразованы в основные единицы измерения, как показано в графе «Выражение через основные единицы СИ» таблицы К.2. Обозначения для производных единиц измерения получены посредством математических операций умножения и деления. Например, производная единица молярной массы (масса, деленная на количество вещества) — килограмм на моль и обозначается как «кг/моль». Таблица К.2 содержит примеры производных единиц измерения, которые выражены через основные единицы измерения системы СИ.

Рабочая группа решила включить основные единицы измерения системы СИ, а также радианы и стерадианы в качестве основных единиц измерения настоящего стандарта. Таким образом, получилось девять единиц измерения, из которых путем умножения и деления могут быть получены остальные производные физические единицы измерения, необходимые для преобразователя. При этом остается лишь порекомендовать способ введения информации в преобразователь.

На основании данных таблицы К.2 можно сделать вывод, что любая конкретная единица измерения может быть представлена как произведение набора основных единиц измерения, каждая из которых возведена в соответствующую степень. Можно заметить, что определенная единица измерения может быть представлена через все основные единицы измерения, возведенные в определенную степень, включая степень 0. Например, для датчика, измеряющего расстояние, пользователь может задать единицы измерения через семь основных единиц измерения следующим способом: $м^1 \cdot кг^0 \cdot с^0 \cdot А^0 \cdot К^0 \cdot моль^0 \cdot кд^0$.

Тем не менее единицы измерения редко записываются со степенью «0», а в случае степени «1» запись значения степени не делается. Единицы измерения расстояния измерительного устройства просто записываются как «м». Но при использовании компьютеров необходимы более четкие инструкции, такие как таблицы, содержащие все степени, записанные в определенном порядке. Для примера, приведенного выше, запись будет иметь вид: 1, 0, 0, 0, 0, 0, 0, что означает $м^1 \cdot кг^0 \cdot с^0 \cdot А^0 \cdot К^0 \cdot моль^0 \cdot кд^0$.

Все это составляет основу метода представления физических единиц, используемого в настоящем стандарте. Рабочая группа решила добавить две производные единицы, радианы и стерадианы, к семи основным единицам измерения системы СИ, таким образом, общее число единиц составляет девять. Тем не менее в некоторых приборах измеряется отношение двух значений с одинаковыми единицами измерения. Например, деформация измеряется в метрах на метр и выражается как м/м, $м^1 \cdot м^{-1}$ или $м^0$. Подобные результаты измерений называются «безразмерными». Несмотря на это, по-прежнему необходимо получить информацию о том, в каких физических единицах устройство проводит измерения. Такая ситуация требует особой работы с системой единиц измерения. Также значения могут измеряться логарифмом какой-либо величины или логарифмом «безразмерного» отношения. Для всех вышеперечисленных случаев требуется метод идентификации значений для возможности их интерпретации. Существуют две категории значений, которые не выражаются через основные единицы измерения системы СИ, это произвольные единицы, такие как твердость (жесткость) и «цифровые данные». Система должна быть способна распознавать такие величины. Данная проблема может быть решена при помощи идентификатора классов единиц измерения.

Таблица К.2 — Производные от основных единиц измерения системы СИ

Величина	Наименование единицы	Обозначение	Выражение через другие единицы СИ	Выражение через основные единицы СИ
Плоский угол	радиан	рад		$м \cdot м^{-1} = 1$
Телесный угол	стерадиан	ср		$м^2 \cdot м^{-2} = 1$

Окончание таблицы К.2

Величина	Наименование единицы	Обозначение	Выражение через другие единицы СИ	Выражение через основные единицы СИ
Частота	герц	Гц		$с^{-1}$
Площадь	квадратный метр			$м^2$
Объем	кубический метр			$м^3$
Ускорение	метр на секунду в квадрате			$м/с^2$
Волновое число	метр в минус первой степени			$м^{-1}$
Плотность	килограмм на кубический метр			$кг/м^3$
Удельный объем	кубический метр на килограмм			$м^3/кг$
Плотность электрического тока	ампер на квадратный метр			$А/м^2$
Напряженность магнитного поля	ампер на метр			А/м
Молярная концентрация компонента	моль на кубический метр			$моль/м^3$
Яркость	кандела на квадратный метр			$кд/м^2$
Сила	ньютон	Н		$м \cdot кг \cdot с^{-2}$
Давление, механическое напряжение	паскаль	Па	$Н/м^2$	$м^{-1} \cdot кг \cdot с^{-2}$
Энергия, работа, количество теплоты	джоуль	Дж	$Н \cdot м$	$м^2 \cdot кг \cdot с^{-2}$
Мощность, поток энергии	ватт	Вт	Дж/с	$м^2 \cdot кг \cdot с^{-3}$
Электрический заряд, количество электричества	кулон	Кл		$с \cdot А$
Электрическое напряжение, электрический потенциал, разность электрических потенциалов, электродвижущая сила	вольт	В	Вт/А	$м^2 \cdot кг \cdot с^{-3} \cdot А^{-1}$
Электрическая емкость	фарад	Ф	Кл/В	$м^{-2} \cdot кг^{-1} \cdot с^4 \cdot А^2$
Электрическое сопротивление	ом	Ом	В/А	$м^2 \cdot кг \cdot с^{-3} \cdot А^{-2}$
Электрическая проводимость	сименс	См	А/В	$м^{-2} \cdot кг^{-1} \cdot с^3 \cdot А^2$
Магнитный поток	вебер	Вб	$В \cdot с$	$м^2 \cdot кг \cdot с^{-2} \cdot А^{-1}$
Магнитная индукция	тесла	Тл	$Вб/м^2$	$кг \cdot с^{-2} \cdot А^{-1}$
Индуктивность	генри	Гн	Вб/А	$м2 \cdot кг \cdot с^{-2} \cdot А^{-2}$
Температура Цельсия	градус Цельсия	°С		К
Световой поток	люмен	лм		$кд \cdot ср$
Освещенность	люкс	лк	$лм/м^2$	$м^{-2} \cdot кд \cdot ср$

Теоретически степени могут принимать любые значения и должны быть логически представлены числами с плавающей точкой, но на практике их значения лежат в пределах ±4 с приемлемым шагом 1/2. Также необходимо отметить два момента, связанных со сложностями описания единиц измерения. Первый из них заключается в том, что наименьшая единица в стандарте ИИЭР 1451.2—1997 является байтом. Второй момент связан с тем, что в стандарте ИИЭР 1451.2—1997 нет определенного способа выражения целого числа со знаком. Так как остальные части данного стандарта не требуют работы с целыми числами со знаком, то степени закодированы с использованием 8-разрядного целого числа без знака. Чтобы получить шаг 1/2, степень умножается на два. Чтобы степень была представлена целым числом со знаком, к степени добавляется 128 после умножения на два. Таким образом, могут быть представлены значения степени от минус 64 до плюс 63. В целях сохранения соответствия со стандартами ИИЭР 1451.2—1997 и ИИЭР 1451.3—2003 данная форма представления информации в настоящем стандарте не меняется.

К.1 Примеры

В таблице К.3 приведены поля для единиц измерения расстояния, выраженного в метрах. Значение «0» нумерации указывает на то, что единицы измерения являются произведением основных единиц измерения. Так как метр является основной единицей измерения, показатель степени для метра не равен нулю.

Таблица К.3 — Расстояние (м)

	Нумерация	рад	ср	м	кг	с	А	К	моль	кд
Степень	0	0	0	1	0	0	0	0	0	0
Двоичное число		128	128	130	128	128	128	128	128	128

В таблице К.4 приведены поля для единиц измерения площади. Значение «0» нумерации указывает на то, что единицы измерения являются произведением основных единиц измерения. Так как единица измерения основана на метре, который является основной единицей измерения, показатель степени для метра не равен нулю. Так как единицей измерения площади является квадратный метр, то есть м· м, то показатель степени для метра равен 2.

Таблица К.4 — Площадь ($м^2$)

	Нумерация	рад	ср	м	кг	с	А	К	моль	кд
Степень	0	0	0	2	0	0	0	0	0	0
Двоичное число		128	128	132	128	128	128	128	128	128

В таблице К.5 приведены поля для единиц измерения давления. Производная единица измерения давления — паскаль. Она содержат три основные единицы: метр, килограмм и секунда.

Таблица К.5 — Давление ($Па = м^{-1} \cdot кг \cdot с^{-2}$)

	Нумерация	рад	ср	м	кг	с	А	К	моль	кд
Степень	0	0	0	–1	1	–2	0	0	0	0
Двоичное число		128	128	126	130	124	128	128	128	128

Поля для единиц электрического сопротивления представлены в таблице К.6. Как показано в данной таблице, единица измерения электрического сопротивления включает в себя четыре основных единицы измерения.

Таблица К.6 — Сопротивление ($Ом = м^2 \cdot кг \cdot с^{-3} \cdot А^{-2}$)

	Нумерация	рад	ср	м	кг	с	А	К	моль	кд
Степень	0	0	0	2	1	–3	–2	0	0	0
Двоичное число		128	128	132	130	122	124	128	128	128

В таблице К.7 представлены поля для единиц измерения спектральной плотности шума, которая также содержит четыре основные единицы измерения. В данном примере есть одна интересная особенность, а именно использование функции квадратного корня, что дает степень минус 5/2.

Таблица К.7 — Спектральная плотность шума — вольт на квадратный корень герц ($В/\sqrt{Гц} = м^2 \cdot кг \cdot с^{-5/2} \cdot А^{-1}$)

	Нумерация	рад	ср	м	кг	с	А	К	моль	кд
Степень	0	0	0	2	1	–5/2	–1	0	0	0
Двоичное число		128	128	132	130	123	126	128	128	128

В таблице К.8 описывается массовая доля. Массовая доля является «безразмерной» величиной, представленной единицей моль на моль. Таким образом, для обозначения отношения величин одинаковой размерности используется нумерация «1».

Таблица К.8 — Массовая доля (моль/моль)

	Нумерация	рад	ср	м	кг	с	А	К	моль	кд
Степень	1	0	0	0	0	0	0	0	1	0
Двоичное число		128	128	128	128	128	128	128	130	128

В таблице К.9 описываются единицы деформации. Деформация является «безразмерной» величиной, представленной единицей метр на метр. Таким образом, для обозначения отношения величин одинаковой размерности используется нумерация «1». Оставшиеся поля заполняются так же, как и для измерения расстояния, приведенного в таблице К.3.

Таблица К.9 — Деформация (м/м)

	Нумерация	рад	ср	м	кг	с	А	К	моль	кд
Степень	1	0	0	1	0	0	0	0	0	0
Двоичное число		128	128	130	128	128	128	128	128	128

Измерение мощности излучения, как правило, проводится в децибелах. В таблице К.10 представлены поля для данной величины с использованием основных единиц, содержащихся в единице бел, равной десяти децибелам.

Таблица К.10 — Мощность излучения [$Б = \log_{10}(Вт/Вт)$; $Вт = м^2 \cdot кг \cdot с^{-3}$]

	Нумерация	рад	ср	м	кг	с	А	К	моль	кд
Степень	3	0	0	2	1	–3	0	0	0	0
Двоичное число		128	128	132	130	122	128	128	128	128

Предположим, что преобразователь ведет подсчет изделий в момент прохождения ими отметки на конвейерной ленте. Подходящая единица измерения для такого преобразователя — «штуки». Но такую единицу измерения невозможно сформировать из девяти основных единиц измерения, определенных в настоящем стандарте. Для идентификации нестандартной единицы измерения в настоящем стандарте предусмотрена нумерация «0». Все степени будут установлены на «0», как показано в таблице К.11. Следует отметить, что для отображения подобных единиц измерения потребуется специальное программное обеспечение.

Таблица К.11 — Подсчет единиц

	Нумерация	рад	ср	м	кг	с	А	К	моль	кд
Степень	0	0	0	0	0	0	0	0	0	0
Двоичное число		128	128	128	128	128	128	128	128	128

Выход преобразователя может представлять настройки блока переключателей или команду для открытия или закрытия клапана. Единицами измерения могут быть, например, следующие: «вкл» (включено), «выкл» (выключено), «вверх», «закрыто», а также любые другие. Такие единицы измерения невозможно сформировать из девяти основных единиц измерения, определенных в настоящем стандарте. Так как такие величины не являются

количественными, то в данном случае для поля нумерации следует задавать значение «4». В таблице К.12 приведен пример единиц для преобразователя такого типа. Для управления преобразователем такого типа в системе пользователя требуется использование специального программного обеспечения.

Таблица К.12 — Положения переключателя

	Нумерация	рад	ср	м	кг	с	А	К	моль	кд
Степень	4	0	0	0	0	0	0	0	0	0
Двоичное число		128	128	128	128	128	128	128	128	128

К.2 Системные ограничения

Все указанные выше ситуации описывают способы идентификации измеряемых физических величин или значений на выходе преобразователя, но зачастую не предоставляют величины, необходимые для отображения оператору. Настоящий стандарт позволяет задавать константы преобразования в ЭТДП калибровки для любого набора величин, необходимых оператору. В результате возникает вопрос, как компьютер может определить, какие из величин ЭТДП калибровки используются? Для решения данной проблемы в ЭТДП калибровки добавлены две константы: «SI units conversion slope» («Коэффициент преобразования единиц СИ») и «SI units conversion intercept» («Сдвиг (смещение) преобразования единиц СИ»). Данные константы необходимы для проведения процесса преобразования и корректировки значений на выходе, с применением констант ЭТДП калибровки, в единицы СИ. В результате процесс определения используемых единиц становится двухступенчатым. На первой ступени необходимо определить, что именно измеряется или отрабатывается на выходе. Данная задача может быть решена на основе представленного выше анализа физических единиц измерения в ЭТДП канала преобразователя. Следующий шаг заключается в рассмотрении постоянной «SI units conversion slope» преобразования единиц СИ. Для всех величин, кроме температуры, значение данной постоянной будет различным в зависимости от набора единиц. Например, рассмотрим возможные единицы измерения и значения постоянной при измерении давления (см. таблицу К.13). Для измерения давления используется множество наборов единиц. Единица измерения давления в системе СИ выражается как $м^{-1}·кг·с^{-2}$ и называется паскалем. Проверка констант в графе «Коэффициент преобразования» показывает, что для каждой отдельной единицы измерения существует своя константа, которая может быть использована для определения единиц, получаемых в результате процесса коррекции. При использовании данной таблицы для произвольного набора единиц измерения может возникнуть следующая проблема. В некоторых случаях одна и та же единица может быть названа двумя различными именами, например, «миллиметр ртутного столба» и «торр». Настоящий стандарт не позволяет системе однозначно определить, какое из двух наименований необходимо использовать. Данный вопрос остается на усмотрение пользователя.

Таблица К.13 — Константы преобразования для давления

Единица измерения	Коэффициент преобразования
паскаль	1
килопаскаль	0,001
фунт на квадратный дюйм (PSI)	6894,757
бар	100 000
миллибар	100
дюйм водяного столба	249,082
миллиметр водяного столба	9,80665
дюйм ртутного столба	3386,38
миллиметр ртутного столба	133,3224
атмосфера	101 325
килограмм на квадратный сантиметр	98066,5
торр	133,3224

Примечание — Некоторые из данных констант зависят от температуры и будут верны только для заданной температуры.

Как было отмечено, такой метод работает для всех единиц, кроме температуры. Проблема становится очевидной при анализе данных, приведенных в таблице К.14. Единицы кельвин и градус Цельсия имеют одинаковые коэффициенты преобразования. Аналогичные рассуждения верны для градуса Фаренгейта и градуса Ранкина. Для определения единиц измерения температуры, измеряемой или отрабатываемой на выходе, требуются как коэффициент преобразования, так и сдвиг (смещение) преобразования.

Таблица К.14 — Преобразование констант для температуры

Единица	Коэффициент преобразования	Сдвиг преобразования
кельвин	1	0
градус Цельсия	1	273,15
градус Фаренгейта	0,5555556	255,3722
градус Ранкина	0,5555556	0

К.3 Выводы

Настоящий стандарт предлагает согласованный набор инструментов, позволяющий решить проблемы отображения данных стандартизованным способом. Предоставляя способ встраивания набора единиц измерения в преобразователь, рабочая группа обеспечивает стандартный способ представления единиц измерения в ЭТДП, а также стандартный способ представления коэффициентов калибровки. Это означает, что преобразователь, изготовленный и откалиброванный в соответствии с настоящим стандартом, должен быть способен подключаться к любой системе. Система, в свою очередь, должна быть способна отображать большинство данных без каких-либо изменений в ней.

К.4 Благодарность

Данный метод представления физических единиц измерения в вычислительной системе разработан Брюсом Гамильтоном с помощью Уильяма Кента, Стефани Яновски и Дана Хепнера — сотрудниками лаборатории Hewlett Packard в Palo Alto в Калифорнии. Гамильтоном написана работа под названием «Компактное представление физических единиц» («A Compact Representation of Physical Units»), в которой изложена теория и представлен список литературы для более детального изучения. Работа доступна в отчете «Agilent Laboratories Technical Report HPL-96-61», 1995 г., в отделе «Aligent Technical Publications Department » [B1].

Приложение L
(справочное)

Протоколы считывания и записи ЭТДП

L.1 Доступ к ЭТДП

Структуры данных ЭТДП могут быть достаточно длинными. Набор команд, установленный для считывания и записи ЭТДП, специально разработан для того, чтобы избежать монополизации каналов связи на длительные периоды времени. При необходимости процесс считывания или записи ЭТДП может запускать систематическую последовательность команд. В нижеследующих подразделах описываются процессы, которые могут быть использованы для считывания или записи ЭТДП.

L.2 Первый шаг для доступа к ЭТДП

Первым шагом в считывании или записи ЭТДП является запуск команды «Query TEDS» («Запросить ЭТДП») для определенной ЭТДП. Данная команда предоставляет следующую информацию:

- доступна ли перезапись существующей ЭТДП;
- поддерживается ли данная ЭТДП;
- максимальный размер ЭТДП;
- является ли данная ЭТДП текстовой или двоичной;
- является ли ЭТДП встроенной или виртуальной;
- верно ли текущее содержание ЭТДП;
- текущий размер ЭТДП;
- контрольная сумма ЭТДП;
- не превышает ли последний образ, записанный в ЭТДП, доступный объем памяти.

L.3 Запись в ЭТДП

Если результат ответа на команду «Query TEDS» («Запросить ЭТДП») отражает, что ЭТДП будет принята ИМП, то следует предпринять следующие шаги:

- взять часть ЭТДП (называемую блоком), размер которой не превышает размер максимального сообщения, и отправить данный блок с использованием команды «Write TEDS segment» («Записать сегмент ЭТДП»). При получении команды «Write TEDS segment» («Записать сегмент ЭТДП»), если при этом постоянная копия изменяется в результате записи, ИМП должен сделать внутреннюю отметку о том, что данная ЭТДП недействительна;
- подождать, пока не будет получен ответ от ИМП на команду «Write TEDS segment» («Записать сегмент ЭТДП»);
- выбрать следующий соответствующий блок и отправить другую команду «Write TEDS segment» («Записать сегмент ЭТДП»);
- процесс должен повторяться до тех пор, пока ЭТДП не будет передана полностью.
- после того как данные ЭТДП были полностью переданы, необходимо отправить команду «Update TEDS» («Обновить ЭТДП») для сообщения о том, что данные ЭТДП были записаны, и дождаться ответа на команду «Update TEDS» («Обновить ЭТДП»);
- ИМП далее должен провести проверку всего содержания полученной ЭТДП и пометить ее как действительную. Как минимум, такой процесс подразумевает проверку встроенной контрольной суммы. Если все параметры верны, то ИМП завершает задачу записи ЭТДП в энергонезависимую память перед отправкой ответа на команду «Update TEDS» («Обновить ЭТДП»).

L.4 Считывание ЭТДП

Если ЭТДП отмечена как виртуальная, она не может быть считана при помощи описанной ниже процедуры, так как такая ЭТДП не размещена в ИМП. Задача обнаружения и считывания виртуальной ЭТДП возлагается на СПП или главный процессор.

Для встроенной ЭТДП, если она поддерживается, а ее текущее содержание действительно, вся ЭТДП может быть полностью считана с применением последовательности следующих операций:

- отправить команду «Read TEDS segment» («Считать сегмент ЭТДП»);
- дождаться ответа;
- повторять два предыдущих шага до тех пор, пока вся ЭТДП не будет считана полностью.

СПП или главный процессор определяет, когда необходимо прекратить процесс считывания, при помощи учета текущего размера ЭТДП или иного механизма. Нет необходимости считывать ЭТДП последовательно или полностью. Это важно при считывании текстовой ЭТДП, когда системе необходимо получить только текстовую информацию, представленную на определенном языке. Тем не менее следует отметить, что за исключением текстовой ЭТДП, частичное считывание ЭТДП не позволяет использовать контрольную сумму для проверки правильности считывания ЭТДП. Для текстовых ЭТДП заголовок и каждый языковой блок внутри ЭТДП имеет отдельную контрольную сумму. ИМП не должен возвращать байты, если запрашиваются данные за пределами окончания ЭТДП.

Приложение М
(справочное)

Конфигурации логических блоков триггера

М.1 Логический блок триггера, дополненный встроенным исполнительным устройством с временной задержкой

Чтобы получить точные одновременные выборки данных от многочисленных ИМП, канал преобразователя может быть дополнен встроенным исполнительным устройством с временной задержкой, позволяющим вводить программируемые задержки с момента, когда триггерное сообщение получено логическим блоком передачи данных, и до момента, когда триггерный сигнал передан в логический блок контроля выборок данных. Данный процесс показан на рисунке М.1. Данная функция является опциональной, а ее реализация остается на усмотрение изготовителя преобразователя для удовлетворения жестких требований к одновременному срабатыванию триггеров на входах и выходах. Данная модель может быть использована для компенсации задержек распространения.

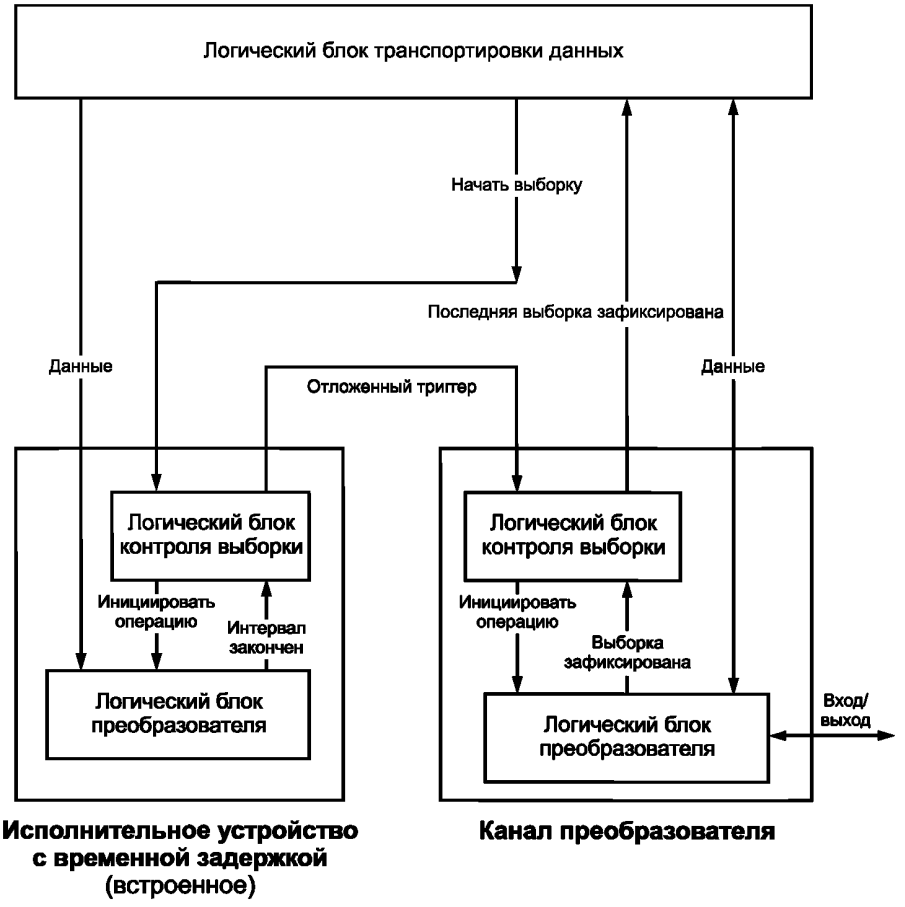


Рисунок М.1 — Канал преобразователя с изменяемым временем задержки триггера

Следует обратить внимание, что исполнительное устройство с временной задержкой реализуется на практике как канал преобразователя с особой аппаратной связью с первичным каналом преобразователя. Он должен быть идентифицирован как группа каналов в мета-ЭТДП. Изготовитель не обязан обеспечивать триггерное сообщение, адресованное исполнительному устройству с временной задержкой.

В номинальной триггерной модели логический блок передачи данных отвечает на триггерные сообщения. При получении триггерного сообщения, адресованного главному каналу преобразователя, при условии, что функция триггера активна, логический блок передачи данных отправляет сигнал «Start Samp» («Начать выборку»), который передается соответствующему исполнительному устройству с временной задержкой. По истечении соответствующего интервала задержки элемент отправляет главному каналу преобразователя команду о начале выборки.

В данной конфигурации нет функций аппаратного оборудования ИМП, позволяющих помочь системе определить время фиксации выборки. Для расчета времени фиксации выборки применяется алгоритм, приведенный в 5.11, расширенный и приспособленный для функции преднамеренной задержки времени, внесенной исполнительным устройством с временной задержкой. Время выборки рассчитывается как поправка ко времени суток, связанному с отправкой триггерного сообщения от СПП.

В данном стандарте не обсуждается, какие функции требуются для измерения задержки распространения; более детально данный вопрос можно изучить в других стандартах комплекса ИИЭР 1451. Предположим, что внутри системы задержка распространения в средствах связи преобразователя может быть динамически определена. Таким образом, получается, что каждый ИМП имеет измеренную задержку распространения «PD» («Propagation Delay»), уникальную для каждого ИМП, но постоянную для всех каналов преобразователя внутри данного ИМП. Каждый канал преобразователя имеет известную величину входящей задержки распространения (tpd_i) внутри логического блока передачи данных, заданную в ЭТДП канала преобразователя. Наличие встроенного исполнительного устройства с временной задержкой позволяет системе вводить компенсационную задержку, чтобы гарантировать точную одновременную выборку данных для многочисленных ИМП в информационном канале преобразователя. Точнее, система может программировать каждое исполнительное устройство с временной задержкой TD_i таким образом, чтобы выражение ($PD_i + tpd_i + TD_i$) являлось бы единой постоянной величиной C для всех каналов преобразователя (то есть для всех значений i) во всех синхронизируемых устройствах. В этом случае для расчета времени первой выборки в наборе данных используется следующая формула:

$$T_1 = T_{tm} + C,$$

где T_{tm} — время суток, в которое от СПП было отправлено триггерное сообщение;
C — постоянная величина, описанная выше;
T_1 — время суток, в которое была зафиксирована первая выборка в наборе данных.

Время любой другой выборки в наборе данных может быть рассчитано с применением алгоритма, представленного в 5.11.3.

М.2 Логический блок триггера, дополненный встроенным датчиком временного события («TimeInstance»)

Если обычная триггерная модель не обеспечивает требуемой точности по времени, то датчик может быть дополнен встроенным датчиком временного события («TimeInstance»), как показано на рисунке М.2.

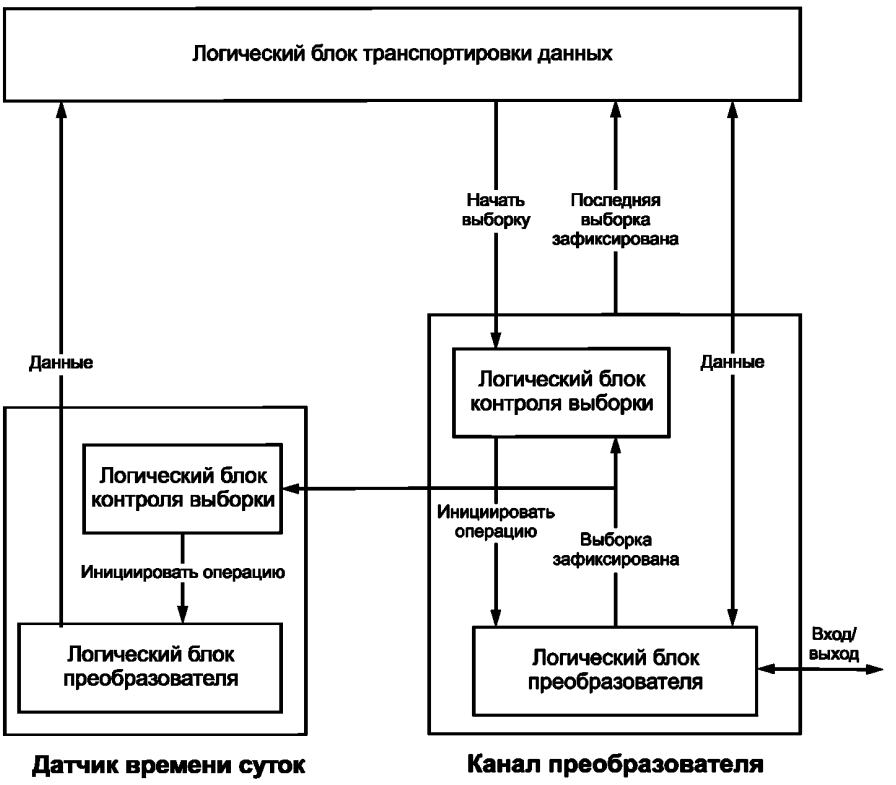


Рисунок М.2 — Канал преобразователя с датчиком временного события («TimeInstance»)

На рисунке М.2 изображен датчик временного события («TimeInstance»), который используется совместно с каналом преобразователя. Следует отметить, что датчик временного события («TimeInstance») реализуется на практике как канал преобразователя с особой аппаратной связью с первичным каналом преобразователя. Он должен быть идентифицирован как группа каналов в мета-ЭТДП. Изготовитель не обязан обеспечивать триггерное сообщение, адресованное датчику временного события («TimeInstance»).

Для каналов преобразователя, дополненных встроенным датчиком временного события («TimeInstance»), ЭТДП канала преобразователя первичного канала преобразователя должна отражать в графе «Source» («Источник») для поля «Time of sample» («Время выборки»), что датчик временного события («TimeInstance») обеспечивает отметку времени. Дальнейшие настройки для значения времени не требуются.

Если первичный канал преобразователя — датчик, то изготовитель может предоставить доступ к обоим ресурсам через прокси-канал преобразователя. Преимущество такого способа заключается в пакетировании набора данных от обоих датчиков в общий набор данных, который считывается за одну операцию.

М.3 Логический блок триггера, дополненный встроенным датчиком временных интервалов «time interval»

Если обычная триггерная модель не обеспечивает требуемой точности по времени, то датчик может быть дополнен встроенным датчиком временных интервалов «time interval». Используемый метод должен быть задан в графе «Source» («Источник») для поля «Time of sample» («Время выборки») в ЭТДП канала преобразователя (см. 8.5.2.40). Датчик временных интервалов реализуется на практике как канал преобразователя с особой аппаратной связью с первичным каналом преобразователя. Он должен быть идентифицирован как группа каналов в мета-ЭТДП. Изготовитель не обязан обеспечивать триггерное сообщение, адресованное датчику временных интервалов.

Если графа «Source» («Источник») для поля «Time of sample» («Время выборки») показывает наличие датчика временного интервала, связанного с помощью интерфейса с первичным каналом преобразователя, то датчик временных интервалов взаимодействует с первичным каналом преобразователя по схеме, приведенной на рисунке М.3.

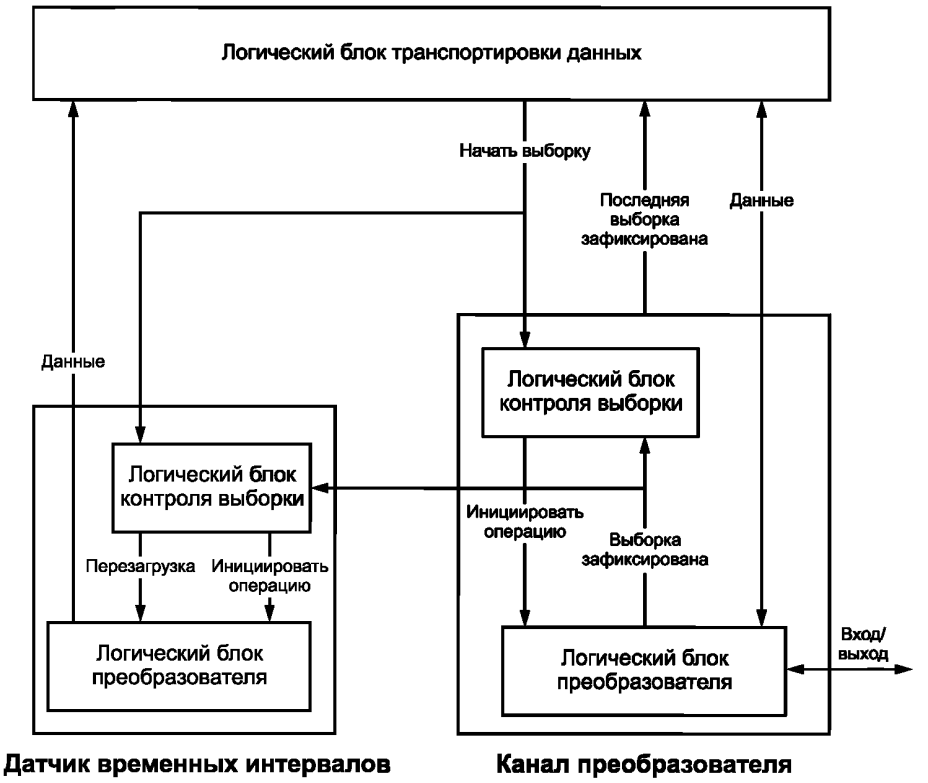


Рисунок М.3 — Канал преобразователя с датчиком временных интервалов, производящий выборку(и)

Датчик временных интервалов измеряет интервалы между временем получения триггерного сообщения и фактическим временем фиксации выборки. В зависимости от модели данных датчика временных интервалов, заданной в поле «Maximum data repetitions» («Максимальное повторение данных») ЭТДП канала преобразователя, изготовителю следует применять данную модель одним из двух способов.

В случае если значение поля максимального повторения данных установлено на «1», изготовителем выбрано единственное значение временного интервала для всего набора данных, что означает, что выборки осуществляются через равные интервалы времени. В таком случае сообщаемое значение представляет собой временной интервал, связанный с первой выборкой в наборе данных. Для расчетов можно воспользоваться следующей формулой:

$$T_1 = T_{tm} + t_{pdi} + t_{tis},$$

где T_{tm} — время суток, отражающее отправку триггерного сообщения СПП;
t_{pdi} — время входящей задержки распространения в поле логического блока передачи данных в ЭТДП канала преобразователя;
t_{tis} — временной интервал, передаваемый датчиком временных интервалов;
T_1 — время суток, отражающее фиксацию первой выборки в наборе данных.

Данная формула позволяет рассчитать время первой выборки в наборе данных. Время любой другой выборки в наборе данных может быть рассчитано с применением алгоритма, приведенного в 5.11.3.

В случае если значение поля максимального повторения данных (в ЭТДП канала преобразователя, см. 8.5.2.28) датчика временных интервалов больше нуля и равно значению максимального повторения данных соответствующего канала преобразователя, изготовитель выбрал возможность установки значения временного интервала для каждой выборки в наборе данных, что означает вариации временных интервалов между выборками. Для расчетов можно воспользоваться следующей формулой:

$$T_{(i)} = T_{tm} + t_{pdi} + t_{tis(i)},$$

где T_{tm} — время суток, отражающее отправку триггерного сообщения СПП;
t_{pdi} — время входящей задержки распространения в поле логического блока передачи данных в ЭТДП канала преобразователя;
$t_{tis\ (i)}$ — временной интервал, передаваемый датчиком временных интервалов для *i*-выборки;
$T_{(i)}$ — время суток, отражающее фиксацию *i*-выборки в наборе данных.

Если первичный канал преобразователя является датчиком, то изготовитель может предоставить доступ к обоим ресурсам через прокси-канал преобразователя. Преимущество такого способа заключается в пакетировании набора данных от обоих датчиков в общий набор данных, который считывается за одну операцию.

Приложение N
(справочное)

Система обозначений для IDL

В стандарте ИИЭР 1451.0 применяется язык описания интерфейса для описания программных интерфейсов, используемых в настоящем стандарте. Система обозначения для таких описаний подчиняется правилам для языка определения интерфейсов IDL («Interface Definition Language»), стандартизованном в ИСО 14750. Дополнительная информация об IDL содержится на веб-сайте Object Management Group:

www.omg.org/gettingstarted/omg_idl.htm.

N.1 Ключевые функции IDL

IDL — язык, который обеспечивает инфраструктуру для разделения интерфейсов программных продуктов и интерфейсов языка программирования данных приложений. Язык IDL обеспечивает написание программных интерфейсов ИИЭР 1451.0 абстрактным способом, не зависящим от фактических кодов, которые могут быть использованы при практической реализации данных интерфейсов. Применение IDL в настоящем стандарте позволяет разработчикам, опирающимся на стандарт ИИЭР 1451.0, инкапсулировать свои решения таким образом, чтобы сторонним лицам были доступны исключительно интерфейсы, определенные IDL.

Интерфейс OMG IDL достаточно хорошо адаптирован под требования уровня ИИЭР 1451.0, в котором сложные структуры данных компактно определены с использованием нескольких стандартных типов данных, оговоренных в IDL.

Такие функции чрезвычайно важны в случае, если уровень ИИЭР 1451.0 находится в разнородной среде, потенциально состоящей из множества различных языков программирования, операционных систем и сетевых протоколов.

Важной особенностью языка IDL также является то, что программные интерфейсы, описанные через обозначения IDL, независимы от языка программирования, который мог быть использован на этапе реализации программных приложений.

N.2 Пример обозначений IDL с разъяснениями

Следующий пример содержит реализованное на языке IDL описание программного интерфейса для класса, определяющего представление во времени.

N.2.1 Пример 1

На основе данных по 4.9 выбран пример описания на языке IDL, изображенный на рисунке N.1.

Данный сегмент на языке IDL снабжен предварительными исходными комментариями для читателей. Такие опциональные комментарии начинаются с символа «//» (прямого двойного слеша, или двойной наклонной черты), как показано в данном примере.

В данном примере слово «struct» является ключевым словом языка IDL, которое является определителем типа структуры, предназначенным для привязки метки «TimeRepresentation» («Представление времени») к структуре данных, обозначенной в секции «interface body» («тело интерфейса») IDL описания.

Тело выражения на языке IDL содержится внутри фигурных скобок «{… IDL interface body …}». Каждое выражение внутри фигурных скобок завершается знаком точки с запятой «;».

```
// Это абстрактный класс, который определяет представление времени. Он подразделяется на
// TimeInstance и TimeDuration. Определение двух данных аргументов представлено в
// Таблице 1


struct TimeRepresentation {
        UInt32 secs;
        UInt32 nsecs;
        } ;
```

Рисунок N.1 — Простой пример системы обозначений на языке IDL, используемой для определения структуры данных, содержащей два элемента

Первый оператор на языке IDL в теле данного выражения IDL показывает, что структура данных «TimeRepresentation» начинается с 32-разрядного целого числа без знака, которое представляет собой число секунд и отмечено идентификатором «secs».

Второй оператор в теле инерфейса данного выражения IDL показывает, что второе значение в структуре данных «TimeRepresentation» является 32-разрядным целым числом без знака, которое представляет собой число наносекунд и отмечено идентификатором «nsecs».

Закрытие фигурных скобок означает, что тело IDL интерфейса завершено.

N.2.2 Пример 2

Выражения на языке IDL, представленные на рисунке N.2, включают в себя частичный листинг описания сервисов модульных связей для случая реализации типа связи «точка-точка» («узел-узел»).

```
//================  Сервисы модульных связей  ===============

module ModuleCommunication

{

 // Comm интерфейс “точка-точка” (узел-узел)

 // Обеспечивается устройствами уровня ИИЭР 1451.X

 // Используется устройствами уровня ИИЭР 1451.0


 interface Comm

 {

  Args::UIntl6 init( );

 };


 // Comm интерфейс “точка-точка” (узел-узел)


 interface P2PComm : Comm

 {

  Args::UIntl6 read( in Args::TimeDuration timeout,

                     in  Args::UInt32       maxLen,

                     out Args::OctetArray  payload,

                     out Args::_Boolean    last );

 };
```

Рисунок N.2 — Операторы IDL, определяющие модуль с описаниями для двух интерфейсов

За первоначальной строкой комментариев следует первая строка данного выражения IDL, которая представляет собой оператор объявления модуля. «Модуль» используется в целях определения области действия переменной для создания уникального пространства имен, чтобы идентификаторы, определенные в рамках данного модуля, могли распознаваться только внутри интерфейса модульных связей. Содержание данного модуля ограничено открывающейся и закрывающейся фигурными скобками «{ ...}». Использование конструкции модуля позволяет сократить конфликты областей действия переменных, которые могут возникнуть в масштабных проектах, и уменьшает необходимость создания большого числа глобальных имен.

После нескольких строк комментариев идет ключевое слово «interface», определяющее секцию IDL, описывающую программные связи для реализации сервиса для клиентов, использующих данный сервис.

В данном примере ключевое слово «interface» определяет начало данного описания и указывает, что данному интерфейсу будет присвоено имя «Comm». Открывающаяся фигурная скобка «{« означает начало тела данного интерфейса.

Первый оператор «Args::UInt16 init( );» в теле интерфейса «Comm» необходим для объявления рабочих операций. Данный оператор показывает, что операция init( ) не имеет параметров, но возвращает 16-разрядное целое число без знака. «Args::Uint16» отображает возвращаемое значение для операции init( ).

Префикс Args с последующим двойным двоеточием «Args::___» означает, что идентификатор Unit16 определен с описанием его области действия в модуле «Args» в каком-либо другом месте документа.

Определение интерфейса «Comm» завершается закрывающейся фигурной скобкой «}» и сопровождается последующим комментарием, который вводит определение следующего интерфейса.

IDL-оператор «interface P2PComm : Comm» запускает определение следующего интерфейса. Данный интерфейс называется «P2PComm» и наследует ранее определенный интерфейс «Comm». Таким образом, интерфейс «P2PComm» создается как производный от интерфейса «Comm», что позволяет интерфейсу «P2PComm» использовать элементы, определенные в интерфейсе «Comm», как если бы данные элементы были определены в самом интерфейсе «P2PComm».

Следующий оператор интерфейса «P2PComm» «Args::UInt16 read(in Args::TimeDurationtimeout, ….» объявляет операцию считывания «read(…)» с ее входными и выходными данными. Текст «Args:Uint16», предшествующий тексту «read(…)», означает, что операция считывания «read» возвращает 16-разрядное целое число без знака, которое обычно отражает информацию о том, была ли операция считывания успешной или неуспешной. В дополнение к возвращаемому после завершения операции 16-разрядному целому числу без знака определение на языке IDL также описывает входные и выходные параметры, используемые при операции считывания «read».

Оператор «in Args::TimeDuration timeout» означает, что операция считывания «read» требует входной параметр типа TimeDuration. Данный параметр определен с заданием области его действия в модуле «Args» и идентифицирован как «timeout».

Баланс определения интерфейса «P2PComm» идентифицирует другие входные и выходные параметры, требуемые интерфейсом «P2PComm». Второй входной параметр, требуемый данным интерфейсом, — это идентификатор «maxLen», который определен как 32-разрядное целое число без знака. Данный интерфейс также включает в себя описание параметров, которые должны быть переданы для операции считывания «read», но заполняются в результате выполнения операции считывания. Как указано в последних двух операторах данного определения IDL, выходные величины, которые генерируются операцией считывания, будут приписаны двум идентификаторам: «payload» и «last». Как показано на рисунке N.2, «payload» является массивом 8-битовых байтов, именуемых байтовым массивом (OctetArray), а «last» является логической переменной типа Boolean, которая может иметь только два возможных состояния: «True» («Истина») или «False» («Ложь»).

Используя ссылку в начале данного приложения, можно найти более полное описание функций и синтаксиса языка IDL. Данные ссылки могут быть полезны при более подробном изучении системы обозначений языка IDL, используемых в настоящем стандарте.

Приложение О
(справочное)

Реализация ЭТДП простого датчика

В О.1—О.4 описана структура ЭТДП простого ИМП. В данном ИМП реализован один канал информации о температуре, считываемый терморезистором. Калибровка представляет собой простую линейную регрессию в пределах рабочего диапазона, а канал не запускается с помощью триггерных команд, но в любое время возвращает показания при запросе. После применения коррекции датчик будет предоставлять данные на выходе в градусах Цельсия (°С). Рабочий диапазон составляет от минус 40 °С до плюс 80 °С с погрешностью ± 2 °С.

В 5.5 указано, что все ИМП содержат мета-ЭТДП, ЭТДП канала преобразователя, ЭТДП с именем преобразователя, заданным пользователем, и ЭТДП физического уровня. Формат ЭТДП физического уровня выходит за рамки рассмотрения настоящего стандарта и здесь не описывается. Кроме того, любой преобразователь, выходной сигнал которого не откалиброван в физических единицах измерения, должен содержать ЭТДП калибровки. Поскольку выходной сигнал канала преобразователя является 12-битным выходным сигналом аналого-цифрового преобразователя, калибровочная ЭТДП поставляется с выходными единицами измерения в виде градусов Цельсия.

О.1 Мета-ЭТДП

Формат мета-ЭТДП описан в 8.4. В настоящем разделе подробно рассмотрена структура, приведенная в таблице 43 и на рисунке 13.

О.1.1 Идентификация мета-ЭТДП

Формат поля идентификации представлен в таблице О.1 (здесь продублирована таблица 41).

Таблица О.1 — Идентификатор ЭТДП

Поле	Содержание	Функция
Тип	03	Поле типа для идентификатора ЭТДП
Длина	04	Данное поле всегда устанавливается равным 04, указывая на то, что поле «Значение» содержит 4 байта
Семейство	00	В данном поле указывается стандарт комплекса стандартов ИИЭР 1451, который определяет данную ЭТДП
Класс	01	В данном поле указывается ЭТДП, к которой осуществляется доступ. Для мета-ЭТДП это значение 01 (см. таблицу 17)
Версия	01	В данном поле указывается версия ЭТДП. Значение представляет собой номер версии, определенный в настоящем стандарте. Значение 01 указывает на то, что данная ЭТДП согласована с первым выпуском настоящего стандарта
Длина кортежа	01	В данном поле указывается число байтов в поле «Длина» всех кортежей в ЭТДП, за исключением данного кортежа. В данном случае это значение 01

Преобразуя содержание полей в байтовый формат TLV, получаем последовательность байтов в следующем виде:

03 04 00 01 01 01.

О.1.2 Поле «УУИД»

Поле «УУИД» описано в 4.12 и в таблице О.2 (таблица О.2 дублирует таблицу 4).

Таблица О.2 — Структура типов данных универсальной уникальной идентификации (УУИД)

Поле	Описание	Число битов
1	Поле «Место нахождения»: значение данного поля должно быть выбрано изготовителем ИМП для определения конкретного места на Земле, места нахождения, над которым изготовитель имеет физический контроль. Данное значение может представлять фактическое место нахождения изготовителя ИМП. Изготовитель может использовать в своей работе различные значения данного поля, но только если они удовлетворяют требованиям настоящего подраздела. Поле «Место нахождения» должно быть представлено 42 битами. Старший значащий бит указывает на северную (бит установлен) или южную (бит не установлен) широту. Следу-	42

Окончание таблицы О.2

Поле	Описание	Число битов
	ющие 20 старших значащих битов данного поля представляют собой значение широты места нахождения как целое число угловых секунд. Следующий старший значащий бит должен указывать на восточную (бит установлен) или западную (бит не установлен) долготу. Остальные 20 бит представляют значение долготы места нахождения как целое число угловых секунд. Значения широты более 90° зарезервированы. Значения долготы более 180° зарезервированы. Примечание — Одна угловая секунда на экваторе составляет около 30 м. Таким образом, диапазон, представляемый каждым 20-битовым полем, составляет от 0 до 1 048 575 угловых секунд или от 0° до 291°, что является достаточным для представления широты и долготы на поверхности Земли	
2	Поле «Изготовитель»: значение данного поля может быть выбрано изготовителем ИМП для любых целей при условии, что не возникает конфликтных ситуаций, связанных с совпадениями при использовании поля «Место нахождения». Такая конфликтная ситуация в поле «Место нахождения» происходит в том случае, если на физический контроль над местом нахождения, заданным в поле «Место нахождения», могут претендовать более одного изготовителя. Если такой конфликт существует, то все пострадавшие изготовители должны согласовать использование значений поля «Изготовитель» для исключения каких-либо совпадений. Таким образом, сочетание поля «Место нахождения» и поля «Изготовитель» должно однозначно определить конкретного изготовителя ИМП. Такое согласование должно возобновляться каждый раз при совпадении, вызывающем конфликтную ситуацию	4
3	Поле «Год»: значение данного поля должно отображать текущий год. Поле «Год» должно быть представлено 12-разрядным целым значением. Диапазон данного поля должен составлять от 0 г. до 4095 г. н. э. Началом года принято считать 1 января, 00:00:00 по TAI	12
4	Поле «Время»: данное значение должно быть выбрано изготовителем ИМП таким образом, чтобы в сочетании с полями «Место нахождения», «Изготовитель» и «Год» результирующий УУИД являлся уникальным для всех ИМП, сделанных под контролем данного изготовителя. Выбор значений для поля «Время» должен быть, кроме того, ограничен таким образом, чтобы значения, которые интерпретируются как время с начала года, не представляли ни время, предшествующее получению изготовителем физического контроля над местом нахождения, ни значений времени в будущем. Поле «Время» должно быть представлено 22-разрядным целым числом. Диапазон должен составлять от 0 до 4 194 303. Если необходимо интерпретировать данное поле как время с начала года, то оно должно быть представлено целым числом 10-секундных интервалов. В этом случае значения времени более одного года зарезервированы. Началом года принято считать 1 января, 00:00:00 по TAI. Примечание — В году примерно 31 536 000 с	22

Например, ИМП был изготовлен 14 августа 2005 г. Он являлся 120-м модулем, произведенным в этот день. Геопространственные координаты места изготовления ИМП — N40.3780 W105.0894.

УУИД будет иметь следующие значения.

Место нахождения: координаты места нахождения завода-изготовителя — N40.3780 W105.0894. В угловых секундах — N145367 W381218.

В этом месте находится только один изготовитель, поэтому значение поля «Изготовитель» равно 0.

Значение поля «Год» — 2005.

Дата изготовления — 14 августа. Датчик являлся 120-м датчиком, произведенным в этот день. Существует определенная свобода в формате данного поля, потому завод-изготовитель использует следующий формат для поля «Время»: день года × 1000 + порядковый номер (то есть порядковый номер модуля, произведенного в этот день). 14 августа является двести двадцать четвертым днем в году, так что поле «Время» составит: 224 × 1000 + 120 = 2 240 120.

Далее необходимо преобразовать поля в двоичный код и объединить их, как показано в таблице О.3.

Таблица О.3 — Развитие УУИД

Поле	Значение	Двоичное значение	Размер поля (биты)
N/S (Север-Юг)	N	1	1
Latitude (Широта)	145367	00000011100000011111	20

Окончание таблицы О.3

Поле	Значение	Двоичное значение	Размер поля (биты)
E/W (Восток-Запад)	W	0	1
Longitude (Долгота)	381218	01011101000100100010	20
Manufacturer (Изготовитель)	0	0000	4
Year (Год)	2005	011111010101	12
Date/Sequence (Дата/Последовательность)	2240120	1000100010111001111000	22

Перестройка и преобразование байтов позволяет получить формат УУИД, показанный в таблице О.4 (в шестнадцатеричной системе счисления).

Таблица О.4 — Байты УУИД

Двоичная система счисления	Шестнадцатеричная система счисления
10000001	81
11000000	C0
11111001	F9
01110100	74
01001000	48
10000001	81
11110101	F5
01100010	62
00101110	2E
01111000	78

В итоге данные значения преобразуются в формат TLV. Поле «тип» для УУИД равно 04, а длина поля составляет 10 байтов. Это приводит к тому, что 2 байта 04 0A должны быть размещены в начале блока. Таким образом, окончательный вид данного поля в виде байтов TLV выглядит следующим образом:

04 0A 81 C0 F9 74 48 81 F5 62 2E 78.

О.1.3 Рабочее время ожидания (тайм-аут)

Рабочее время ожидания для данного ИМП составляет 0,5 с, что является временем ожидания для обычных операций. Число 0,5 будет представлено в виде числа с плавающей точкой F32. Преобразуя данное значение в двоичную систему счисления, получаем его в следующем виде (подробная информация представлена в ИИЭР 754—1985):

0 01111110 00000000000000000000000.

Данное число, в свою очередь, принимает следующий вид в шестнадцатеричной системе счисления:

3F 00 00 00.

Добавление значения 10 для поля «Тип» (шестнадцатеричное значение 0A) и значения 4 для поля «Длина» позволяет получить следующую окончательную последовательность байтов TLV:

0A 04 3F 00 00 00.

О.1.4 Время ожидания при медленном доступе

Время ожидания при медленном доступе не отличается от обычного времени ожидания из-за отсутствия каких-либо медленных операций, кроме операции самодиагностики (описанной ниже). В связи с этим данное необязательное поле не включается.

О.1.5 Время ожидания при самодиагностике

Время ожидания при самодиагностике для данного ИМП составляет 5,0 с. Данное время необходимо для выполнения самодиагностики. Число 5,0 представляется в виде числа с плавающей точкой F32. Преобразование данного числа в двоичную систему счисления позволяет получить его в следующем виде:

1 10000001 01000000000000000000000,

который, в свою очередь, принимает следующий вид в шестнадцатеричной системе счисления:

C0 A0 00 00.

Добавление значения 12 для поля «Тип» и значения 4 для поля «Длина» позволяет получить следующую окончательную последовательность байтов TLV:

0C 04 C0 A0 00 00.

О.1.6 Число каналов преобразователя

Данный ИМП содержит единственный канал. Таким образом, значение данного поля — 1. Тип данных — U16E, и тип поля — 13. Таким образом, окончательная последовательность байтов TLV:

0D 02 00 01.

О.1.7 Другие поля

Данный ИМП имеет только один канал, поэтому мета-ЭТДП не содержит полей контрольной группы, векторной группы или прокси-канала.

О.1.8 Окончательный байтовый формат мета-ЭТДП

В итоге необходимо объединить и добавить к заголовку структуры информацию об общей длине. Данный процесс описан в 8.1. Общее число байтов для нашей мета-ЭТДП — 36 (24 в шестнадцатеричной системе счисления), включая 34 байта блока данных и 2 байта контрольной суммы. Поле «Длина» составляет 4 байта, так что вводится байтовое значение 00 00 00 24.

Контрольная сумма вычисляется путем сложения байтов полей «Данные» и «Длина» и дополнения их суммы до единицы, как описано в 8.1.6. Сумма байтов равна 0x72E, а ее дополнение будет 0xF8D1. Окончательный вид мета-ЭТДП в байтовом формате приведен в таблице О.5.

Таблица О.5 — Пример мета-ЭТДП

MSB LSB	Определение поля
00 00	Общая длина
00 24	
03 04	Заголовок
00 01	
01 01	03 04 00 01 01 01
04 0A	УУИД
81 C0	04 0A 81 C0 F9 74 48 81 F5 62 2E 78
F9 74	
48 81	
F5 62	
2E 78	
0A 04	Рабочее время ожидания
3F 00	0A 04 3F 00 00 00
00 00	
0C 04	Время ожидания для самодиагностики
C0 A0	0C 04 C0 A0 00 00
00 00	
0D 02	Число каналов
00 01	0D 02 00 01
F8 82	Контрольная сумма

О.2 ЭТДП канала преобразователя

Формат ЭТДП канала преобразователя приведен в 8.5.2, таблице 49 и на рисунке 14.

О.2.1 Идентификация ЭТДП канала преобразователя

В таблице О.6 перечислены форматы для поля идентификации (из таблицы 41).

Таблица О.6 — Идентификатор ЭТДП канала преобразователя

Поле	Содержание	Функция
Тип	03	Поле типа для идентификатора ЭТДП
Длина	04	Данное поле всегда устанавливается равным 04, указывая на то, что поле «Значение» содержит 4 байта
Семейство	00	В данном поле указывается стандарт комплекса стандартов ИИЭР 1451, который определяет данную ЭТДП
Класс	03	В данном поле указывается ЭТДП, к которой осуществляется доступ. Для ЭТДП канала преобразователя это значение 03 (из таблицы 17)
Версия	01	В данном поле указывается версия ЭТДП. Значение представляет собой номер версии, определенный в настоящем стандарте. Значение 01 указывает на то, что данная ЭТДП согласована с первым выпуском настоящего стандарта
Длина кортежа	Число байтов	В данном поле указывается число байтов в поле «Длина» всех кортежей в ЭТДП, за исключением данного кортежа. В данном случае это значение 01

Преобразуя содержание полей в байтовый формат TLV, получаем последовательность байтов в следующем виде:

03 04 00 03 01 01.

О.2.2 Ключ калибровки

Данный ИМП предоставляет информацию о калибровке, которую необходимо применить в СПП или приложении. Выходной сигнал самого ИМП представляет собой отсчеты аналого-цифрового преобразователя. СПП использует информацию о калибровке для преобразования в температуру. Соответствующее значение ключа калибровки равно 1 (CAL_SUPPLIED).

Тип поля для ключа калибровки равен 10 (0А в шестнадцатеричной системе счисления). Преобразуя данные значения в байтовый формат TLV, получаем последовательность байтов в следующем виде:

0А 01 01.

О.2.3 Ключ типа канала преобразователя

Данный ИМП имеет один канал датчика, поэтому тип преобразователя имеет значение 0 (датчик). Тип поля для типа канала преобразователя имеет значение 11. Преобразуя данные значения в байтовый формат TLV, получаем последовательность байтов в следующем виде:

0В 01 00.

О.2.4 Физические единицы

Данный ИМП выдает на выходе температуру в градусах Цельсия после применения коррекции. Описание единиц измерения приведено в 4.11. Первое поле дескриптора UNIT установлено на 0 (PUI_SI_UNITS). Поля 2—10 установлены на 128, за исключением поля 8 (Kelvin), которое установлено на 130, чтобы показать линейную температуру. Управление параметрами преобразования в градусы Цельсия осуществляется в ЭТДП калибровки. Таким образом, дескриптор единиц измерения имеет вид, указанный в таблице О.7.

Таблица О.7 — Физические единицы измерения

Поле	Описание	Тип поля TLV (шестнадцатеричное значение типа поля TLV)	Значение	Шестнадцатеричное значение
1	Интерпретация физических единиц измерения (см. таблицу 3)	50 (32)	0	0
2	(2 × <показатель степени для единицы «радиан»>) + 128	51 (33)	128	80
3	(2 × <показатель степени для единицы «стерадиан»>) + 128	52 (34)	128	80

Окончание таблицы О.7

Поле	Описание	Тип поля TLV (шестнадцатеричное значение типа поля TLV)	Значение	Шестнадцатеричное значение
4	(2 × <показатель степени для единицы «метр»>) + 128	53 (35)	128	80
5	(2 × <показатель степени для единицы «килограмм»>) + 128	54 (36)	128	80
6	(2 × <показатель степени для единицы «секунда»>) + 128	55 (37)	128	80
7	(2 × <показатель степени для единицы «ампер»>) + 128	56 (38)	128	80
8	(2 × <показатель степени для единицы «кельвин»>) + 128	57 (39)	130	82
9	(2 × <показатель степени для единицы «моль»>) + 128	58 (3A)	128	80
10	(2 × <показатель степени для единицы «кандела»>) + 128	59 (3B)	128	80

Тип поля для физических единиц измерения имеет значение 12. Так как данный датчик только измеряет температуру, в ЭТДП должны быть введены кельвины. Значения для всех других основных единиц измерения по умолчанию устанавливаются на 128. Преобразуя данные значения в байтовый формат TLV, получаем последовательность байтов в следующем виде:

0C 06 32 01 00 39 01 82.

О.2.5 Проектная нижняя рабочая граница

Данный ИМП имеет нижнюю рабочую границу минус 40 °С. Тем не менее нижняя рабочая граница должна быть указана в единицах СИ, то есть в кельвинах. Следовательно, необходимо выполнить преобразование с использованием формулы «градусы Кельвина (К) = градусы Цельсия (°С) + 273,15». Так как значение данной границы является приблизительным, округляем полученное значение поля до 233. Формат поля — F32, таким образом, преобразование данного числа в двоичную систему счисления позволяет получить его в следующем виде (более подробная информация представлена в ИИЭР 754—1985):

0 10000110 11010010000000000000000,

который, в свою очередь, принимает следующий вид в шестнадцатеричной системе счисления:

43 69 00 00.

Добавление поля типа 13 (шестнадцатеричное значение 0D) и длины 4 позволяет получить окончательную последовательность байтов TLV в следующем виде:

0D 04 43 69 00 00.

О.2.6 Проектная верхняя рабочая граница

Данный ИМП имеет верхнюю рабочую границу 80 °С. Тем не менее верхняя рабочая граница должна быть указана в единицах СИ, то есть в кельвинах. Следовательно, должно быть выполнено преобразование с использованием формулы «градусы Кельвина (К) = градусы Цельсия (°С) + 273,15». Так как значение данной границы является приблизительным, округляем полученное значение поля до 353. Формат поля — F32, таким образом, преобразуя данное значение в двоичную систему счисления, получаем его в следующем виде (более подробная информация представлена в ИИЭР 754—1985):

0 10000111 01100001000000000000000,

который, в свою очередь, принимает следующий вид в шестнадцатеричной системе счисления:

43 B0 80 00.

Добавление поля типа 14 (шестнадцатеричное значение 0E) и длины 4 позволяет получить следующую окончательную последовательность байтов TLV:

0E 04 43 80 00 B0.

О.2.7 Погрешность при наихудших условиях

Данный ИМП имеет максимальную ошибку в 2,0 °С. Поскольку это ошибка, то нет необходимости преобразовывать ее в градусы Кельвина. Формат поля — F32, таким образом, преобразование данного числа в двоичную систему счисления позволяет получить его в следующем виде:

0 10000000 00000000000000000000000,

который, в свою очередь, принимает следующий вид в шестнадцатеричной системе счисления:

40 00 00 00.

Добавление поля типа 15 и длины 4 позволяет получить следующую окончательную последовательность байтов:

0F 04 40 00 00 00.

О.2.8 Ключ самодиагностики

Данный ИМП имеет возможность самодиагностики, поэтому данное поле имеет значение 1. Тип поля имеет значение 16, таким образом, последовательность байтов после преобразования в шестнадцатеричную систему счисления имеет вид:

10 01 01.

О.2.9 Возможность работы в нескольких диапазонах

Данный преобразователь не имеет возможности работы в нескольких диапазонах, поэтому данное поле опущено.

О.2.10 Определение образца

Поле определения образца состоит из трех подполей. Окончательная последовательность байтов будет состоять из типа поля 18 и общей длины последующих трех полей и их целых пакетов. ИМП выводит необработанные данные с аналого-цифрового преобразователя, который является 12-разрядным.

О.2.11 Модель данных

ИМП выводит необработанные данные с аналого-цифрового преобразователя, так что значение данного поля будет 0, что означает целое N-байтовое число. Тип поля имеет значение 40, таким образом, окончательная последовательность байтов имеет вид:

28 01 00.

О.2.11.1 Длина модели данных

Поскольку выходной сигнал получают с 12-разрядного аналого-цифрового преобразователя, то ширина поля будет составлять 2 байта. Тип поля имеет значение 40, таким образом, окончательная последовательность байтов имеет вид:

29 01 02.

О.2.11.2 Значащие биты модели

Значение данного поля будет 12, что соответствует 12-разрядному аналого-цифровому преобразователю. Тип поля имеет значение 42, таким образом, окончательная последовательность байтов имеет вид:

2A 01 0C.

О.2.12 Окончательный байтовый формат для определения образца

Суммарная длина всех трех подполей — 9. Объединение данных длин и добавление собственной длины и кода поля для определения образца (18) позволяет получить окончательную последовательность байтов в следующем виде:

12 09 28 01 00 29 01 02 30 01 0C.

О.2.13 Определение набора данных

ИМП производит только одиночные считывания, поэтому число повторений равно 0, и, следовательно, данное поле может быть опущено.

О.2.14 Время обновления канала преобразователя

Данный преобразователь находится в автономном режиме выборки данных и обновляется со скоростью 10 выборок в секунду, то есть 10/с. Таким образом, данное поле имеет значение 0,1, что соответствует максимальному времени между получением команды «Считать» и доступностью отсчета. Тип данных имеет значение F32, а код поля — 20, таким образом, окончательная последовательность байтов имеет вид:

14 04 3D CC CC CD.

О.2.15 Время подготовки к считыванию канала преобразователя

Время подготовки к считыванию для данного преобразователя составляет 25 мкс. Поле имеет формат F32, а код поля — 22, таким образом, окончательная последовательность байтов имеет вид:

16 04 37 D1 B7 17.

О.2.16 Период дискретизации канала преобразователя

Период дискретизации так же, как и время обновления, составляет 0,1 с. Поле имеет формат F32, а код поля — 23, таким образом, окончательная последовательность байтов имеет вид:

17 04 3D CC CC CD.

О.2.17 Время готовности канала преобразователя

Время готовности данного преобразователя составляет 30 с. Код поля имеет значение 24, а формат — F32, поэтому окончательная последовательность байтов имеет вид:

18 04 41 F0 00 00.

О.2.18 Время задержки считывания канала преобразователя

Наибольшее время задержки существующих данных составляет 25 мкс. Код поля имеет значение 25, а формат — F32, поэтому окончательная последовательность байтов имеет вид:

19 04 37 D1 B7 17.

О.2.19 Требование ко времени самодиагностики канала преобразователя

Время самодиагностики для данного канала является таким же, как и общее время самодиагностики, указанное в мета-ЭТДП, и составляет 5 с. Код поля имеет значение 26, а формат — F32, таким образом, окончательная последовательность байтов имеет вид:

1A 04 40 A0 00 00.

О.2.20 Источник времени отсчета

Так как данный датчик работает в автономном режиме выборки данных и без триггерных сигналов, то данное поле имеет значение 0, что указывает на то, что точное время фиксации выборки не может быть определено. Таким образом, данное поле может быть опущено.

О.2.21 Задержки распространения и погрешность выборки

Поскольку канал работает без триггерных сигналов и в автономном режиме выборки данных, то данные поля не имеют значения и не включаются.

О.2.22 Атрибут выборки данных

Преобразователь работает без триггерных сигналов и в автономном режиме выборки данных. Код поля для комбинированного атрибута составляет 31.

О.2.22.1 Возможность выбора режима выборки данных

Преобразователь всегда работает в автономном режиме выборки данных без предварительно заданного счетчика триггера. Таким образом, устанавливается только бит 2 со значением данной установки, равным 2. Код поля имеет значение 48 и формат — U8E, таким образом, окончательная последовательность байтов имеет вид:

30 01 02.

О.2.22.2 Режим выборки данных по умолчанию

Поскольку единственно возможным значением для режима выборки данных является автономный режим без предварительно заданного счетчика триггера, он также является и режимом выборки данных по умолчанию, а данное поле может быть опущено.

О.2.22.3 Окончательная последовательность байтов режима выборки данных

Объединение полей и добавление общей длины 6 байтов позволяет получить окончательную последовательность байтов в следующем виде:

1F 03 30 01 02.

О.2.23 Атрибут буферизации

Существует только один буфер для последовательного считывания. Таким образом, данное поле имеет значение 0 и может быть опущено.

О.2.24 Операция «End of data set» («Окончание набора данных»)

Данное поле доступно только для исполнительных устройств и в данном случае не включается.

О.2.25 Атрибут передачи данных

Простой преобразователь в виде датчика температуры возвращает данные только по запросу без осуществления потоковой передачи данных. Таким образом, данное поле может быть опущено.

О.2.26 Остальные поля

Остальные поля относятся к исполнительным устройствам, датчикам событий или геопространственным датчикам и не включены в данную ЭТДП.

О.2.27 Окончательный формат ЭТДП канала преобразователя

Ниже представлены окончательные поля данных для данной ЭТДП по порядку.

```
03 04 00 03 01 01 (идентификатор)
0A 01 01 (ключ калибровки)
0B 01 00 (ключ типа)
0C 06 32 01 00 39 01 82(единицы измерения)
0D 04 43 69 00 00 (нижняя граница)
0E 04 43 B0 80 00 (верхняя граница)
0F 04 40 00 00 00 (погрешность)
10 01 01 (самодиагностика)
12 09 28 01 00 29 01 02 30 01 0C (определение выборки)
14 04 3D CC CC CD (время обновления)
16 04 37 D1 B7 17 (время подготовки к считыванию)
17 04 3D CC CC CD (период дискретизации)
18 04 41 F0 00 00 (время готовности)
19 04 37 D1 B7 17 (время задержки считывания)
1A 04 40 A0 00 00 (время самодиагностики)
1F 03 31 01 02 (выборка данных)
```

Общая длина ЭТДП канала, включая 2 байта контрольной суммы, составляет 95 байтов. Сочетание полей и добавление 4 байтов информации о длине позволяет получить окончательный формат данной ЭТДП. Контрольная сумма составляет EF2C.

```
00 00 00 5F 03 04 00 03 01 01
0A 01 01 0B 01 00 0C 06 32 01
```

```
00 39 01 82 0D 04 43 69 00 00
0E 04 43 B0 80 00 0F 04 40 00
00 00 10 01 01 12 09 28 01 00
29 01 02 30 01 0C 14 04 3D CC
CC CD 16 04 37 D1 B7 17 17 04
3D CC CC CD 18 04 41 F0 00 00
19 04 37 D1 B7 17 1A 04 40 A0
00 00 1F 03 31 01 02 EF 2C
```

О.3 ЭТДП калибровки

ИМП содержит один терморезистор, который предоставляет показания температуры. Данный терморезистор помещен в микросхему делителя напряжения, и на него подано питающее напряжение 10 В с десятивольтового преобразователя. Характеристики микросхемы обеспечивают линейную кривую в пределах рабочего диапазона со следующими характеристиками напряжения: наклон 0,07625 и пересечение 2,4742. Далее необходимо осуществить преобразование в аналого-цифровые отсчеты. 12-разрядный десятивольтовый цифро-аналоговый преобразователь обеспечивает 4096 импульсов счета в диапазоне 10 В, или 409,6 имп/В. Умножение значений наклона и пересечения на данное значение позволяет получить наклон импульсов счета 312,32 и пересечение импульсов счета 1013,43. Следует обратить внимание, что пересечение показывают в градусах Цельсия, так что оно должно быть преобразовано в градусы Кельвина, чтобы показать соответствующие единицы СИ. Поскольку необходимо, чтобы выходной сигнал был в градусах Цельсия, требуется его корректировка в разделе единиц измерения СИ.

Можно использовать упрощенную форму калибровки, описанную в 8.6.3.10.

ЭТДП калибровки для данного ИМП будет содержать следующие поля, описанные ниже.

О.3.1 Идентификация ЭТДП калибровки

В таблице О.8 приведено содержание поля идентификации (из таблицы 41).

Таблица О.8 — Идентификатор ЭТДП калибровки

Поле	Содержание	Функция
Тип	03	Поле типа для идентификатора ЭТДП
Длина	04	Данное поле всегда устанавливается равным 04, указывая на то, что поле «Значение» содержит 4 байта
Семейство	00	В данном поле указывается стандарт комплекса стандартов ИИЭР 1451, который определяет данную ЭТДП
Класс	05	В данном поле указывается ЭТДП, к которой осуществляется доступ. Для ЭТДП калибровки данное значение — 05 (указано в таблице 17)
Версия	01	В данном поле указывается версия ЭТДП. Значение представляет собой номер версии, определенный в настоящем стандарте. Значение 01 указывает на то, что данная ЭТДП согласована с первым выпуском настоящего стандарта
Длина кортежа	01	В данном поле указывается число байтов в поле «Длина» всех кортежей в ЭТДП, за исключением данного кортежа

Преобразуя содержание полей в байтовый формат TLV, получаем окончательную последовательность байтов в следующем виде:

03 04 00 05 01 01.

О.3.2 Дата последней калибровки

Данный ИМП последний раз был откалиброван в процессе изготовления, так что дата калибровки совпадает с датой изготовления. Таким образом, дата калибровки — 14 августа 2005 г., а время — 14:00:00. Представление данной информации в формате времени, описанном в 4.9, позволяет получить общее количество секунд с 1 января 1970 г.: 1124114400.

Преобразуя данное значение в шестнадцатеричную систему счисления и используя формат TimeInstance, получаем значение 43 00 9F E0 00 00 00 00.

Значение заголовка поля — 10, а длина — 8, поэтому получаем окончательную последовательность байтов в следующем виде:

0A 04 43 00 9F E0 00 00 00 00.

О.3.3 Калибровочный интервал

Данный ИМП имеет калибровочный интервал 1 год. Преобразуя его в секунды, получаем значение 365 × 24 × 60 × 60 = 31 536 000 или байтовое значение 01 E1 33 80. Подставляя это значение в формате TimeDuration, получаем значение 01 E1 33 80 00 00 00 00.

Значение заголовка поля — 10, а длина — 8, поэтому получаем окончательную последовательность байтов в следующем виде:

0B 08 01 E1 33 80 00 00 00 00.

О.3.4 Постоянные преобразования единиц СИ

Поскольку данные калибровки выводятся в градусах Цельсия, то мы должны обеспечить коэффициент преобразования единиц СИ в градусы Кельвина. Градусы Кельвина = градусы Цельсия + 273,15. Таким образом, получаем множитель 1, а смещение 273,15.

О.3.4.1 Наклон

Поскольку значение наклона 1, данное поле можно опустить и использовать значение по умолчанию.

О.3.4.2 Пересечение

Значение пересечения равно 273,15. Значение кода поля — 31, а тип поля — F32, поэтому получаем окончательный формат байтов данного поля в следующем виде:

1F 04 43 88 93 33.

О.3.4.3 Окончательный формат преобразования единиц СИ

Код поля для комбинированного поля равен 12, а общая длина — 6, поэтому получаем окончательный формат байтов в следующем виде:

0C 06 1F 04 43 88 93 33.

О.3.5 Верхняя и нижняя границы и погрешность

ИМП будет использовать аналогичные поля, описанные в мета-ЭТДП. В связи с этим данные поля опущены.

О.3.6 Действие до и после преобразования

Применение каких-либо операций до или после преобразования не требуется. Поэтому данные поля опущены.

О.3.7 Метод линейного преобразования

Данная ЭТДП будет использовать линейный метод преобразования, который представлен кодом поля 20. Следовательно, ЭТДП не будет включать в себя коды полей 21, 22 или любого из их подполей. Поэтому необходимо указать коэффициенты, приведенные в 8.6.3.24, но опустить сведения как о канале данных, так и о ключе канала, поскольку есть только один канал, который возвращает необработанные аналого-цифровые отсчеты данных.

О.3.7.1 Набор коэффициентов

Данная ЭТДП будет использовать единственный набор коэффициентов, представляющих наклон и пересечение, как описано в 8.6.3.24. Формат каждого из данных значений — F32, и пара также представляет массив F32. Первое значение, то есть значение наклона, составляет 312,32. Преобразование к F32 позволяет получить значение байтов 43 9C 28 F6. Значение пересечения составляет 1013,43. Преобразование к F32 позволяет получить значение байтов 44 7D 5B 85.

Тип поля — 51, а суммарная длина равна 8, поэтому получаем окончательную последовательность байтов в следующем виде:

33 08 43 9C 28 F6 44 7D 5B 85.

О.3.7.2 Окончательный байтовый формат линейного преобразования

Код поля — 20, а общая длина подполя — 10, что позволяет получить окончательный формат байтов:

14 0A 33 08 43 9C 28 F6 44 7D 5B 85.

О.3.8 Окончательный формат ЭТДП калибровки

В связи с тем, что другие поля не применяются к простой линейной регрессирующей калибровке, они опущены. Включенные поля представлены ниже.

```
03 04 00 05 01 01 (заголовок)
0A 08 43 00 9F E0 00 00 00 00 (дата последней калибровки)
0B 08 01 E1 33 80 00 00 00 00 (калибровочный интервал)
0C 06 1F 04 43 88 93 33 (преобразование единиц СИ)
14 0A 33 08 43 9C 28 F6 44 7D 5B 85 (линейное преобразование)
```

Общая длина байтов составляет 48, включая 2 байта контрольной суммы. Контрольная сумма составляет F688. Поэтому окончательный формат байтов ЭТДП калибровки будет иметь вид:

```
00 00 00 30 03 04 00 05 01 01
0A 08 43 00 9F E0 00 00 00 00
0B 08 01 E1 33 80 00 00 00 00
0C 06 1F 04 43 88 93 33
14 0A 33 08 43 9C 28 F6 44 7D
5B 85 F6 88.
```

О.4 ЭТДП с именем преобразователя, заданным пользователем

ЭТДП с именем преобразователя, заданным пользователем, является последней обязательной ЭТДП.

О.4.1 Идентификация ЭТДП с именем преобразователя, заданным пользователем

В таблице О.9 приведено содержание полей идентификации (из таблицы 41).

Таблица О.9 — Идентификатор ЭТДП с именем преобразователя, заданным пользователем

Поле	Содержание	Функция
Тип	03	Поле типа для идентификатора ЭТДП
Длина	04	Данное поле всегда устанавливается равным 04, указывая на то, что поле «Значение» содержит 4 байта
Семейство	00	В данном поле указывается стандарт комплекса стандартов ИИЭР 1451, который определяет данную ЭТДП
Класс	12	В данном поле указывается ЭТДП, к которой осуществляется доступ. Для ЭТДП с именем преобразователя, заданным пользователем, это значение 12 (указано в таблице 17)
Версия	01	В данном поле указывается версия ЭТДП. Значение представляет собой номер версии, определенный в настоящем стандарте. Значение 01 указывает на то, что данная ЭТДП согласована с первым выпуском настоящего стандарта
Длина кортежа	01	В данном поле указывается число байтов в поле «Длина» всех кортежей в ЭТДП, за исключением данного кортежа

Преобразуя содержание полей в байтовый формат TLV, получаем следующую последовательность байтов:
03 04 00 0C 01 01.

О.4.2 Формат

Чтобы упростить реализацию, данный преобразователь будет использовать текстовое поле, определенное пользователем. Таким образом, значение данного поля будет 0. Код поля — 4, поэтому окончательный формат байтов для данного поля имеет вид:

04 01 00.

О.4.3 Поле «TCName» («Имя канала преобразователя»)

Поле TCName будет содержать номер модели данного ИМП. В данном примере номер модели будет «ACME-100». Поскольку содержимое данного поля зависит от пользователя, то для него нет TLV разметки, а последовательность байтов для данного поля является просто значениями символов ASCII:

41 43 4D 45 2D 31 30 30.

О.4.4 Окончательный формат ЭТДП с именем преобразователя, заданным пользователем

Объединенные поля имеют следующий вид:

```
03 04 00 0C 01 01 (заголовок)
04 01 00 (формат)
41 43 4D 45 2D 31 30 30 (TCName).
```

Общая длина этой ЭТДП составляет 19 байтов, включая 2 байта информации о контрольной сумме. Контрольная сумма — FD FE, таким образом, окончательная последовательность байтов для данной ЭТДП:

```
00 00 00 13 03 04 00 0C 01 01
04 01 00 41 43 4D 45 2D 31 30
30 FD FE.
```

О.5 Обязательные команды

Данный ИМП реализует минимальный набор команд, обязательных для работы в качестве действительного датчика ИИЭР 1451.

О.5.1 Общие команды для ИМП и канала преобразователя

В таблице О.10 приведены обязательные команды для выполнения, как показано в таблице 16 (таблица О.10 воспроизводит бóльшую часть таблицы 16).

Таблица О.10 — Обязательные команды

Функция	Команда	Состояние		Ответ ожидается	Обязательная/ необязательная
		Канала преобразователя	ИМП		
0	Зарезервировано	—	—	—	—
1	Запросить ЭТДП	Любое	Активен	Да	Обязательная
2	Считать сегмент ЭТДП	Любое	Активен	Да	Обязательная

Окончание таблицы О.10

Функция	Команда	Состояние		Ответ ожидается	Обязательная/ необязательная
		Канала преобразователя	ИМП		
3	Записать сегмент ЭТДП	Любое	Активен	Нет	Обязательная
4	Обновить ЭТДП	Любое	Активен	Да	Обязательная
5	Запустить самодиагностику	Режим ожидания	Активен	Нет	Обязательная
6	Записать маску сервисного запроса	Любое	Активен	Нет	Обязательная
7	Считать маску сервисного запроса	Любое	Активен	Да	Обязательная
8	Считать регистр состояния-события	Любое	Активен	Да	Обязательная
9	Считать регистр состояния-условия	Любое	Активен	Да	Обязательная
10	Очистить регистр состояния-события	Любое	Активен	Нет	Обязательная
11	Записать состояние протокола состояния-события	Режим ожидания	Активен	Нет	Обязательная
12	Считать состояние протокола состояния-события	Любое	Активен	Да	Обязательная

О.5.1.1 Команда «Query TEDS» («Запросить ЭТДП»)

Данная команда возвращает статус заданной ЭТДП. Единственный параметр данной команды содержит идентификатор ЭТДП. Данный ИМП будет принимать значения 1, 3, 5, 12 и 13. Если получено любое другое значение, то команда возвращает нули для всех полей, кроме поля атрибута. Поле атрибута возвращается в виде UInt8 с установленными битами 1 и 2 и очищенными остальными битами.

Запрос мета-ЭТДП, ЭТДП физического уровня и ЭТДП канала преобразователя должен возвращать атрибуты с установленным битом «только для чтения». ЭТДП калибровки и ЭТДП с именем преобразователя, заданным пользователем, должны возвращать атрибуты, для которых бит «только для чтения» не установлен. Все остальные поля будут реализованы так, как описано в 7.1.1.1. Следует обратить внимание, что при адресации к ЭТДП канала преобразователя и ЭТДП калибровки будет доступен только канал 1, а для всех остальных допустимых ЭТДП потребуется канал 0.

О.5.1.2 Команды «Read/write/update TEDS segment» («Считать/записать/обновить сегмент ЭТДП»)

Данные команды реализованы в соответствии с описанием в 7.1.1.2, 7.1.1.3 и 7.1.1.4.

Аналогично предыдущему подпункту единственные допустимые номера каналов преобразователя — 1 для ЭТДП калибровки и ЭТДП канала преобразователя и 0 для остальных допустимых ЭТДП.

О.5.1.3 Команда «Run self-test» («Запустить самодиагностику»)

Данная команда будет принимать только канал 0 для проверки всего ИМП. Проверка будет состоять из проверки устройств связи, аналого-цифрового преобразователя и тестирования подключения терморезистора.

О.5.1.4 Команды «Read/write service request mask» («Считать/записать маску сервисного запроса»)

Данные команды будут считывать или устанавливать маску сервисного запроса как для всего ИМП, так и для канала 1.

Все биты в маске сервисного запроса могут быть перезаписываемыми в маске. Однако не все биты состояния будут реализованы в регистре событий. Наиболее значительными являются биты, приведенные ниже.

О.5.1.4.1 Бит «Trigger acknowledged» приема сигнала триггера

ИМП не принимает триггерные сигналы, поэтому данный бит никогда не устанавливается ни для канала, ни для ИМП.

О.5.1.4.2 Бит «Missed data or event» («Пропущенные данные или событие»)

Когда команда «Read data» считывания данных направляется в канал 1, получение данных происходит раньше, чем данные будут возвращены. Если до завершения предыдущей операции получена другая команда считывания данных, то бит пропущенных данных будет установлен как для канала, так и для ИМП.

О.5.1.4.3 Нереализованные биты

Следующие биты не применяются к ИМП, поэтому регистр событий будет всегда возвращать 0:

- данные/событие;
- аппаратная ошибка;

- данные доступны/данные обработаны;
- коррекции отключены;
- закончились расходные материалы;
- не первое считывание набора данных.

О.5.1.5 Команды «Read status event/condition register» («Считать регистр состояния события/условия»)

Данные команды реализованы согласно описанию.

О.5.1.6 Команда «Clear status event register» («Очистить регистр состояния-события»)

Данная команда реализована согласно описанию.

О.5.1.7 Команды «Read/write status event protocol state» («Считать/записать состояние протокола состояния-события»)

Данное событие не будет предоставлять потоки состояния, поэтому данные команды не будут реализованы.

О.5.2 Команды для преобразователя в режиме ожидания

В таблице О.11 приведены команды для преобразователя в режиме ожидания, которые реализуются в данном ИМП.

Таблица О.11 — Команды для преобразователя в режиме ожидания

Функция	Команда	Класс адреса			Ответ ожидается	Обязательная/ необязательная
		Канал преобразователя	Прокси	Группа		
3	Установить адресную группу	Да	Да	Нет	Нет	Обязательная
10	Калибровать канал преобразователя	Да	Да	Да	Да	Необязательная

О.5.2.1 Команда «AddressGroup definition» («Установить адресную группу»)

Данная команда позволяет приписать канал 1 указанной адресной группе. Все остальные каналы будут приводить к генерации ошибки.

О.5.2.2 Команда «Calibrate Transducer Channel» («Калибровать канал преобразователя»)

Данная команда будет приводить к калибровке датчика.

О.5.3 Команды для рабочего режима преобразователя

В таблице О.12 приведены команды, которые преобразователь будет выполнять в рабочем режиме.

Таблица О.12 — Команды для рабочего режима преобразователя

Функция	Команда	Класс адреса			Ответ ожидается	Обязательная/ необязательная
		Канал преобразователя	Прокси	Группа		
1	Считать сегмент набора данных канала преобразователя	Да	Да	Нет	Да	См. О.5.3.11[1)]
3	Запустить триггер	Да	Да	Да	Нет	Обязательная

О.5.3.1 Команда «Read Transducer Channel data set segment» («Считать сегмент набора данных канала преобразователя»)

Параметры для данной команды будут ограничены значением 0 для смещения и значением 1 для длины. Все остальные значения будут приводить к генерации ошибки команды. При получении данной команды канал будет собирать данные от аналого-цифрового преобразователя и возвращать данные в ответ на команду.

О.5.3.2 Команда «Trigger» («Запустить триггер»)

Данная команда является обязательной, но она будет рассматриваться как «нет операции» («пустая команда») (НОП).

[1)] В оригинале ISO/IEC/IEEE 21450:2010 допущена ошибка. Ошибочно приведена ссылка на примечание («См. примечание»).

О.5.4 Общие команды для режима ожидания и рабочего режима преобразователя

Команды, перечисленные в таблице О.13, будут реализованы в соответствии с описанием в 7.1.4, за исключением команды «Read Transducer Channel trigger state» («Считать состояние триггера канала преобразователя»).

Таблица О.13 — Общие команды для режима ожидания и рабочего режима преобразователя

Функция	Команда	Класс адреса			Ответ ожидается	Обязательная/ необязательная
		Канал преобразователя	Прокси	Группа		
1	Рабочий режим канала преобразователя	Да	Да	Да	Нет	Обязательная
3	Режим ожидания канала преобразователя	Да	Да	Да	Нет	Обязательная
4	Считать состояние триггера канала преобразователя	Да	Да	Да	Нет	Обязательная
7	Считать привязку к адресной группе	Да	Да	Нет	Да	Обязательная

О.5.4.1 Команда «Read Transducer Channel trigger state» («Считать состояние тригера канала преобразователя»)

Данная команда всегда возвращает значение «False» («Ложь»).

О.5.5 Команды для спящего режима ИМП

Команда выхода из спящего режима, приведенная в таблице О.14, будет выполнена в соответствии с описанием в 7.1.5.1.

Таблица О.14 — Команды для спящего режима ИМП

Функция	Команда	Класс адреса		Ответ ожидается	Обязательная/ необязательная
		ИМП	Глобальный		
1	Выйти из спящего режима	Да	Нет	Нет	Необязательная

О.5.6 Команды для активного режима ИМП

Команды для активного режима, приведенные в таблице О.15, будут реализованы в соответствии с описанием в 7.1.6.

Таблица О.15 — Команды для активного режима ИМП

Функция	Команда	Класс адреса		Ответ ожидается	Обязательная/ необязательная
		ИМП	Глобальный		
1	Считать версию ИМП	Да	Нет	Да	Обязательная
2	Перейти в спящий режим ИМП	Да	Нет	Нет	Необязательная
3	Сохранить рабочие настройки	Да	Нет	Нет	Обязательная
4	Восстановить рабочие настройки	Да	Нет	Нет	Обязательная
5	Считать версию ИИЭР 1451.0	Да	Нет	Да	Обязательная

О.5.6.1 Команда «Read TIM version» («Считать версию ИМП»)

Данная команда всегда возвращает значение 1 для первого выпуска ИМП.

О.5.6.2 Команда «TIM sleep» («Перейти в спящий режим ИМП»)

Данная команда реализована в соответствии с описанием в 7.1.6.2. Так как это беспроводной ИМП, то выключение питания имеет важное значение для сохранения заряда батареи.

О.5.6.3 Команда «Store state» («Сохранить состояние»)

Данная команда реализована в соответствии с описанием в 7.1.6.3. Существует только одно состояние, помеченное как состояние 0. Для данного ИМП нет устанавливаемых параметров, поэтому команды «сохранить состояние» и «сбросить состояние» аналогичны, и данная операция рассматривается как НОП.

О.5.6.4 Команда «Recall state» («Восстановить состояние»)

Данная команда реализована в соответствии с описанием в 7.1.6.4. Существует только одно состояние, помеченное как состояние 0. Для данного ИМП нет устанавливаемых параметров, поэтому команды «сохранить состояние» и «сбросить состояние» аналогичны, и данная операция рассматривается как НОП.

О.5.6.5 Команда «Read the IEEE 1451.0 version» («Считать версию ИИЭР 1451.0»)

Данная команда реализована в соответствии с описанием в 7.1.6.5. Команда всегда будет возвращать 1.

О.5.7 Команды для любого режима ИМП

Перечисленные ниже команды могут выполняться, когда ИМП находится в любом режиме. В таблице О.16 приведены соответствующие части таблицы 38, в которых перечислены команды данного класса.

Для всех команд данного класса требуется нулевое значение номера канала-получателя преобразователя. Если номер канала-получателя преобразователя в байтовом массиве не равен нулю, то команда должна быть проигнорирована, а в регистре состояния-условия канала преобразователя (см. 5.13) должен быть установлен бит «Command rejected» («Отказ от выполнения команды»).

Таблица О.16 — Команды для любого режима ИМП

Функция	Команда	Класс адреса		Ответ ожидается	Обязательная/ необязательная
		ИМП	Глобальный		
1	Перезагрузка	Да	Нет	Нет	Необязательная

О.5.7.1 Команда «Reset» («Перезагрузка»)

Данная команда реализована и приводит к вызову последовательности процедур инициализации.

Приложение Р
(справочное)

Список участников ИИЭР

Участники

На момент представления данного стандарта на рассмотрение Комиссии по стандартам Ассоциации стандартов ИИЭР общее руководство и рабочая группа по ЭТДП имели следующий состав:

Кэнг Ли (Kang Lee), председатель,

Виктория Свитсер (Victoria K. Sweetser), заместитель председателя,

Джей Дж. Неметс-Иоханес (Jay J. Nemeth-Johannes), секретарь,

Ли Эклес (Lee H. Eccles), редактор,

Джеферсон Бюрч (Jefferson B. Burch), разработчик API интерфейсов,

Пэт Блейкли (Pat Blakely), Мурат Бог (Murat Bog), Чарльз Джонс (Charles H. Jones), Эдвард Кох (Edward Koch), Дэн Максвел (Dan Maxwell), Дэвид Перрасел (David B. Perrussel), Эндрю Сигал (Andrew Segal), Меликарьюн Шанкар (Mallikarjun Shankar), Роберт Синклер (Robert Sinclair), Евгений Сонг (Eugene Song), Рич Вальд (Rich Valde), Ларри Уипл (Larry Whippl), Джеймс Визер (James J. Wiczer), Дэрольд Вобсхол (Darold Wobschall).

Следующие люди внесли вклад в разработку настоящего стандарта:

Терстон Брукс (Thurston Brooks), Вэйн Кетлин (Wayne Catlin), Баожи Чен (Baozhi Chen), Стефан Чен (Stephen Chen), Раян Колеман (Ryan Coleman), Кен Корнет (Ken Cornett), Фил Элерброк (Phil Ellerbrock), Питер Флитнер (Peter Flitner), Фернандо Генкуонг (Fernando GenKuong), Роберт Джонсон (Robert N. Johnson), Элис Ло (Alice Law), Миунг Ли (Myung Lee), Ларри Мэлходи (Larry Malchodi), Лорен Риттл (Loren Rittle), Джей Ворриэр (Jay Warrior), Стэн Вудс (Stan Woods).

Голосование по данному стандарту проводилось с использованием индивидуальных бюллетеней с полями «за», «против» и «воздержался». Состав комитета по голосованию:

Вильям Акерман (William J. Ackerman), Крис Бейдж (Chris B. Bagge), Джеферсон Бюрч (Jefferson B. Burch), Хуан Кареон (Juan C. Carreon), Данила Чернецов (Danila Chernetsov), Кейт Чоу (Keith Chow), Томи Купер (Tommy P. Cooper), Мэтью Дэвис (Matthew T. Davis), Дэвид Дрост (David B. Droste), Ли Эклес (Lee H. Eccles), Майкл Гейпл (Michael D. Geipel), Фернандо ГенКуонг (Fernando Genkuong), Серджио Гома (Sergiu R. Goma), Пэтрик Гониа (Patrick S. Gonia), Рэндел Грувс (Randall C. Groves), Рюсэк Хасегава (Ryusuke Hasegawa), Вернер Хоэзл (Werner Hoelzl), Деннис Хорвитс (Dennis Horwitz), Дженс Халт (Jens Hult), Роберт Джонсон (Robert N. Johnson), Чарльз Джонс (Charles H. Jones), Иннокент Камва (Innocent Kamwa), Петр Кароки (Piotr Karocki), Джеймс Кемерлинг (James C. Kemerling), Эдвард Кох (Edward Koch), Чарльз Леннон-младший (Charles A. Lennon, Jr.), Вильям Лампкинс (William Lumpkins), Дж. Лари (G. L. Luri), Джозеф Маршал-младший (Joseph R. Marshall, Jr.), Гарри Мичел (Gary L. Michel), Ингуа Мин (Yinghua Min), Джорджес Монтиле (Georges F. Montillet), Джерри Мерфи (Jerry R. Murphy), Джей Дж. Неметс-Иоханес (Jay J. Nemeth-Johannes), Майкл Ньюман (Michael S. Newman), Крис Остерло (Chris L. Osterloh), Ховард Пенроуз (Howard W. Penrose), Дэвид Перрасел (David B. Perrussel), Викрем Пунж (Vikram Punj), Богдан Зелигер (Bogdan Seliger), Меликарьюн Шанкар (Mallikarjun Shankar), Томас Сип (Thomas M. Siep), Мэтью Смит (Matthew L. Smith), Виктория Свитсер (Victoria K. Sweetser), Лерой Тильман (Leroy O. Thielman), Стефан Веб (Stephen C. Webb), Джеймс Визер (James J. Wiczer), Эрнесто Джордж Виденбруг (Ernesto Jorge Wiedenbrug), Дарольд Вобсхол (Darold Wobschall), Дерек Ву (Derek T. Woo), Эрик Вудс (Eric V. Woods), Орен Йен (Oren Yuen), Янус Залевский (Janusz Zalewski).

Состав Комиссии по стандартам Ассоциации стандартов ИИЭР 22 марта 2007 г.:

Стив Милс (Steve M. Mills), председатель,

Роберт Гроу (Robert M. Grow), заместитель председателя,

Дональд Райт (Donald F. Wright), экс-председатель,

Джудит Горман (Judith Gorman), секретарь,

Ричард Дебласио (Richard DeBlasio), Алекс Гельман (Alex Gelman), Вильям Голдбах (William R. Goldbach), Арнольд Гринспан (Arnold M. Greenspan), Джоанна Гуенин (Joanna N. Guenin), Джулиан Фостер (Julian Forster), Кенет Ханус (Kenneth S. Hanus), Вильям Хопф (William B. Hopf), Ричард Халет (Richard H. Hulett), Герман Кох (Hermann Koch), Джозеф Копфингер (Joseph L. Koepfinger)[1)], Джон Кулик (John D. Kulick), Дэвид Ло (David J. Law), Глен Парсонс (Glenn Parsons), Рональд Питерсон (Ronald C. Petersen), Том Превост (Tom A. Prevost), Нараянан Рамашандран (Narayanan Ramachandran), Грег Ратта (Greg Ratta), Роби Робсон (Robby Robson), Анна-Мария Сахазизиан (Anne-Marie Sahazizian), Вирджиния Сальцбергер (Virginia C. Sulzberger), Малкольм Таден (Malcolm V. Thaden), Ричард Таунсенд (Richard L. Townsend), Ховард Вольфман (Howard L. Wolfman).

Следующие лица, не принимавшие участие в голосовании, являются ответственными по взаимодействию Комиссии по стандартам Ассоциации стандартов ИИЭР:

Сэтиш Аггарвэл (Satish K. Aggarwal), представитель NRC,

Алан Куксон (Alan H. Cookson), представитель NIST,

Дженни Штайнхаген (Jennie M. Steinhagen), руководитель программ стандартов ИИЭР, специалист по разработке документации,

Норма Дэвис (Norma B. Davis), руководитель программ стандартов ИИЭР, специалист по разработке технических программ.

[1)] Почетный член.

Приложение ДА
(справочное)

Сведения о соответствии ссылочных международных стандартов национальным стандартам Российской Федерации

Таблица ДА.1

Обозначение ссылочного международного стандарта	Степень соответствия	Обозначение и наименование соответствующего национального стандарта
АНСИ X3.4—1986	NEQ	ГОСТ 27463—87 «Системы обработки информации. 7-битные кодированные наборы символов»
Extensible Markup Language (XML) 1.0 (Second Edition)	—	*
RFC 2616	—	*
ИИЭР 754—1985	—	*
ИИЭР 802.3—2002	—	*
ИИЭР 1451.1—1999	—	*
ИИЭР 1451.2—1997	—	*
ИИЭР 1451.3—2003	—	*
ИИЭР 1451.4—2004	—	*
ИИЭР 1588—2002	—	*
ИСО 639:1988	NEQ	ГОСТ 7.75—97 «Система стандартов по информации, библиотечному и издательскому делу. Коды наименований языков»
ИСО 19136	—	*
ИСО/МЭК 14750:1999	—	*

* Соответствующий национальный стандарт отсутствует. До его утверждения рекомендуется использовать перевод на русский язык данного международного стандарта. Перевод данного международного стандарта находится в Федеральном информационном фонде технических регламентов и стандартов.

Примечание — В настоящей таблице использовано следующее условное обозначение степени соответствия стандартов:

- NEQ — неэквивалентные стандарты.

УДК 004.93'1:006.89:006.354 ОКС 35.040 П85

Ключевые слова: информационные технологии, интерфейс интеллектуального преобразователя для датчиков и исполнительных устройств, датчики, исполнительные устройства, общие функции, протоколы взаимодействия, форматы электронной таблицы данных преобразователя, электронная таблица данных преобразователя

Редактор *А.С. Бубнов*
Корректор *Е.Р. Ароян*
Компьютерная верстка *Ю.В. Половой*

Сдано в набор 11.06.2016. Подписано в печать 12.07.2016. Формат 60 × $84^{1}/_{8}$. Гарнитура Ариал.
Усл. печ. л. 33,02.

Набрано в ИД «Юриспруденция», 115419, Москва, ул. Орджоникидзе, 11.
www.jurisizdat.ru y-book@mail.ru

Издано во ФГУП «СТАНДАРТИНФОРМ», 123995 Москва, Гранатный пер., 4.
www.gostinfo.ru info@gostinfo.ru