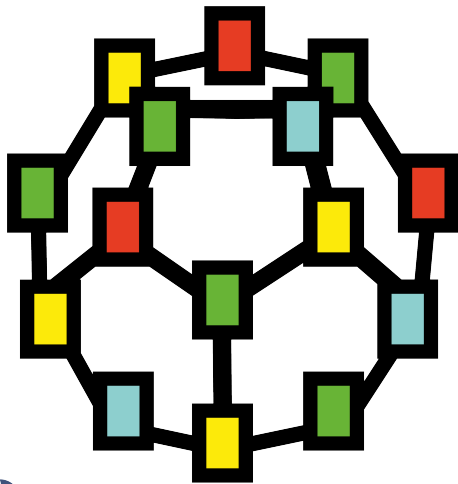




# Библиотека раІЕС104



Руководство пользователя

09.2024  
версия 1.2

---

# Содержание

<b>Используемые термины и сокращения</b> .....	<b>3</b>
<b>Введение</b> .....	<b>4</b>
<b>1 Общие сведения</b> .....	<b>5</b>
1.1 Основные сведения о стандартах МЭК 60870-5 .....	5
1.2 Структура ASDU .....	6
<b>2 Библиотека raIEC104</b> .....	<b>9</b>
2.1 Реализация протокола МЭК 60870-5-101 (IEC101uni).....	10
2.2 Реализация протокола МЭК 60870-5-104 (IEC104uni).....	13
2.3 Реализация протокола МЭК 60870-5-104. Сервер (IEC104Server).....	16
2.4 Входной буфер (IECBufIn).....	20
2.5 Выходной буфер (IECBufOut).....	21
2.6 Разбиение структуры типа ASDU M_EP_* (IECEPFromInt).....	24
2.7 Формирование структуры типа ASDU M_EP_* (IECIntFromEP).....	24
2.8 Расшифровка описателя качества типа ASDU M_IT_* (IECITSQIn) .....	25
2.9 Формирование описателя качества типа ASDU M_IT_* (IECITSQOut) .....	25
2.10 Расшифровка временных меток (IECTransTime) .....	26
2.11 Получение отладочной информации (IECInfo).....	27
<b>3 Примеры настройки обмена по протоколу МЭК 60870-5-104</b> .....	<b>29</b>
3.1 Настройка обмена в режиме сервера по протоколу 5-104 .....	29
3.1.1 Передача информации о процессе в направлении контроля (M) .....	29
3.1.2 Прием информации о процессе в направлении управления (C).....	36
3.1.3 Передача упакованных данных (M_EP_*) .....	38
3.1.4 Передача интегральных сумм (M_IT_*).....	40
3.2 Настройка обмена в режиме клиента протокола 5-104 .....	41
3.2.1 Прием информации о процессе в направлении контроля (M) .....	42
3.2.2 Передача информации о процессе в направлении управления (C).....	48
3.2.3 Прием упакованных данных (M_EP_*) .....	54
3.2.4 Прием интегральных сумм (M_IT_*).....	56
<b>4 Примеры настройки обмена по протоколу МЭК 60870-5-101</b> .....	<b>58</b>
4.1 Настройка обмена в режиме сервера протокола 5-101 .....	58
4.2 Настройка обмена в режиме клиента протокола 5-101 .....	59
<b>5 Отладка и диагностика обмена</b> .....	<b>61</b>
<b>ПРИЛОЖЕНИЕ А. Описание типов информации ASDU</b> .....	<b>64</b>
<b>ПРИЛОЖЕНИЕ Б. Причины передачи (COT)</b> .....	<b>67</b>

---

## Используемые термины и сокращения

**Балансная передача** – режим передачи, при котором каждая из двух связанных станций (только в канале топологии точка-точка) является **комбинированной** и может начинать передачу сообщения в любой момент времени.

**Идентификатор типа** – однобайтное поле, определяющее структуру, тип и формат информационного объекта. Все информационные объекты одного блока данных прикладного уровня (**ASDU**) имеют одинаковую структуру, тип и формат.

**Комбинированная станция** – станция, которая может выступать в качестве **контролирующей и контролируемой**.

**Контролируемая станция (Controlled Station, Outstation, Slave, Server)** – станция, которая передает информацию только по запросу контролирующей станции. Далее в документе – «сервер» или «контролируемая станция».

**Контролирующая станция (Controlling Station, Master, Client)** – станция, которая инициирует процедуру обмена информацией. Далее в документе – «клиент» или «контролирующая станция».

**Небалансная передача** – режим передачи, при котором только одна станция – **контролирующая** – может начинать передачу информации, а другая станция (все остальные станции при многоточечной топологии) – **контролируемая** – передает информацию только после запроса контролирующей станции.

**Объект информации** или **информационный объект** – группа данных вместе с ее идентификатором (адресом, наименованием).

**Описатель качества** – набор атрибутов, обеспечивающих дополнительной информацией о качестве объекта информации при передаче данных.

**Периодическая передача** (в Полигон это **циклическая передача**) – передача наборов данных, повторяющаяся через заданные промежутки времени.

**ПЛК** – программируемый логический контроллер.

**Причина передачи** – однобайтное или двухбайтное поле, которое прилагается к **ASDU** для пояснения источника, инициирующего передачу данных в канал.

**Спорадическая передача** (в Полигон это **спонтанная передача**) – передача данных, инициируемая процессом пользователя при возникновении событий или изменений данных.

**ТИТ** – телеизмерение.

**ТС** – телесигнализация.

**ТУ** – телеуправление.

**Фоновое сканирование** – передача не изменившихся **ТС** и **ТИТ** в свободное время канала.

**Элемент информации** или **информационный элемент** – неделимая переменная, например, значение измеряемой величины или данные двухпозиционного состояния.

**ASDU (Application Service Data Unit)** – блок данных, обслуживаемый прикладным уровнем протокола.

**EPA (Enhanced Performance Architecture)** – архитектура с повышенной производительностью. Вместо классической семиуровневой **модели OSI** архитектура EPA использует только три уровня: прикладной, канальный и физический, чтобы уменьшить время реакции при ограниченной скорости передачи.

**OPC UA (Open Platform Communications, Unified Architecture)** – промышленный протокол для обмена данными с ПЛК и для управления ими.. Спецификация OPC UA разработана организацией OPC Foundation.

**SQL (Structured Query Language)** – язык программирования для хранения и обработки информации в реляционной базе данных.

**Unix-время** – количество секунд, прошедших с полуночи (00:00:00 UTC) 1 января 1970 года (четверг).

---

## Введение

Настоящее руководство описывает настройку обмена данными по протоколам **МЭК 60870-5-101** и **МЭК 60870-5-104** для контроллеров ОВЕН, программируемых в среде **Полигон**. Предполагается, что читатель обладает базовыми навыками работы с **Полигон**, поэтому общие вопросы (например, создание и загрузка проектов) в данном документе не рассматриваются – они подробно описаны в документах [Руководство по программированию](#), [Библиотека raCore](#) и [Быстрый старт](#).

Настройка обмена данными по протоколам **МЭК 60870-5-101** и **МЭК 60870-5-104** в среде **Полигон** осуществляется с помощью функциональных блоков из библиотеки **paIEC104**. Данная библиотека доступна для работы при наличии соответствующей лицензии runtime (см. описание лицензионных пакетов [на странице среды программирования Полигон](#)).

Примеры в документе актуальны для версии среды **Полигон – 1994**, версии библиотеки **paIEC104 – 921** и выше.

# 1 Общие сведения

## 1.1 Основные сведения о стандартах МЭК 60870-5

**МЭК 60780** – набор стандартов, определяющий системы, используемые для телеуправления и SCADA в электротехнике и энергетике. **Часть 5** определяет протоколы для контроля и управления с использованием постоянного соединения.

Стандарт разработан техническим комитетом МЭК №57 (рабочая группа 03).

Первая часть стандарта была опубликована в 1990 году и описывала преимущественно нижние уровни сетевой модели. Первое издание протокола **5-101** опубликовано в 1995 году. В 2000 году представлен **5-104** протокол, который во многом схож с последовательным **5-101** протоколом, но основан на применении стека [TCP/IP](#).

Стандарт оптимизирован для применения SCADA-системами: позволяет транслировать данные с удаленных станций на главную, обеспечивает оперативное управление с главной станции удаленными станциями.

Надежная передача данных обеспечивается показателями качества данных, а также контрольной суммой.

Стандарт позволяет оптимизировать трафик сети и использовать более экономичные сети с низкой пропускной способностью.

Дополнительно обеспечиваются такие функции, как временные отметки, блокировка операций, предотвращение несанкционированного доступа.

Существует идентичный российский стандарт ГОСТ Р МЭК 60870-5 «Устройства и системы телемеханики. Часть 5. Протоколы передачи» ([раздел 101](#), [раздел 104](#)).

В **МЭК 60870-5** входят основные стандарты::

- **5-1 Transmission Frame Formats** – описание передаваемых кадров;
- **5-2 Data Link Transmission Services** – описание сервисов канального уровня;
- **5-3 General Structure of Application Data** – общая структура прикладного уровня;
- **5-4 Definition and Coding of Information Elements**– определение и кодирование информации;
- **5-5 Basic Application Functions** – применение сервисов прикладного уровня;
- **5-7 Security extension** – расширение системы безопасности;
- **5-101 Transmission Protocols** – сопутствующий стандарт для базовых задач телеуправления;
- **5-102 Transmission Protocols** – сопутствующий стандарт для передачи интегральных данных измерений (не получил распространения);
- **5-103 Transmission Protocols** – сопутствующий стандарт интерфейса защитного оборудования;
- **5-104 Transmission Protocols** – сопутствующий стандарт для использования стека TCP/IP.

Стандарт оперирует следующими понятиями:

- **Контролирующая станция (Controlling Station, Master, Client)** – станция, инициирующая процедуру обмена информацией;
- **Контролируемая станция (Controlled Station, Outstation, Slave, Server)** – станция, передающая информацию только по запросу контролирующей станции.

Поддерживаемые топологии сети изображены на рисунке ниже.

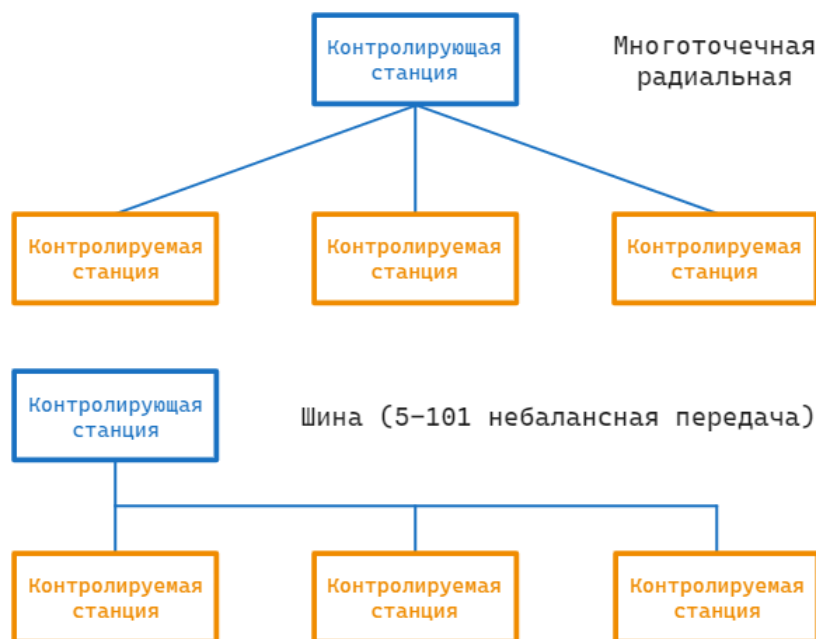


Рисунок 1.1 – Поддерживаемые топологии сети

Таблица 1.1 – Уровни сетевой модели, определяемые стандартом 5-101 (модель EPA)

Уровень	Описание
Прикладной	Блоки данных, события с временными метками, синхронизация времени, управление, опрос (периодический и спорадический)
Канальный	Управление связью, контрольные суммы, адресация, подтверждение связи
Физический	RS-232, RS-485 и др.

Таблица 1.2 – Уровни сетевой модели, определяемые стандартом 5-104

Уровень	Описание
Прикладной	Блоки данных, события с временными метками, синхронизация времени, управление, опрос (периодический и спорадический). Интеграция со стеком TCP/IP
Транспортный и сетевой	TCP/IP
Канальный	RFC 894
Физический	IEEE 802.3 Ethernet

## 1.2 Структура ASDU

На прикладном уровне определяются блоки данных **ASDU (Application Service Data Unit)**.

**ASDU** содержит:

- Тип блока данных;
- Причину передачи;
- Адрес объектов информации;
- Объекты информации.

Таблица 1.3 – Структура ASDU

Название	Размер	Описание
<b>Идентификатор блока данных</b>		
Идентификатор типа (Type Identification – <b>Type ID</b> )	1 байт	Структура, тип и формат данных
Классификатор переменной структуры (Variable Structure Qualifier – <b>VSQ</b> )	1 байт	Структура блока – тип объектов информации (объекты или элементы) и их количество
Причина передачи (Cause Of Transmission – <b>COT</b> )	1...2 байта	Пояснения источника, инициирующего передачу данных

## Продолжение таблицы 1.3

Название	Размер	Описание
Общий адрес станции ( <b>Common Address</b> of ASDUs)	1...2 байта	Уникальный адрес станции в сети
<b>Объекты информации (один или несколько)</b>		
Адрес объекта информации		
Элемент информации (один или несколько)		
Метка времени		

Идентификатор типа **Type ID** составляет первое однобайтовое поле и определяет структуру, тип и формат объекта информации. Идентификатор типа принимает значения:

- **1...44** – статус и измерения;
- **45...99** – управляющие команды;
- **100...109** – команды для мониторинга;
- **110...119** – параметры измеряемых величин;
- **120...127** – передача файла.

При определении типов используют следующие условные обозначения (метки):

- 1-й элемент: **M** – передача в направлении контроля (от сервера к клиенту), **C** – передача в направлении управления (от клиента к серверу), **P** – передача параметров, **F** – передача файлов;
- 2-й элемент – вид информации (две буквы);
- 3-й элемент: **T** – наличие метки времени, **N** – отсутствие метки времени;
- 4-й элемент – формат данных (**A, B, C** и т. д.).

Общее описание типов информации приведено в [Приложении А](#).

Классификатор переменной структуры **VSQ** составляет второе однобайтовое поле и определяет структуру блока, то есть тип информационных компонентов (объекты или элементы) и их количество.

Классификатор переменной структуры состоит из:

- **SQ** (бит 8) – тип объекта информации: **0** – Объект, **1** – элемент;
- **N** (биты 7...1) – количество объектов информации: **1...127**.

Причина передачи **COT** составляет третье однобайтовое поле (опционально – двухбайтовое).

Причина передачи состоит из:

- **Test** (бит 8) – тестовый режим передачи ASDU: **0** – рабочая передача, **1** – тестовая передача;
- **P/N** (бит 7): **0** – положительное подтверждение, **1** – отрицательное подтверждение;
- Причина передачи (биты 6...1): **1...47**.

Причины передачи перечислены в [Приложении Б](#).

**Общий адрес станции** занимает четвертое однобайтовое поле (опционально – двухбайтовое). Широковещательный адрес – **FF** (или **FFFF**).

**Объекты информации** представлены тремя типами:

- Одноэлементная информация – одна команда, событие или измерение;
- Многоэлементная информация – измерение и описатель качества;
- Последовательность элементов информации – серия измерений.

**Элемент информации** представлен четырьмя типами данных:

- Битовый;
- Целый;
- Вещественный;
- Строка битов (**Bitstring**).

В некоторые элементы информации включен **описатель качества**.

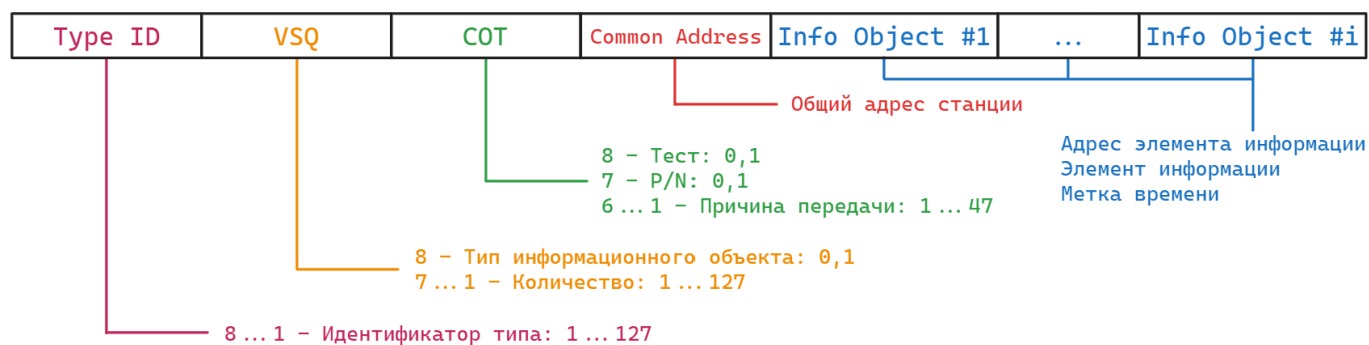


Рисунок 1.2 – Структура ASDU

Пример элемента информации типа **M\_SP\_NA**: одноэлементная информация (1 байт), в младшем разряде передается значение бита **SPI (Single Point Information)**. Описатель качества содержит:

- **IV (Invalid/Valid)**: **0** – действительная, **1** – недействительная;
- **NT (Not topical/Topical)**: **0** – актуальное значение, **1** – неактуальное значение;
- **SB (Substituted/Not substituted)**: **0** – нет замещения, **1** – есть замещение. Значение величины поступает от оператора (**1**) или от автоматического источника (**0**);
- **BL (Blocked/Not blocked)**: **0** – нет блокировки, **1** – есть блокировка;
- **IN (Inversion)** – инверсное значение
- **GN (General)** – обобщенная величина.

7	6	5	4	3	2	1	0
IV	NT	SB	BL	IN	GN	0	SPI
Описатель качества							

Рисунок 1.3 – Пример структуры элемента информации M\_SP\_NA



## 2 Библиотека paIEC104

Библиотека **paIEC104** содержит функциональные блоки для реализации обмена по протоколам **МЭК 60870-5-101** и **МЭК 60870-5-104**.

В структуру библиотеки входят блоки **IEC101uni** и **IEC104uni**, с помощью которых настраивается обмен по протоколам **5-101** (режим клиент) и **5-104** (режим сервер).. Для работы с этими блоками требуется подключить блоки работы с системными интерфейсами – **210-RS485/210-RS485** из библиотеки **paOwenIO** и **TcplpCIA** из библиотеки **paCore**.



### ПРИМЕЧАНИЕ

Для настройки сервера по протоколу **5-104** рекомендуется использовать более новый блок **IEC104Server**. Для его работы не требуется отдельный блок для работы с системным сокетом.

Блоки **IECBufIn** и **IECBufOut** реализуют входной и выходной буфер данных протокола, соответственно.

Остальные блоки библиотеки являются вспомогательными.

Блок **IECInfo** используется для отладочных целей.

Для добавления библиотеки **paIEC104** в проект следует:

1. Перейти в меню **Окна/Проекты**. В появившемся окне отобразится текущий проект и добавленные библиотеки.

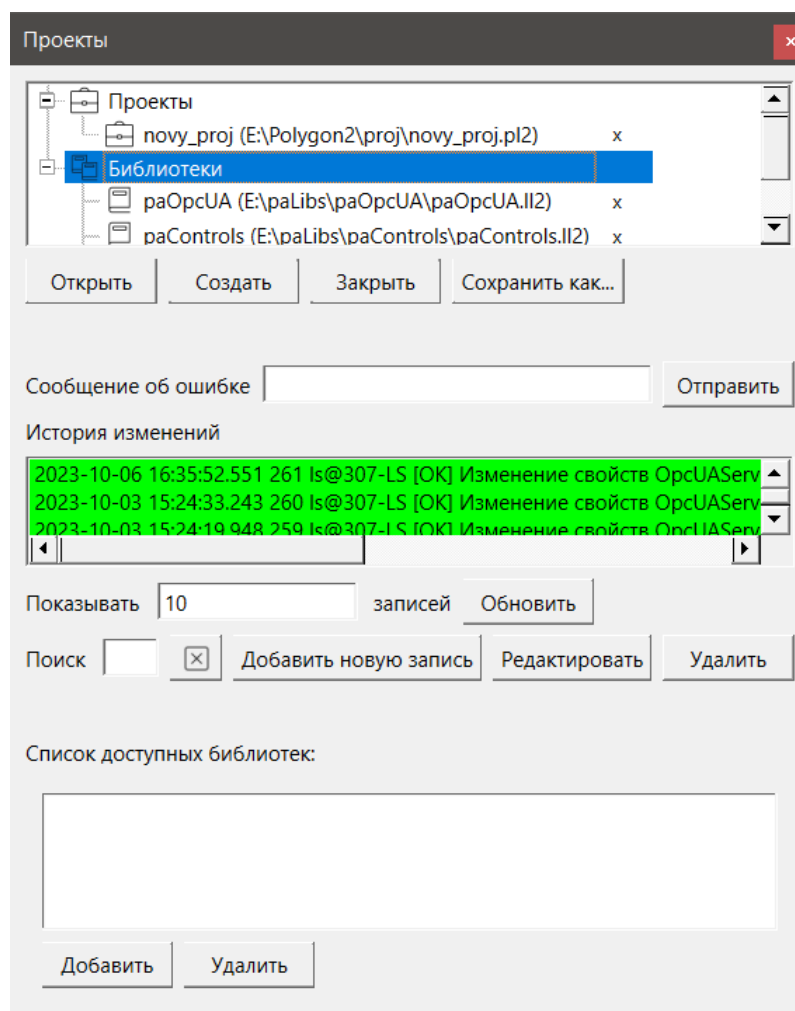


Рисунок 2.1 – Добавление библиотеки paIEC104 в проект

2. Нажать кнопку **Открыть** и перейти в папку с файлами библиотеки, которую необходимо добавить. Затем в выпадающем списке выбрать тип файла **Библиотека Полигон 2 (\*.I12)**.

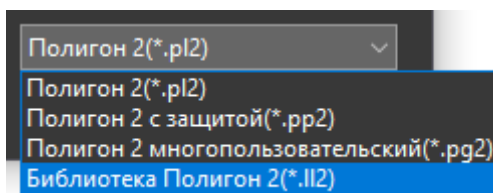


Рисунок 2.2 – Добавление библиотеки paIEC104 в проект

3. В окне появится файл библиотеки с расширением **.ll2**. Следует выбрать его и нажать **Открыть**.

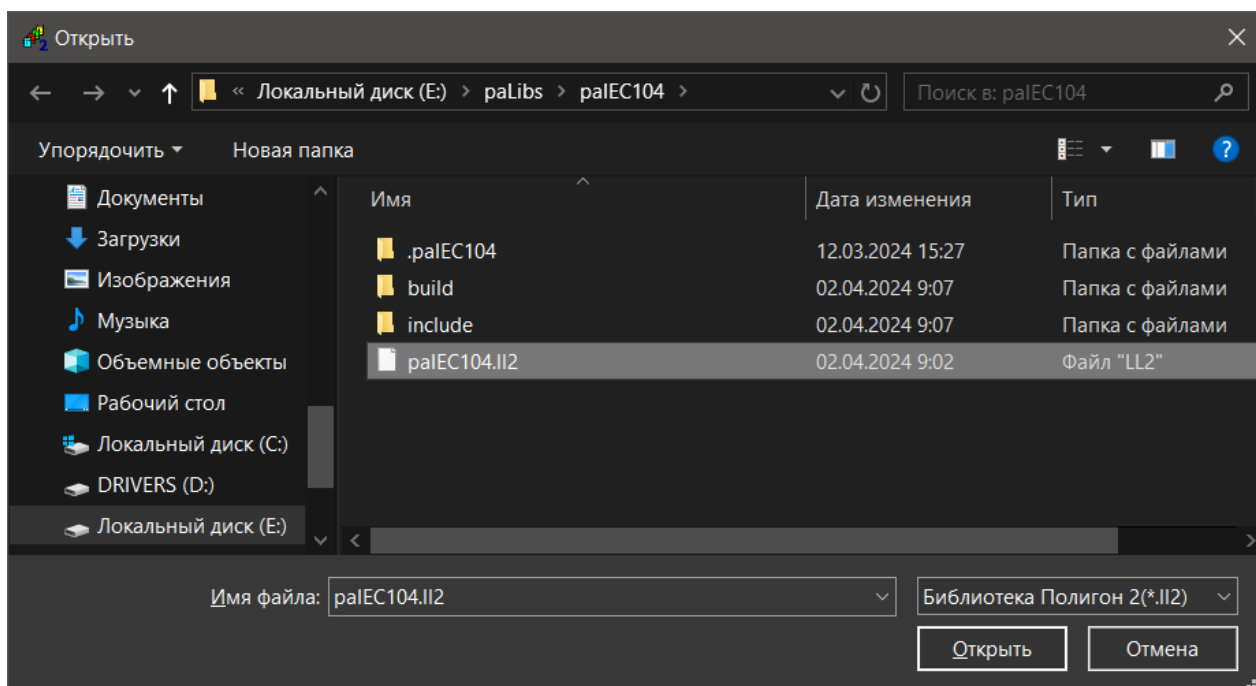


Рисунок 2.3 – Добавление библиотеки paIEC104 в проект

Добавленная библиотека отобразится в окне **Проекты**.

## 2.1 Реализация протокола МЭК 60870-5-101 (IEC101uni)

Блок **IEC101uni** реализует протокол стандарта **МЭК 60870-5-101** и определяет функции контролирующей (клиента) и контролируемой станции (сервера).

Совместно с **IEC101uni** используются специальные буфера – **IECBufIn** для получаемых данных/команд, **IECBufOut** для отправляемых данных/команд. С помощью данных буферов создается база объектов информации, участвующих в обмене по указанному протоколу.

Так как работа блока занимает значительное время, данный блок следует размещать только в **Фоне**.

Таблица 2.1 – Назначение входов и выходов IEC101uni

Элемент	Описание
<b>Входы</b>	
spc	Связь с блоком последовательного интерфейса <b>210-RS485/210-RS232</b> из библиотеки <b>paOwenIO</b>
enbl	Режим работы блока: <b>0</b> – блок отключен; <b>1</b> – блок в работе
ctl	Инициация команды управления. Битовая маска: <b>Бит 1</b> – инициировать команду общего опроса; <b>Бит 7</b> – отсылать диагностические сообщения ( <b>бит 12</b> на входе <b>cfg</b> должен быть установлен до компиляции проекта)
tm1	Не используется

Продолжение таблицы 2.1

Элемент	Описание
tm2	Не используется
dbg	Конфигурация отладки. Битовая маска: <b>Бит 0...7</b> – битовая маска, которая определяет, какие типы сообщений выводить: <b>бит 0</b> – тип 1, ..., <b>бит 7</b> – тип 8 (высший уровень); <b>Бит 8</b> – выводить в консоль; <b>Бит 9</b> – выводить через блок <i>RamLog</i> из библиотеки <i>paCore</i>
bo	Входы для подключения <i>IECBufOut</i> (циклический)
<b>Входы (константные)</b>	
tmp	Период вызова главной функции блока, мс
prm	Тип канального уровня: <b>IEC_MASTER</b> – инициирует соединение; <b>IEC_SLAVE</b> – ожидает соединение
ltp	Режим канального уровня: <b>IEC_BALANCED</b> – балансный; <b>IEC_UNBALANCED</b> – небалансный
oa	Не используется
lka	Адрес станции в сети, если используется ( <b>Link Address</b> )
las	Длина поля адреса станции. Определяется протоколом. Допустимые значения на входе: <b>LINK_ADDR_NO</b> – адрес станции не используется; <b>LINK_ADDR_1</b> – 1 байт; <b>LINK_ADDR_2</b> – 2 байта
aas	Длина поля общего адреса ASDU. Определяется протоколом. Допустимые значения на входе: <b>ASDU_ADDR_1</b> – 1 байт; <b>ASDU_ADDR_2</b> – 2 байта
cts	Длина поля причины передачи (COT). Определяется протоколом. Допустимые значения на входе: <b>COT_1</b> – 1 байт; <b>COT_2</b> – 2 байта
iaa	Длина адреса объекта информации. Определяется протоколом. Допустимые значения на входе: <b>IOA_1</b> – 1 байт; <b>IOA_2</b> – 2 байта; <b>IOA_3</b> – 3 байта
c1s	Глубина очереди сообщений класса 1
c2s	Глубина очереди сообщений класса 2
ofp	Если в течение этого времени (в мс) нет обмена, то по интерфейсу на выход <b>sts бит 0</b> блока выдается значение 0 – нет связи
frg	Максимальное межсимвольное расстояние при приеме кадра. Если это время (в мс) истекло – кадр считается недействительным
cpr	Период отправки с причиной передачи циклический, мс. Необходимость передачи данных с этой причиной задается для каждого выходного буфера. Рекомендуется задавать <b>cpr ≠ bpr</b> так, чтобы моменты срабатывания таймеров как можно реже пересекались
bpr	Период отправки сообщений с причиной передачи «фоновое сканирование», мс. Необходимость передачи данных с этой причиной задается для каждого выходного буфера. Рекомендуется задавать <b>cpr ≠ bpr</b> так, чтобы моменты срабатывания таймеров как можно реже пересекались

## Продолжение таблицы 2.1

Элемент	Описание
cfg	Конфигурация. Определяет поведение узла при старте и в процессе работы. Битовая маска: <b>Бит 0</b> – синхронизировать время после установления связи (используется системное время); <b>Бит 1</b> – послать команду общего опроса после установления связи; <b>Бит 4</b> – заблокировать команду синхронизации времени; <b>Бит 5</b> – включать метку времени при отправке данных с причиной передачи: циклические; <b>Бит 6</b> – включать метку времени при отправке данных с причиной передачи: фоновое сканирование; <b>Бит 8</b> – использовать структурированный адрес, прибавление стандартного смещения (для ТИТ смещение – <b>0x2000</b> , для ТС смещение – <b>0x1000</b> ); <b>Бит 12</b> – подключить протокол отправки диагностических сообщений
ito	Таймаут выполнения команды информация о процессе в направлении управления <b>C</b> (см. Приложение А), мс
sto	Таймаут выбора команды информация о процессе в направлении управления <b>C</b> (см. Приложение А), мс
Выходы	
itr	Выход для подключения <i>IECBufln</i>
st1	Выход статуса. Битовая маска: <b>Бит 0</b> – есть связь; <b>Бит 1</b> – выполнена команда синхронизации времени; <b>Бит 2</b> – первичный канальный уровень активен; <b>Бит 3</b> – вторичный канальный уровень активен
st2	Не используется

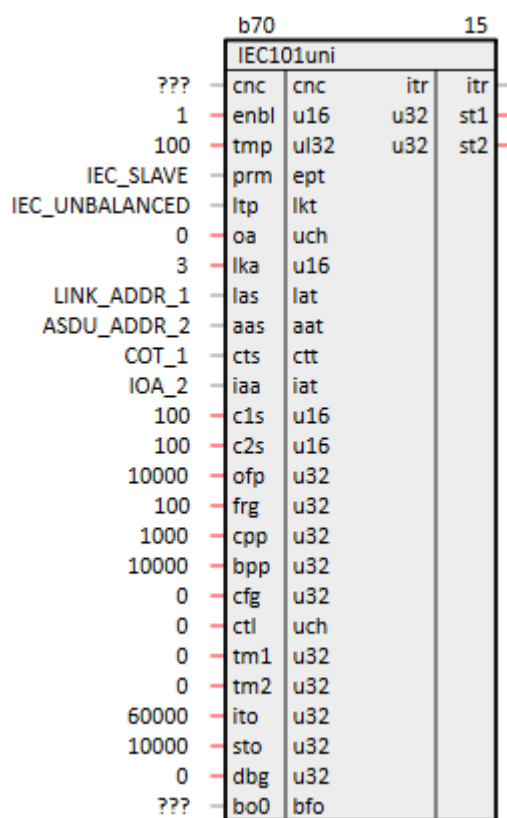


Рисунок 2.4 – Блок IEC101uni

Примеры работы с блоком приведены в [разделе 4.1](#).

## 2.2 Реализация протокола МЭК 60870-5-104 (IEC104uni)

Блок *IEC104uni* реализует протокол стандарта **МЭК 60870-5-104** и определяет функции контролирующей (клиента) и контролируемой станции (сервера).



### ПРИМЕЧАНИЕ

Для настройки сервера по протоколу **5-104** рекомендуется использовать более новый блок *IEC104Server*. Для него не требуется отдельный блок для работы с системным сокетом.



### ПРИМЕЧАНИЕ

В данном документе рассматривается обновленный блок *IEC104uni* (ревизия 2, версия библиотеки *palEC104 – 902* и выше).

Совместно с *IEC104uni* используются специальные буфера – *IECBufIn* для получаемых данных/команд, *IECBufOut* для отправляемых данных/команд. С помощью данных буферов создается база объектов информации, участвующих в обмене по указанному протоколу.

Данный блок следует размещать только в **Фоне**, потому что работа блока занимает значительное время.

**Таблица 2.2 – Назначение входов и выходов IEC104uni**

Элемент	Описание
<b>Входы</b>	
spc	Связь с блоком <i>TcpIpCIA</i> (для <i>IEC_MASTER</i> ) или <i>TcpIpSrA</i> (для <i>IEC_SLAVE</i> ) из библиотеки <i>paCore</i>
enbl	Режим работы блока: <b>0</b> – блок отключен; <b>1</b> – блок в работе; <b>2</b> – блок в работе, нет отправки данных (может использоваться для резерва)
ctl	Инициация команды управления. Битовая маска: <b>Бит 1</b> – инициировать команду общего опроса; <b>Бит 2</b> – инициировать команду общего опроса счетчиков; Бит 4 – не используется; Бит 5 – не используется; Бит 6 – не используется; Бит 7 – не используется
tm1	Не используется
tm2	Не используется
dbg	Конфигурация отладки. Битовая маска: <b>Бит 0...7</b> – битовая маска, которая определяет, какие типы сообщений выводить: <b>бит 0</b> – тип 1, ..., <b>бит 7</b> – тип 8 (высший уровень); <b>Бит 8</b> – выводить в консоль; <b>Бит 9</b> – выводить через блок <i>RamLog</i> из библиотеки <i>paCore</i> ; Бит 10 – выводить через протокол отправки диагностических сообщений (для разработчиков); <b>Бит 11</b> – выводить в консоль данные таймеров времени выполнения алгоритма и периода вызова алгоритма
bo	Входы для подключения <i>IECBufOut</i> (циклический)
<b>Входы (константные)</b>	
tmp	Период вызова главной функции блока, мс (см. <a href="#">рисунок 2.6</a> )
prn	Тип канального уровня: <b>IEC_MASTER</b> – иницирует соединение; <b>IEC_SLAVE</b> – ожидает соединение
oa	Не используется
w	Число пакетов, после получения последовательности которых узел обязан передать подтверждение о приеме (см. <i>t2</i> ): <b>w &lt; k</b> , рекомендуется <b>w = 2/3k</b>
k	Число пакетов, которое может отправить узел без подтверждения. Если <b>k</b> пакетов отправлено без подтверждения, то передача останавливается до получения подтверждения (см. <i>t1</i> ): <b>w &lt; k</b> , рекомендуется <b>w = 2/3k</b>

Продолжение таблицы 2.2

Элемент	Описание
t1	Таймаут при посылке кадров, мс. Если через время <b>t1</b> не было получено подтверждение – соединение закрывается: <b>t2 &lt; t1</b>
t2	Таймаут для подтверждения, если число принятых пакетов <b>w</b> не достигнуто, мс: <b>t2 &lt; t1</b>
t3	Таймаут для посылки блоков тестирования при долгом простое, мс. С периодом <b>t3</b> передается тестовый пакет для проверки канала, если ответа на тестовый пакет не последует – соединение закрывается
qsz	Длина очереди сообщений
crr	Период отправки с причиной передачи «циклический», мс. Необходимость передачи данных с этой причиной задается для каждого выходного буфера. Рекомендуется задавать <b>crr ≠ brr</b> так, чтобы моменты срабатывания таймеров как можно реже пересекались
brr	Период отправки сообщений с причиной передачи «фоновое сканирование», мс. Необходимость передачи данных с этой причиной задается для каждого выходного буфера. Рекомендуется задавать <b>crr ≠ brr</b> так, чтобы моменты срабатывания таймеров как можно реже пересекались
cfg	Конфигурация. Определяет поведение узла при старте и в процессе работы. Битовая маска: <b>Бит 0</b> – синхронизировать время после установления связи (используется системное время); <b>Бит 1</b> – послать команду общего опроса после установления связи; <b>Бит 2</b> – использовать глобальное время (UTC) в метках времени; <b>Бит 3</b> – принудительно считать временные метки действительными; <b>Бит 4</b> – заблокировать команду синхронизации времени; <b>Бит 5</b> – включать метку времени при отправке данных с причиной передачи: циклические; <b>Бит 6</b> – включать метку времени при отправке данных с причиной передачи: фоновое сканирование; <b>Бит 7</b> – включать метку времени при отправке данных с причиной передачи: общий опрос; <b>Бит 8</b> – использовать структурированный адрес, прибавление стандартного смещения (для ТИТ смещение – <b>0x2000</b> , для ТС смещение – <b>0x1000</b> ); <b>Бит 9</b> – отключить замещение старых данных новыми в очереди; Бит 12 – не используется; Биты 16, 17, 18 и 19 – 4-х битовое число. В режиме контролируемой станции ( <b>IEC_SLAVE</b> ), определяет максимальное количество одновременно подключаемых контролируемых станций
ito	Таймаут выполнения команды информация о процессе в направлении управления <b>C</b> (см. Приложение А), мс
sto	Таймаут выбора команды информация о процессе в направлении управления <b>C</b> (см. Приложение А), мс
bufsz	Размер буферов приема и передачи, байт
<b>Выходы</b>	
itr	Выход для подключения <b>IECBufln</b>
st0	Выход статуса. Битовая маска: <b>Бит 0</b> – есть связь; <b>Бит 1</b> – выполнена команда синхронизации времени
st1	Не используется

	b31		10		
	IEC104uni				
подключение к ТСРIP клиенту\серверу ???	cnc	cnc	itr	itr	коннектор для входных буферов
режим работы 1	enbl	u16	u32	st0	статус 0
период главного цикла блока (мс) 100	tmp	u132	u32	st1	не используется
тип канального уровня IEC_SLAVE	prm	ept			
не используется 0	oa	uch			
параметр W 8	w	u16			
параметр K 12	k	u16			
таймаут 1 15000	t1	u32			
таймаут 2 10000	t2	u32			
таймаут 3 20000	t3	u32			
глубина очереди сообщений 100	qsz	u16			
период отправки циклических данных 1000	cpp	u32			
период отправки данных фонового сканирования 10000	bpp	u32			
конфигурация 0	cfg	u32			
управление 0	ctl	uch			
не используется 0	tm1	u32			
не используется 0	tm2	u32			
таймаут выполнения команды 60000	ito	u32			
таймаут выбора команды 10000	sto	u32			
конфигурация отладки 0	dbg	u32			
размер буферов приема\передачи 4096	bufsz	u32			
коннектор выходного буфера ???	bo0	bfo			

Рисунок 2.5 – Блок IEC104uni

Рассмотрим алгоритм работы связки блоков *TcpIpCIA* и *IEC104uni* (см. рисунок 2.6).

В фоновом потоке последовательно вызываются главные функции блоков. В алгоритме работы протокола можно выделить четыре основные подпрограммы, первые две из которых реализуют работу с системным сокетом (блок *TcpIpCIA*).

Период вызова главной функции блока *IEC104uni* (и соответствующих подпрограмм) определяется входом **tmp**. Задание **tmp** может быть полезно для разгрузки потока в пользу других алгоритмов. При отсутствии ограничений можно установить **tmp = 1**.

При возникновении события на входе данные вместе с временной меткой (если она используется) передаются в очередь сообщений (длину очереди задает параметр **qsz**). Для каждого входа выделяется отдельная позиция в очереди.

Из очереди сообщений формируется пакет для отправки. Однотипные данные из очереди упаковываются в один пакет до достижения максимальной длины пакета, определенной стандартом.

Затем сформированный пакет помещается в буфер для передачи.

При задании параметра «длина очереди» **qsz** следует учитывать количество передаваемых объектов информации и частоту их изменения.

Размеры входных и выходных буферов **bufsz** следует выбирать из тех же соображений.

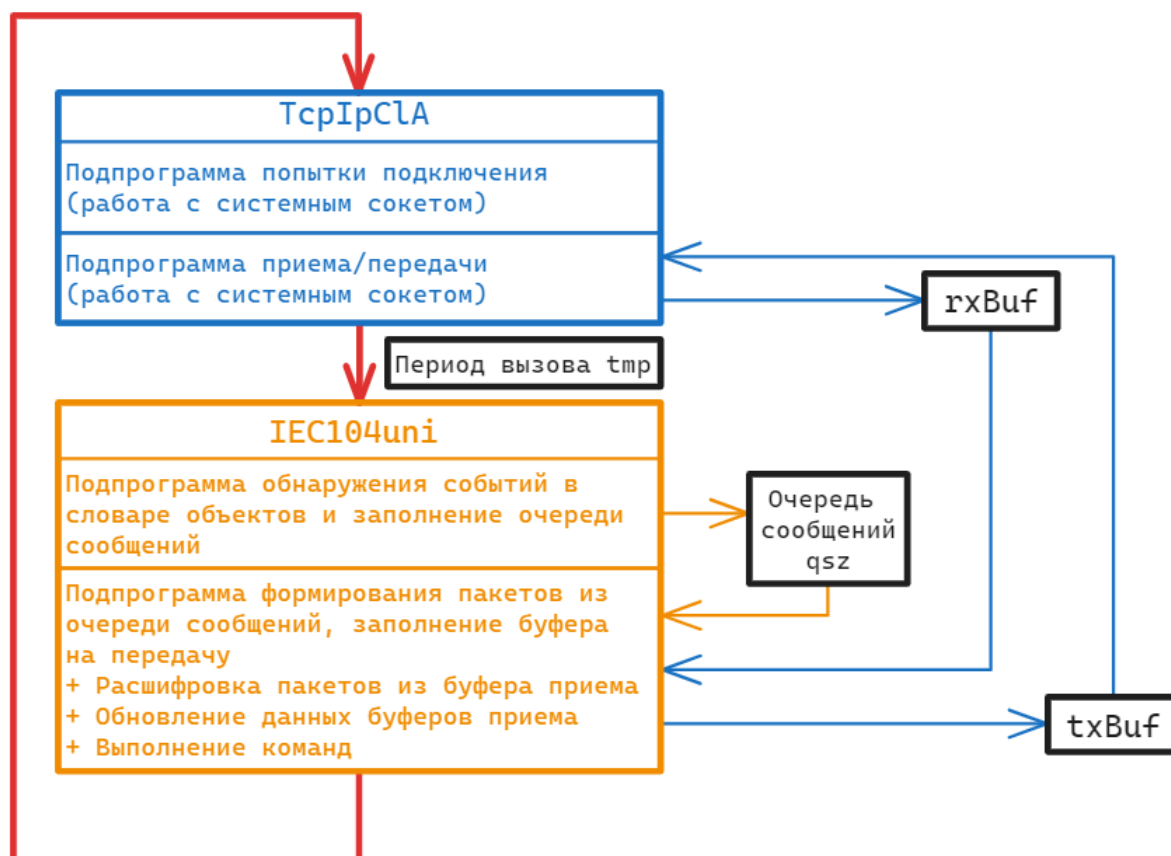


Рисунок 2.6 – Работа IEC104uni: последовательность вызова подпрограмм в фоновом потоке

Пример работы с блоком приведен в [разделе 3.2](#).

## 2.3 Реализация протокола МЭК 60870-5-104. Сервер (IEC104Server)

Блок **IEC104Server** реализует протокол стандарта **МЭК 60870-5-104** и определяет функцию контролируемой станции (сервера).



### ПРИМЕЧАНИЕ

Принципиальное отличие блока **IEC104Server** от **IEC104uni** состоит в том, что в **IEC104Server** уже включены подпрограммы работы с системными сокетами. Помимо этого, различные подпрограммы протокола выделены в отдельные потоки (см. [рисунок 2.8](#)). Таким образом, работа протокола независима от нагрузки фонового потока.

Совместно с **IEC104Server** используются специальные буфера – **IECBufIn** для получаемых данных/команд, **IECBufOut** для отправляемых данных/команд. С помощью данных буферов создается база объектов информации, участвующих в обмене по указанному протоколу.

Данный блок следует размещать только в **Фоне**, потому что работа блока занимает значительное время.

Таблица 2.3 – Назначение входов и выходов IEC104Server

Элемент	Описание
<b>Входы</b>	
enbl	Режим работы блока: <b>0</b> – блок отключен; <b>1</b> – блок в работе; <b>2</b> – блок в работе, нет отправки данных (может использоваться для резерва)
ctl	Инициация команды управления. Битовая маска: <b>Бит 1</b> – инициировать команду общего опроса; <b>Бит 2</b> – инициировать команду общего опроса счетчиков
tm1	Не используется
tm2	Не используется



Продолжение таблицы 2.3

Элемент	Описание
dbg	Конфигурация отладки. Битовая маска: <b>Бит 0...7</b> – битовая маска, которая определяет, какие типы сообщений выводить: <b>бит 0</b> – тип 1, ..., <b>бит 7</b> – тип 8 (высший уровень); <b>Бит 8</b> – выводить в консоль; <b>Бит 9</b> – выводить через блок <i>RamLog</i> из библиотеки <i>paCore</i> ; Бит 10 – выводить через протокол отправки диагностических сообщений (для разработчиков); <b>Бит 11</b> – выводить в консоль данные таймеров времени выполнения алгоритма и периода вызова алгоритма
bo	Входы для подключения <i>IECBufOut</i> (циклический)
<b>Входы (константные)</b>	
tmp	Период вызова подпрограммы обнаружения событий, мс (см. рисунок 2.8)
lip	Локальный IP-адрес
lprt	Локальный порт
sdr	Сетевой стек, для ПЛК ОВЕН "/"
evtsprio	Приоритет потока обработки событий (см. рисунок 2.8)
rxtprio	Приоритет потока приема/передачи (см. рисунок 2.8)
slts	Ограничение количества одновременных подключений клиентами
gxbsz	Размер буфера приема, байт
txbsz	Размер буфера передачи, байт
oa	Не используется
w	Число пакетов, после получения последовательности которых, узел обязан передать подтверждение о приеме (см. <b>t2</b> ): $w < k$ , рекомендуется $w = 2/3k$
k	Число пакетов, которое может отправить узел без подтверждения. Если <b>k</b> пакетов отправлено без подтверждения, то передача останавливается до получения подтверждения (см. <b>t1</b> ): $w < k$ , рекомендуется $w = 2/3k$
t1	Таймаут при посылке кадров, мс. Если через время <b>t1</b> не было получено подтверждение – соединение закрывается: $t2 < t1$
t2	Таймаут для подтверждения, если число принятых пакетов <b>w</b> не достигнуто, мс: $t2 < t1$
t3	Таймаут для посылки блоков тестирования при долгом простое, мс. С периодом <b>t3</b> передается тестовый пакет для проверки канала, если ответа на тестовый пакет не последует – соединение закрывается
qsz	Длина очереди сообщений
crr	Период отправки с причиной передачи циклический, мс. Необходимость передачи данных с этой причиной задается для каждого выходного буфера. Рекомендуется задавать <b>crr</b> $\neq$ <b>brp</b> так, чтобы моменты срабатывания таймеров как можно реже пересекались
brp	Период отправки сообщений с причиной передачи фоновое сканирование, мс. Необходимость передачи данных с этой причиной задается для каждого выходного буфера. Рекомендуется задавать <b>crr</b> $\neq$ <b>brp</b> так, чтобы моменты срабатывания таймеров как можно реже пересекались

## Продолжение таблицы 2.3

Элемент	Описание
cfg	Конфигурация. Определяет поведение узла при старте и в процессе работы. Битовая маска: <b>Бит 0</b> – синхронизировать время после установления связи (используется системное время); <b>Бит 1</b> – послать команду общего опроса после установления связи; <b>Бит 2</b> – использовать глобальное время (UTC) в метках времени; <b>Бит 3</b> – принудительно считать временные метки действительными; <b>Бит 4</b> – заблокировать команду синхронизации времени; <b>Бит 5</b> – включать метку времени при отправке данных с причиной передачи: циклические; <b>Бит 6</b> – включать метку времени при отправке данных с причиной передачи: фоновое сканирование; <b>Бит 7</b> – включать метку времени при отправке данных с причиной передачи: общий опрос; <b>Бит 8</b> – использовать структурированный адрес, прибавление стандартного смещения (для ТИТ смещение – <b>0x2000</b> , для ТС смещение – <b>0x1000</b> ); <b>Бит 9</b> – отключить замещение старых данных новыми в очереди
ito	Таймаут выполнения команды информация о процессе в направлении управления <b>C</b> (см. Приложение А), мс
sto	Таймаут выбора команды информация о процессе в направлении управления <b>C</b> (см. Приложение А), мс
Выходы	
itr	Выход для подключения <i>IECBufln</i>
st0	Выход статуса. Битовая маска: <b>Бит 0</b> – есть связь; <b>Бит 1</b> – выполнена команда синхронизации времени
st1	Не используется

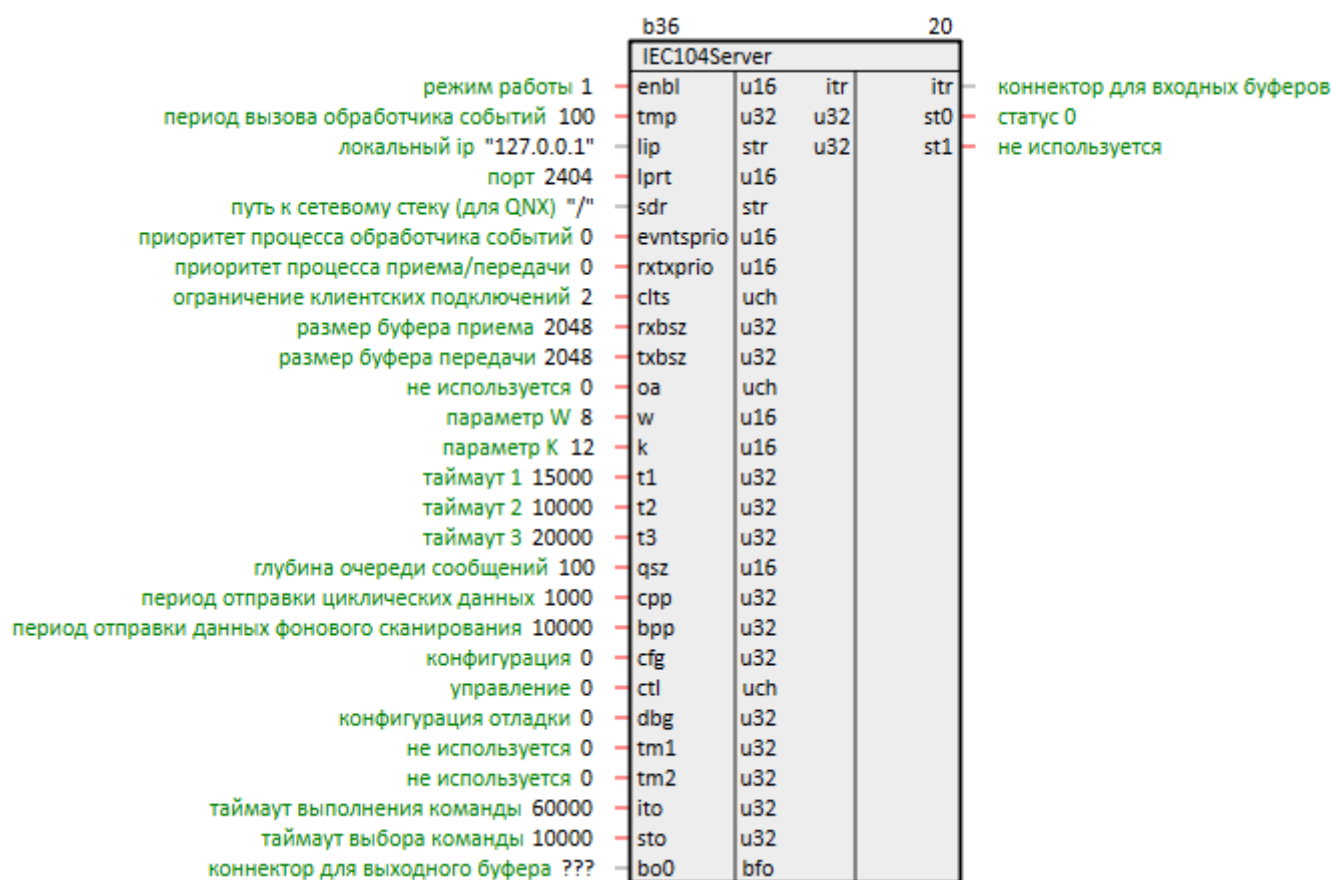


Рисунок 2.7 – Блок IEC104Server

Рассмотрим алгоритм работы блока *IEC104Server*.

В алгоритме работы протокола можно выделить четыре основные подпрограммы, первые две из которых реализуют работу с системным сокетом. Подпрограммы блока **IEC104Server** разделены на отдельные потоки (см. [рисунок 2.8](#)).

В фоновом потоке вызывается подпрограмма обнаружения подключения. В первом потоке с приоритетом **rxtxprio** вызывается программа приема/передачи данных. Во втором потоке с приоритетом **evntsprio** вызываются подпрограммы обнаружения событий с периодом **tmp** и формирования пакетов.

При возникновении события на входе данные вместе с временной меткой (если она используется) передаются в очередь сообщений (длину очереди задает **qsz**). Для каждого входа выделяется отдельная позиция в очереди.

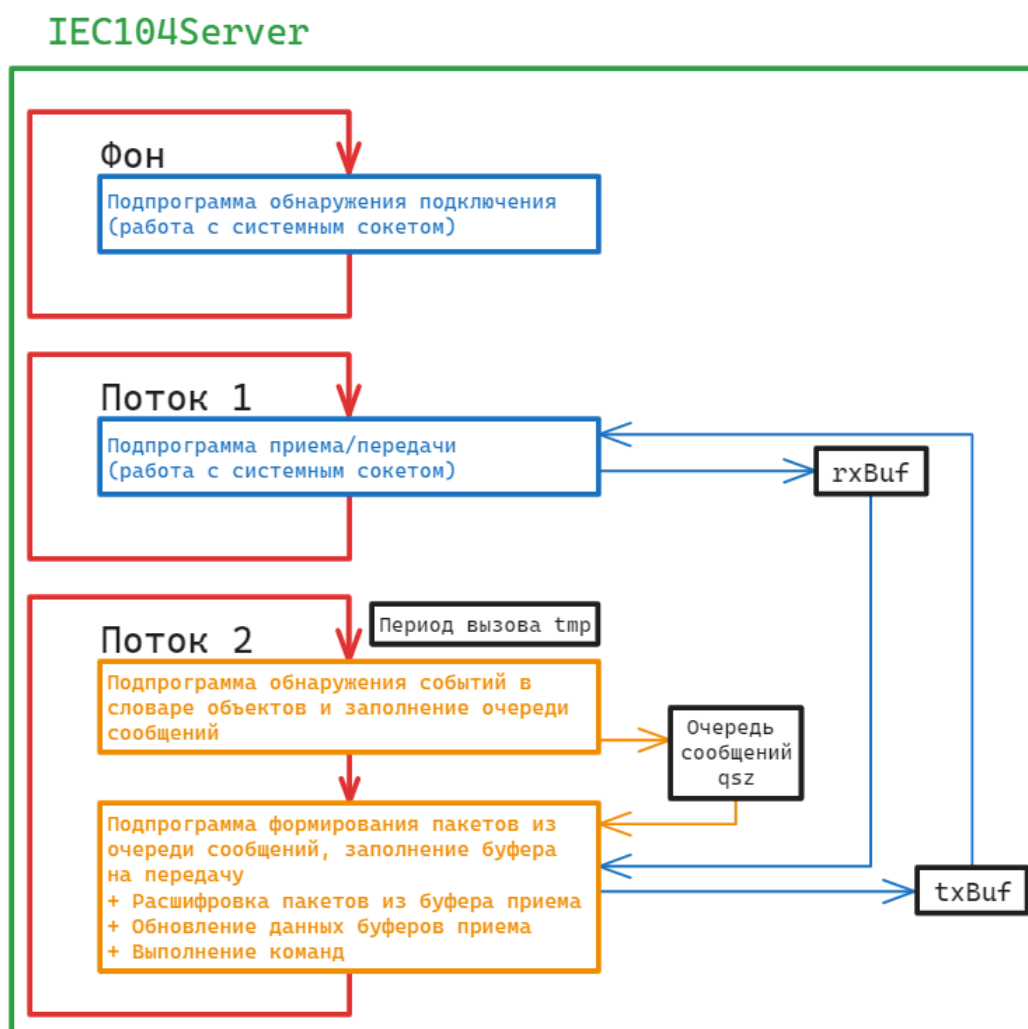
Из очереди сообщений формируется пакет для отправки. Однотипные данные из очереди упаковываются в один пакет до достижения максимальной длины пакета, определенной стандартом.

Затем сформированный пакет помещается в буфер для передачи.

При задании параметра «длина очереди» **qsz** следует учитывать количество передаваемых объектов информации и частоту их изменения.

Размеры входных и выходных буферов, соответственно, **rxbsz** и **txbsz** следует выбирать из тех же соображений.

Отследить переполнение очереди сообщений можно с помощью блока **IECInfo**.



**Рисунок 2.8 – Работа IEC104Server: разделение на потоки и последовательность вызова подпрограмм**

Пример работы с блоком приведен в [разделе 3.1](#).

## 2.4 Входной буфер (IECBufIn)

Блок *IECBufIn* используется совместно с блоками *IEC101uni*, *IEC104uni* и *IEC104Server* как входной буфер.

Данный блок можно размещать только в **Фоне**.

**Таблица 2.4 – Назначение входов и выходов IECBufIn**

Элемент	Описание
<b>Входы</b>	
itr	Вход для подключения к блоку протокола
ini	Вход данных для начальной инициализации выхода (циклический)
ctl	Не используется (циклический)
<b>Входы (константные)</b>	
tid	Тип ASDU ( <b>Type ID</b> ) для всех объектов информации в пределах буфера. Определяется протоколом. Допустимые значения на входе описаны в справке среды на библиотеку и в <a href="#">Приложении А</a>
adr	Адрес ASDU (общий адрес сектора)
itg	Не используется
dcl	Не используется
ioa	Адрес объекта информации в пределах сектора (циклический)
tp	Тип данных на выходах <sup>1)</sup> (циклический). Определяется средой <b>Полигон</b> . Допустимые значения на входе: <b>DO</b> (Char), <b>uDO</b> (Unsigned Char) – символьный; <b>IO</b> (Short), <b>uIO</b> (Unsigned Short) – 16-битное целое; <b>LO</b> (Long), <b>uLO</b> (Unsigned Long), <b>LX</b> (int32_t), <b>uLX</b> (uint32_t) – 32-битное целое; <b>AO</b> (Float) – число с плавающей запятой
flg	Не используется (циклический)
<b>Выходы</b>	
bi	Не используется
o	Выход данных (циклический)
qlf	Квалификатор (циклический). Выход имеет различные функции, в зависимости от типа объекта информации (см. <a href="#">Приложение А</a> ): <ol style="list-style-type: none"> <li>Для типов <b>M_*</b> (кроме <b>M_IT_*</b>) – как выход значения квалификатора, определяется протоколом (битовая маска):  <b>Бит 0 – OV</b>: переполнение (для <b>M_ME_*_1</b>);  <b>Бит 4 – BL</b>: блокировка (для <b>M_*_*_1</b>);  <b>Бит 5 – SB</b>: проведено замещение (для <b>M_*_*_1</b>);  <b>Бит 6 – NT</b>: неактуальное значение (для <b>M_*_*_1</b>);  <b>Бит 7 – IV</b>: недействительное значение (для <b>M_*_*_1</b>)</li> <li>Для типов <b>M_IT_*</b> – как выход вспомогательных полей данных<sup>2)</sup>, определяется протоколом (битовая маска):  <b>Биты 0...4 – SQ</b>: номер передаваемой последовательности;  <b>Бит 5 – CY</b>: произошло переполнение счетчика;  <b>Бит 6 – CA</b>: счетчик установлен;  <b>Бит 7 – IV</b>: показание счетчика недействительно</li> </ol>
tm1	Выход времени, младшее слово <sup>3)</sup> (циклический)
tm2	Выход времени, старшее слово <sup>3)</sup> (циклический)
sts	Не используется (циклический)

**ПРИМЕЧАНИЕ**

- 1) Для типов ASDU **M\_EP\_\*** (см. Приложение А) следует использовать базовый тип **uLX** на входе **tp**, т.к. входные данные для данного ASDU структурированы. Для расшифровки структуры используется блок **IECEPFromInt**.
- 2) Для расшифровки вспомогательных полей данных типов **M\_IT\_\*** используется блок **IECITSQIn**.
- 3) Для расшифровки данных о времени на выходах используется блок **IECTransTime**.

Блоки **IECBufIn** и **IECBufOut** образуют словарь объектов информации, который участвует в обмене по протоколу. Словарь состоит из секторов и объектов внутри секторов.

Вход **adr** определяет общий адрес сектора, а вход **ioa** – адрес объекта информации внутри сектора. Сочетание **adr** и **ioa** должно быть уникальным для каждого объекта информации внутри станции.

Вход **tp** определяет тип данных на выходах блока. Желательно выбирать тип данных, соответствующий типу объекта информации. При несовпадении типов происходит явное преобразование типа, заданного **tp** к типу, определенному протоколом.

Например, для типов ASDU **C\_BO\_\*\_1** и **M\_BO\_\*\_1** следует установить тип входа **uLX** (`uint32_t`), т.к. для инкапсуляции передаваемых данных в протоколе используется структура данных типа `uint32_t`. Для типов ASDU **M\_ME\_NB\_1**, **M\_ME\_TB\_1** и **M\_ME\_TE\_1** следует установить тип входа **IO** (`Short`), т.к. для инкапсуляции используется структура данных типа `int16_t`.

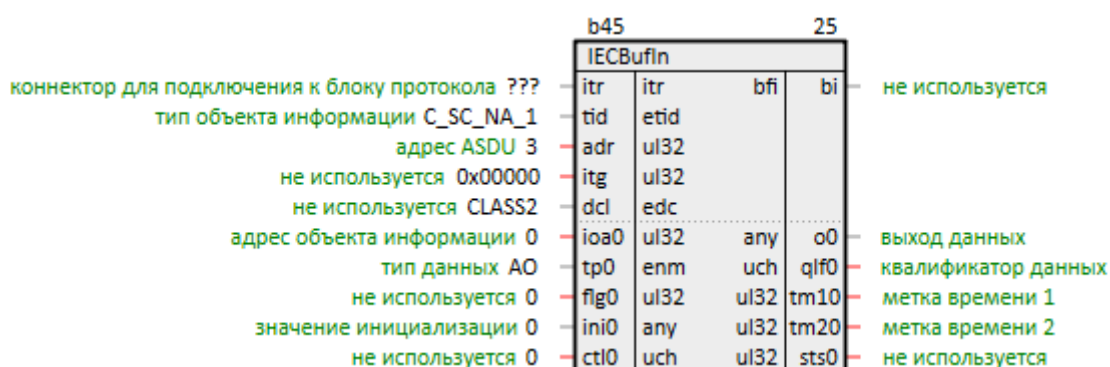


Рисунок 2.9 – Блок IECBufIn

Примеры работы с блоком приведены в разделе 3.1.

## 2.5 Выходной буфер (IECBufOut)

Блок **IECBufOut** используется совместно с блоками **IEC101uni**, **IEC104uni** и **IEC104Server** как выходной буфер.

Данный блок можно размещать только в **Фоне**.

Таблица 2.5 – Назначение входов и выходов IECBufOut

Элемент	Описание
<b>Входы (константные)</b>	
tid	Тип ASDU ( <b>Type ID</b> ) для всех объектов информации в пределах буфера. Определяется протоколом. Допустимые значения на входе описаны в справке среды на библиотеку и в <a href="#">Приложении А</a>
adr	Адрес ASDU (общий адрес сектора)
itg	Группа опроса (в т.ч. опроса счетчиков). Определяется протоколом. Допустимые значения на входе (битовая маска): <b>Бит 0</b> – посылать данные при общем запросе; <b>Бит 1</b> – посылать данные при запросе группы 1; <b>Бит 2</b> – посылать данные при запросе группы 2; ... <b>Бит 16</b> – посылать данные при запросе группы 16
dcl	Класс данных (только для <b>IEC101uni</b> ). Определяется протоколом. Допустимые значения на входе: <b>CLASS1</b> – класс 1; <b>CLASS2</b> – класс 2

Продолжение таблицы 2.5

Элемент	Описание
trt	<p>Определяет причину передачи (см. Приложение А). Определяется протоколом. Допустимые значения на входе (битовая маска):</p> <p><b>Бит 0</b> – передавать данные при изменении (причина передачи <b>3</b>). Для типов ASDU <b>M_ME_xx_1</b> посредством входа <b>flg</b> есть возможность задать дополнительные условия передачи;</p> <p><b>Бит 1</b> – передавать данные циклически (причина передачи <b>1</b>) с периодом, указанным на входе <b>cpp</b> блока протокола;</p> <p><b>Бит 2</b> – передавать данные фоновым сканированием (причина передачи <b>2</b>) с периодом, указанным на входе <b>bpp</b> блока протокола;</p> <p><b>Бит 3</b> – отправить при активации бита <b>0</b> на входе <b>ctl</b>: команда для типов ASDU <b>C_*</b> (причины передачи <b>6, 8, 10</b>);</p> <p><b>Бит 4</b> – отправить при активации бита <b>1</b> на входе <b>ctl</b> (причина передачи определяется битом <b>6</b> на входе <b>flg</b>)</p>
ioa	Адрес объекта информации в пределах сектора (циклический)
tp	<p>Тип данных на выходах<sup>1)</sup> (циклический). Определяется средой <b>Полигон</b>. Допустимые значения на входе:</p> <p><b>DI</b> (Char), <b>uDI</b> (Unsigned Char) – символьный;</p> <p><b>II</b> (Short), <b>uII</b> (Unsigned Short) – 16-битное целое;</p> <p><b>LI</b> (Long), <b>uLI</b> (Unsigned Long), <b>LX</b> (int32_t), <b>uLX</b> (uint32_t) – 32-битное целое;</p> <p><b>AI</b> (Float) – число с плавающей запятой</p>
flg	<p>Флаги (циклический). Для типов ASDU <b>M_ME_xx_1</b> задает условия передачи спонтанных сообщений (см. Приложение А). Определяется протоколом. Допустимые значения на входе (битовая маска):</p> <p><b>Бит 0</b> – передавать данные, в случае, если текущее значение на входе <b>in</b> блока отличается от предыдущего более, чем на величину, определяемую входом <b>tsh</b> (<b>Threshold</b>) блока (мертвая зона);</p> <p><b>Бит 1</b> – не используется;</p> <p><b>Бит 2</b> – передавать данные, в случае, если значение на входе <b>in</b> меньше величины, определяемой входом <b>llm</b> (<b>Low Limit</b>);</p> <p><b>Бит 3</b> – передавать данные, в случае, если значение на входе <b>in</b> больше величины, определяемой входом <b>hlm</b> (<b>High Limit</b>);</p> <p>Для команд (тип ASDU <b>C_*</b>):</p> <p><b>Бит 4</b> – посылать команду при изменении значения на входе;</p> <p><b>Бит 5</b> – использовать предварительный выбор;</p> <p>Для всех, кроме команд:</p> <p><b>Бит 6</b> – причина передачи в принудительном режиме, т.е. при активации бита на входе <b>ctl</b>: <b>0</b> – причина передачи <b>1</b> (циклически), <b>1</b> – причина передачи <b>2</b> (фоновое сканирование)</p>
<b>Входы</b>	
in	Входы данных (циклический)
qlf	<p>Квалификатор<sup>2)</sup> (циклический). Вход имеет различные функции, в зависимости от типа объекта информации (см. Приложение А):</p> <ol style="list-style-type: none"> <li>Для типов <b>M_*</b> (кроме <b>M_IT_*</b>) – как вход значения квалификатора, определяется протоколом (битовая маска): <ul style="list-style-type: none"> <li><b>Бит 0</b> – <b>OV</b>: переполнение (для <b>M_ME_*_1</b>);</li> <li><b>Бит 4</b> – <b>BL</b>: блокировка (для <b>M_*_*_1</b>);</li> <li><b>Бит 5</b> – <b>SB</b>: проведено замещение (для <b>M_*_*_1</b>);</li> <li><b>Бит 6</b> – <b>NT</b>: неактуальное значение (для <b>M_*_*_1</b>);</li> <li><b>Бит 7</b> – <b>IV</b>: недействительное значение (для <b>M_*_*_1</b>)</li> </ul> </li> <li>Для типов <b>M_IT_*</b> – как вход вспомогательных полей данных<sup>2)</sup>, определяется протоколом: <ul style="list-style-type: none"> <li><b>Биты 0...4</b> – <b>SQ</b>: номер передаваемой последовательности;</li> <li><b>Бит 5</b> – <b>CY</b>: произошло переполнение счетчика;</li> <li><b>Бит 6</b> – <b>CA</b>: счетчик установлен;</li> <li><b>Бит 7</b> – <b>IV</b>: показание счетчика недействительно</li> </ul> </li> </ol>

## Продолжение таблицы 2.5

Элемент	Описание
tsh	Вход данных <b>Threshold</b> (циклический). Имеет значение только для типов ASDU <b>M_ME*_1</b> (см. Приложение А) (мертвая зона)
smt	Не используется (циклический)
llm	Вход данных <b>Low Limit</b> (циклический). Имеет значение только для типов ASDU <b>M_ME*_1</b> (см. Приложение А)
hlm	Вход данных <b>High Limit</b> (циклический). Имеет значение только для типов ASDU <b>M_ME*_1</b> (см. Приложение А)
ctl	Управление (циклический). Битовая маска: <b>Бит 0</b> – выполнить команду ( <b>бит 3</b> на входе <b>trt</b> ); <b>Бит 1</b> – отправить принудительно ( <b>бит 4</b> на входе <b>trt</b> )
Выходы	
bo	Выход для подключения к блоку протокола
sts	Не используется (циклический)

**ПРИМЕЧАНИЕ**

<sup>1)</sup> Для типов ASDU **M\_EP\*** (см. Приложение А) следует использовать базовый тип **uLX** на входе **tp**, т.к. входные данные для данного ASDU структурированы. Для формирования структуры используется блок **IECIntFromEP**.

<sup>2)</sup> Для формирования вспомогательных полей данных типов **M\_IT\*** используется блок **IECITSQOut**.

Блоки **IECBufIn** и **IECBufOut** образуют словарь объектов информации, который участвует в обмене по протоколу. Словарь состоит из секторов и объектов внутри секторов.

Вход **adr** определяет общий адрес сектора, а вход **ioa** – адрес объекта информации внутри сектора. Сочетание **adr** и **ioa** должно быть уникальным для каждого объекта информации внутри станции.

Вход **tp** определяет тип данных на выходах блока. Желательно выбирать тип данных, соответствующий типу объекта информации. При несовпадении типов происходит явное преобразование типа, заданного **tp** к типу, определенному протоколом.

Например, для типов ASDU **C\_BO\*\_1** и **M\_BO\*\_1** следует установить тип входа **uLX** (`uint32_t`), т.к. для инкапсуляции передаваемых данных в протоколе используется структура данных типа `uint32_t`. Для типов ASDU **M\_ME\_NB\_1**, **M\_ME\_TB\_1** и **M\_ME\_TE\_1** следует установить тип входа **II** (Short), т.к. для инкапсуляции используется структура данных типа `int16_t`.

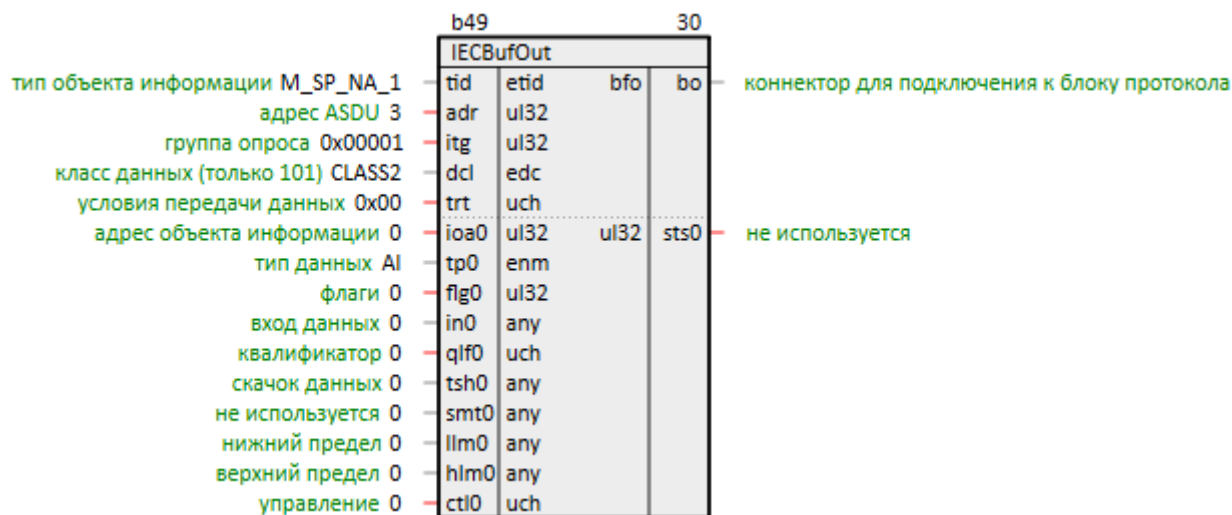


Рисунок 2.10 – Блок IECBufOut

Примеры работы с блоком приведены в разделе 3.1.



## 2.6 Разбиение структуры типа ASDU M\_EP\_\* (IECEPFromInt)

Блок *IECEPFromInt* предназначен для разбиения структурированных данных на выходе входного буфера с типом ASDU M\_EP\_\* (см. Приложение A) на отдельные поля данных. Базовый тип выхода буфера (определяется входом *tp* буфера *IECBufIn*) должен быть *uLX*.

Данный блок можно размещать только в **Фоне**.

Таблица 2.6 – Назначение входов и выходов *IECEPFromInt*

Элемент	Описание
<b>Входы</b>	
struct	Вход для подключения к выходу данных <i>o</i> блока <i>IECBufIn</i>
<b>Выходы</b>	
state	Выход состояния, определяется протоколом
msec	Выход интервала времени, мс

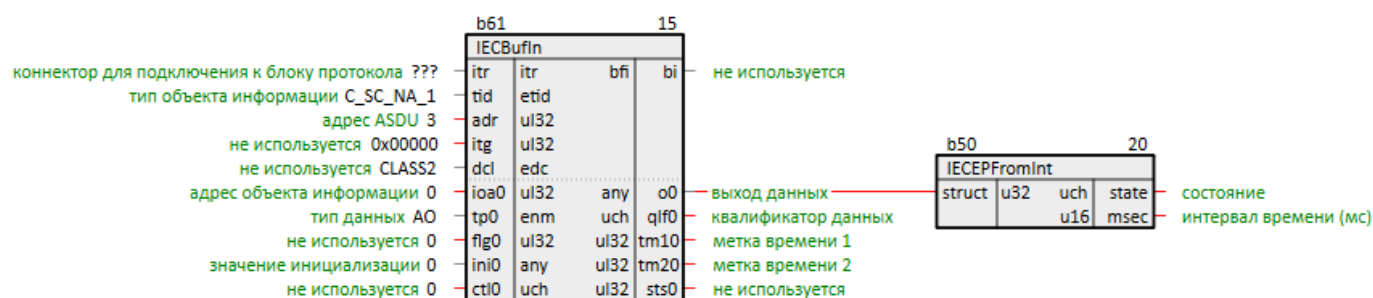


Рисунок 2.11 – Блок *IECEPFromInt*

Примеры работы с блоком приведены в разделах 3.1.2 и 3.2.3.

## 2.7 Формирование структуры типа ASDU M\_EP\_\* (IECIntFromEP)

Блок *IECIntFromEP* предназначен для объединения входных данных с типом ASDU M\_EP\_\* (см. Приложение A) в структурированное слово для использования в выходном буфере. Базовый тип входа буфера (определяется входом *tp* буфера *IECBufOut*) должен быть *uLX*.

Данный блок можно размещать только в **Фоне**.

Таблица 2.7 – Назначение входов и выходов *IECIntFromEP*

Элемент	Описание
<b>Входы</b>	
state	Вход состояния, определяется протоколом
msec	Вход интервала времени, мс
cfg	Вход конфигурации (константный). Битовая маска: <b>Бит 0</b> – обновлять выход только при обновлении входа <b>state</b> (это может быть полезно при спорадическом типе передачи данных)
<b>Выходы</b>	
struct	Выход для подключения к входу данных <i>in</i> блока <i>IECBufOut</i>



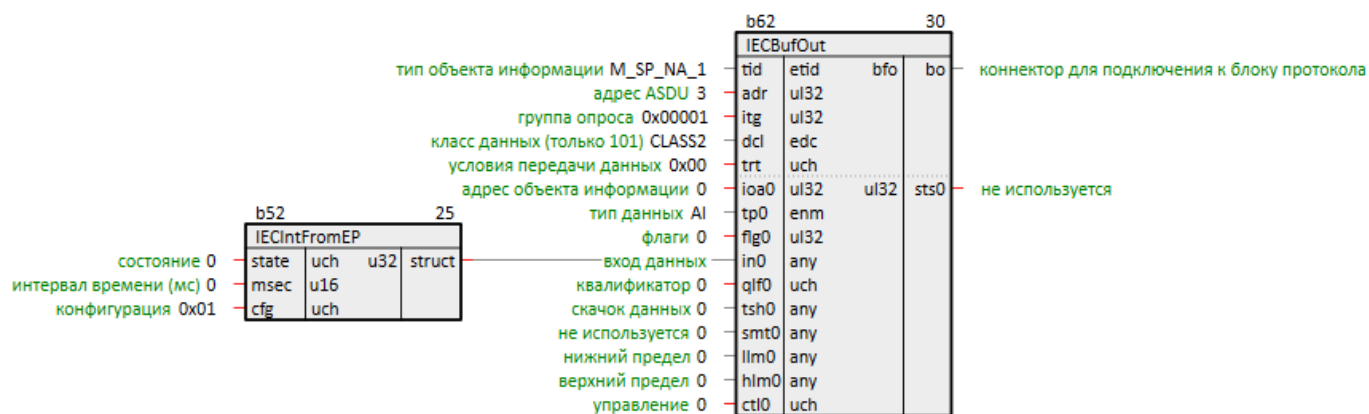


Рисунок 2.12 – Блок IECIntFromEP

Примеры работы с блоком приведены в [разделах 3.1.3 и 3.2.3](#).

## 2.8 Расшифровка описателя качества типа ASDU M\_IT\_\* (IECITSQIn)

Блок *IECITSQIn* предназначен для разбиения вспомогательных структурированных данных на выходе входного буфера *IECBufIn* с типом ASDU M\_IT\_\* (см. [Приложение А](#)) на отдельные поля.

Данный блок можно размещать только в **Фоне**.

Таблица 2.8 – Назначение входов и выходов IECITSQIn

Элемент	Описание
<b>Входы</b>	
in	Вход для подключения к выходу данных <i>qlf</i> блока <i>IECBufIn</i>
<b>Выходы</b>	
SQ	Номер передаваемой последовательности – передается в 5 младших битах
CY	<b>0</b> – за соответствующий период интегрирования не произошло переполнение счетчика; <b>1</b> – за соответствующий период интегрирования произошло переполнение счетчика
CA	<b>0</b> – после последнего считывания счетчик не был установлен; <b>1</b> – после последнего считывания счетчик был установлен
IV	<b>0</b> – показания счетчика действительны; <b>1</b> – показания счетчика недействительны

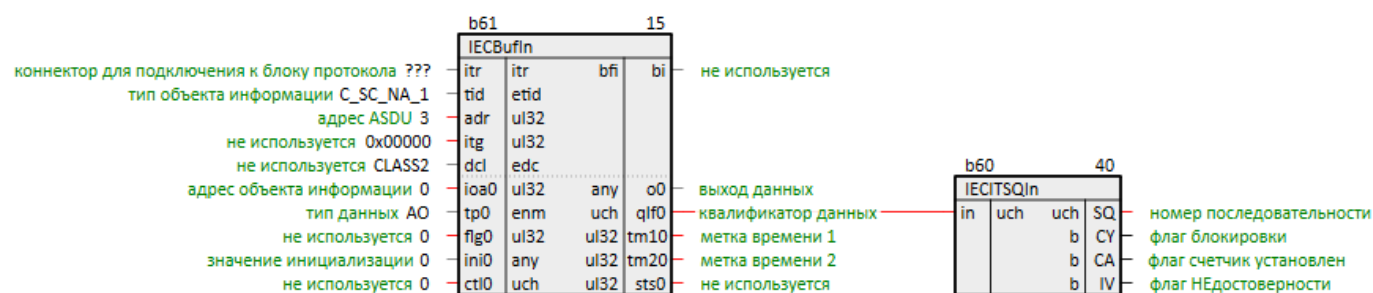


Рисунок 2.13 – Блок IECITSQIn

Примеры работы с блоком приведены в [разделах 3.1.4 и 3.2.4](#).

## 2.9 Формирование описателя качества типа ASDU M\_IT\_\* (IECITSQOut)

Блок *IECITSQOut* предназначен для компоновки вспомогательных данных с типом ASDU M\_IT\_\* (см. [Приложение А](#)) и передачи на вход *qlf* выходного буфера *IECBufOut*.

Данный блок можно размещать только в **Фоне**.

Таблица 2.9 – Назначение входов и выходов IECITSQOut

Элемент	Описание
<b>Входы</b>	
SQ	Номер передаваемой последовательности – передается в 5 младших битах
CY	0 – за соответствующий период интегрирования не произошло переполнение счетчика; 1 – за соответствующий период интегрирования произошло переполнение счетчика
CA	0 – после последнего считывания счетчик не был установлен; 1 – после последнего считывания счетчик был установлен
IV	0 – показания счетчика действительны; 1 – показания счетчика недействительны
<b>Выходы</b>	
out	Выход для подключения к входу <b>qlf</b> блока <b>IECBufOut</b>

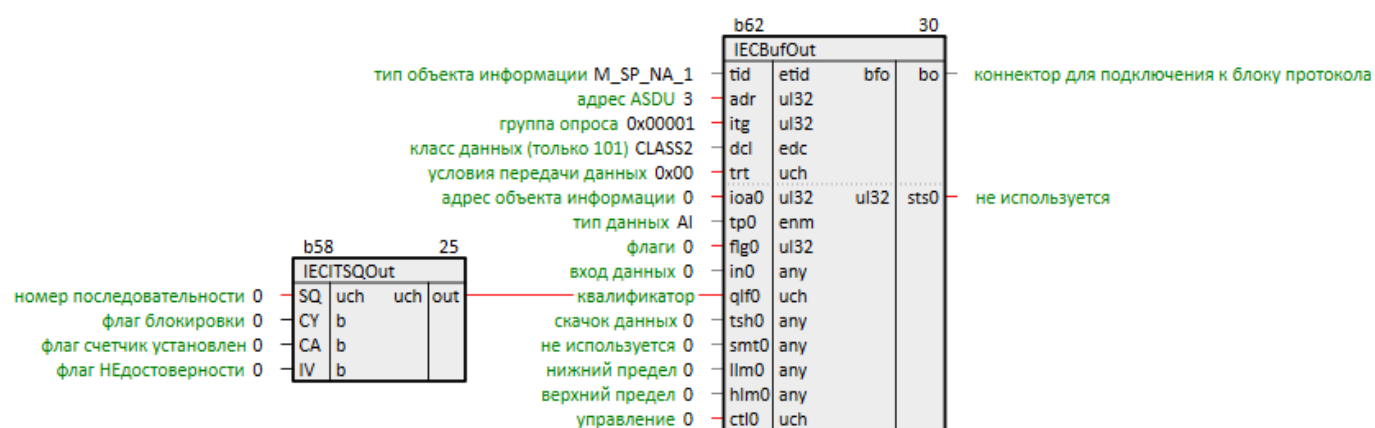


Рисунок 2.14 – Блок IECITSQOut

Примеры работы с блоком приведены в [разделах 3.1.4](#) и [3.2.4](#).

## 2.10 Расшифровка временных меток (IECTransTime)

Блок **IECTransTime** предназначен для преобразования формата вывода временных меток входными буферами **IECBufIn** в поля данных.

Данный блок можно размещать только в **Фоне**.

Таблица 2.10 – Назначение входов и выходов IECTransTime

Элемент	Описание
<b>Входы</b>	
tm1	Вход для подключения к выходу <b>tm1</b> блока <b>IECBufIn</b>
tm2	Вход для подключения к выходу <b>tm2</b> блока <b>IECBufIn</b>
<b>Выходы</b>	
msec	Выход времени, мс (Unix)
IV	Выход сигнализации недостоверного времени
str	Строковое представление времени (для отладки)

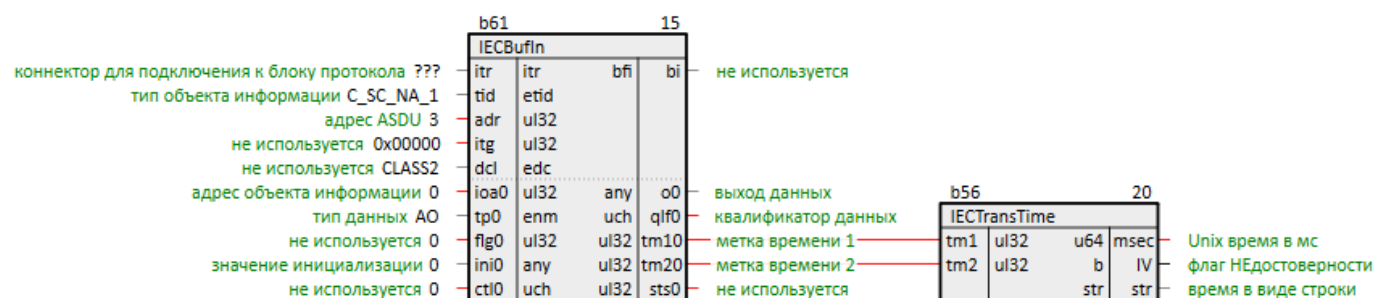


Рисунок 2.15 – Блок IECTransTime

Примеры работы с блоком приведены в [разделе 3.1](#).

## 2.11 Получение отладочной информации (IECInfo)

Блок **IECInfo** предназначен для получения отладочной информации о работе протокола. Блок **IECInfo** совместим только с блоком **IEC104Server**.

Первая часть выходов блока позволяет отследить времена выполнения функций блока **IEC104Server**. Обозначения (см. [рисунок 2.8](#)):

- **main** – поток 2;
- **evnts** – подпрограмма регистрации событий;
- **aux** – подпрограмма формирования буфера на передачу/расшифровки буфера на прием;
- **txrx** – поток 1.

Вторая часть выходов позволяет отследить используемый размер очереди сообщений, переполнение очереди и др. Параметры размера очередей и буферов приводятся в байтах.



### ПРИМЕЧАНИЕ

Блок **IECInfo** считывает значения **msgQ\*** только после того, как завершается подпрограмма формирования пакетов из очереди. Таким образом, по этим значениям можно судить о количестве сообщений, которые не были помещены в буфер на отправку за данный цикл. Возможные причины: недостаточный размер буфера на передачу, нехватка производительности подсистемы сокетов и др.

Данный блок можно размещать только в **Фоне**.

Таблица 2.11 – Назначение входов и выходов IECInfo

Элемент	Описание
<b>Входы</b>	
itr	Вход для подключения к блоку <b>IEC104Server</b>
ctl	Не используется
tmout	Период вычисления времен исполнения, мс
rst	Сброс абсолютных значений
<b>Выходы</b>	
*tmmin	Минимальное время исполнения
*tmmax	Максимальное время исполнения
*tmavrg	Среднее время исполнения
*tmnum	Количество вызовов подпрограммы
*tmabsmax	Максимальное время исполнения за все время работы <b>IECInfo</b>
<b>Выходы (циклические)</b>	
msgQmax	Максимальный используемый размер очереди сообщений
msgQavrg	Средний используемый размер очереди сообщений
msgQnum	Количество вызовов подпрограммы
msgQabsmax	Максимальный используемый размер очереди сообщений за все время работы <b>IECInfo</b>
msgQfull	Инкрементируется при переполнении очереди
gplQfull	Инкрементируется при переполнении очереди для отправки подтверждений
reachedK	Инкрементируется при достижении числа пакетов, отправленных без подтверждения, <b>K</b>
txbufmax	Максимальный размер буфера передачи
txbufavrg	Средний размер буфера передачи
txbufnum	Количество вызовов подпрограммы
txbufabsmax	Максимальный размер буфера передачи за все время работы <b>IECInfo</b>
txbufReachedLim	Инкрементируется при переполнении выходного буфера

		b63			15
IECInfo					
???	itr		itr	flt	maintmmin
0	ctl		u32	flt	maintmmax
1000	tmout		u32	flt	maintmavrg
0	rst		b	u32	maintmnum
				flt	maintmabsmax
				flt	evntstmmin
				flt	evntstmmax
				flt	evntstmavrg
			u32	flt	evntstmnum
				flt	evntstmabsmax
				flt	auxtmmin
				flt	auxtmmax
				flt	auxtmavrg
			u32	flt	auxtmnum
				flt	auxtmabsmax
				flt	txrxtmmin
				flt	txrxtmmax
				flt	txrxtmavrg
			u32	flt	txrxtmnum
				flt	txrxtmabsmax
			u32	flt	msgQmax0
			u32	flt	msgQavrg0
			u32	flt	msgQnum0
			u32	flt	msgQabsmax0
			u32	flt	msgQfull0
			u32	flt	rplQfull0
			u32	flt	reachedK0
			u32	flt	txbufmax0
			u32	flt	txbufavrg0
			u32	flt	txbufnum0
			u32	flt	txbufabsmax0
			u32	flt	txbufReachedLim0

Рисунок 2.16 – Блок IECInfo

## 3 Примеры настройки обмена по протоколу МЭК 60870-5-104

### 3.1 Настройка обмена в режиме сервера по протоколу 5-104

В данном примере ПЛК210 будет выступать в роли контролируемой станции (сервера).

Пример доступен для скачивания по [ссылке](#). Пароль для доступа к отладчику – 1.

Для настройки обмена в режиме сервера по протоколу 5-104 следует:

1. Добавить на любую страницу места работы **Фон** блок **IEC104Server**.
2. На вход **lip** подать IP-адрес контроллера в виде SQL-запроса. Запрос IP-адреса (prop\_ip):  
"<sql>SELECT value FROM blocks\_prop WHERE indx=:module AND type="prop\_ip"</sql>"
3. Добавить коннектор для выходного буфера **IECBufOut**.
4. Установить период для отправки данных с причиной 1 (циклический) равным 5 секундам – **cpp = 5000**.

Другие параметры блока перечислены на рисунке ниже.

"<sql>SELECT value FROM blocks\_prop WHERE indx=:module AND type="prop\_ip"</sql>"

	Server			5	
	<b>IEC104Server</b>				
режим работы 1	enbl	u16	itr	itr	коннектор для входных буферов
период вызова обработчика событий 100	tmp	u32	u32	st0	статус 0
локальный ip "[SQL]"	lip	str	u32	st1	не используется
порт 2404	lppt	u16			
путь к сетевому стеку (для QNX) "/"	sdr	str			
приоритет процесса обработчика событий 0	evntsprio	u16			
приоритет процесса приема/передачи 0	rxtxprio	u16			
ограничение клиентских подключений 2	clts	uch			
размер буфера приема 2048	rxbsz	u32			
размер буфера передачи 2048	txbsz	u32			
не используется 0	oa	uch			
параметр W 8	w	u16			
параметр K 12	k	u16			
таймаут 1 15000	t1	u32			
таймаут 2 10000	t2	u32			
таймаут 3 20000	t3	u32			
глубина очереди сообщений 100	qsz	u16			
период отправки циклических данных 5000	cpp	u32			
период отправки данных фонового сканирования 10000	bpp	u32			
конфигурация 0	cfg	u32			
управление 0	ctl	uch			
конфигурация отладки 0x11F	dbg	u32			
не используется 0	tm1	u32			
не используется 0	tm2	u32			
таймаут выполнения команды 60000	ito	u32			
таймаут выбора команды 10000	sto	u32			
коннектор для подключения к блоку протокола ???	bo0	bfo			

Рисунок 3.1 – Настройка IEC104Server

#### 3.1.1 Передача информации о процессе в направлении контроля (M)

В этом подразделе приведены примеры обмена данными в направлении контроля. Такому обмену в ASDU присваивается метка **M** (см. [Приложение А](#)).

Например, необходимо отправить от сервера клиенту объект информации с типом **M\_SP\_NA\_1** – одноэлементный объект информации с описателем качества.

Для настройки передачи информации о процессе в направлении контроля (M) следует:

1. Добавить на любую страницу места работы **Фон** выходной буфер **IECBufOut**.



#### ПРИМЕЧАНИЕ

В данном примере настраивается передача данных в прямом направлении – от сервера к клиенту. Аналогично настраивается обмен в обратном направлении – от клиента к серверу.

- Установить **tid = M\_SP\_NA\_1**, адрес ASDU **adr = 3**, адрес объекта информации **ioa = 1**.  
Тип данных **tp** установим равным **DI**, так как у **M\_SP\_NA\_1** в младшем разряде передается **1 бит** ТС.
- Условия передачи данных **trt** установить равным **0x03**: **бит 0** – спорадическая передача (причина передачи **3**), **бит 1** – циклическая передача с периодом **cpp = 5000** мс (причина передачи **1**) (см. Приложение Б).

На вход **qlf** заведем выход с блока **ToReg8** из библиотеки **paCore** – формирователь битовой маски описателя качества.

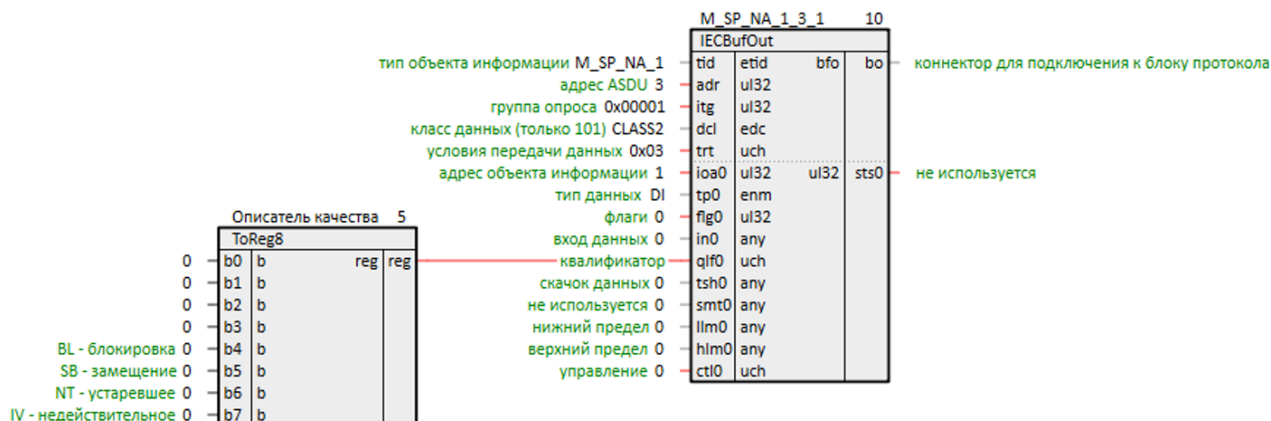


Рисунок 3.2 – Настройка IECBufOut. M\_SP\_NA\_1

- В данном примере в качестве клиента используется **Multi-Protocol MasterOPC**. Можно также запустить в качестве клиента проект, рассмотренный в разделе 3.1, на виртуальном контроллере с помощью панели отладки.
- Настройка клиента в Multi-Protocol MasterOPC показана на рисунках ниже.

Протокол <<IEC104CLIENT>> Устройство <<iec104client>>	
<b>Общие настройки</b>	
Комментарий	60870-5 iec104 Device
Включено в работу	true
Период опроса	1000
Размерность периода опроса	мс
Начальная фаза	0
Размерность фазы	мс
Старт после запуска	true
Разрешение отладочных сообщений	true
<b>Свойства протокола</b>	
IP Адрес	10.2.12.12
IP Порт	2404
Адрес ASDU	1
Использовать время компьютера	false
Время устройства (true-utc,false-local)	true
Запрашивать данные групповым запросом при старте	true
Общий опрос <100> (секунды) - 0 нет	0
Группы общего опроса (0,1,...15,16)	0
Опрос счетчиков <101> (секунды) - 0 нет	0
Группы опроса счетчиков (0,1,...15,16,...)	0
Команда считывания <102> (секунды) - 0 нет	0
Синхронизация времени (секунды) - 0 нет	60
K (передача без подтверждения)	12
W (прием без подтверждения)	8
T0 (с) тайм-аут при установлении соединения	30
T1 (с) тайм-аут при отправке или тестировании APDU	15
T2 (с) тайм-аут для подтверждения если нет данных	10
T3 (с) тайм-аут долгого простоя	20
Использовать резервные каналы	false

Рисунок 3.3 – Настройки Multi-Protocol MasterOPC

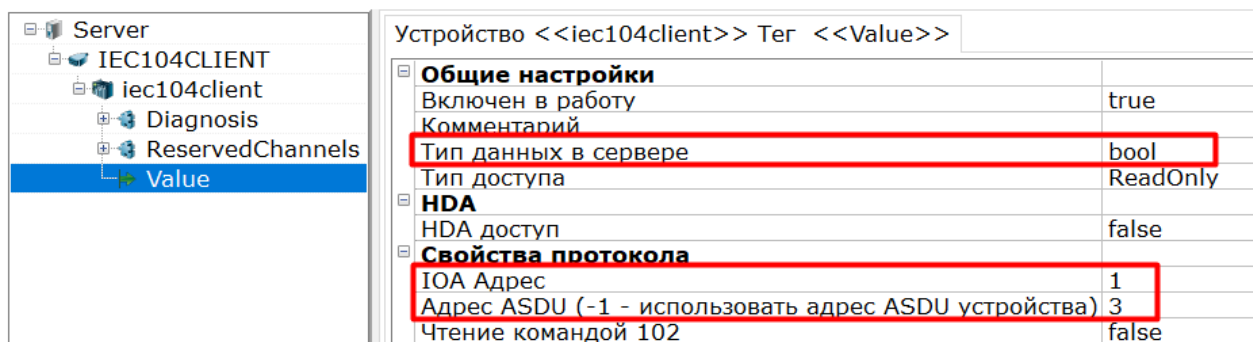


Рисунок 3.4 – Настройки канала Multi-Protocol MasterOPC

6. Запустить программы на ПЛК и OPC-сервере. Корректный обмен изображен на рисунках ниже.

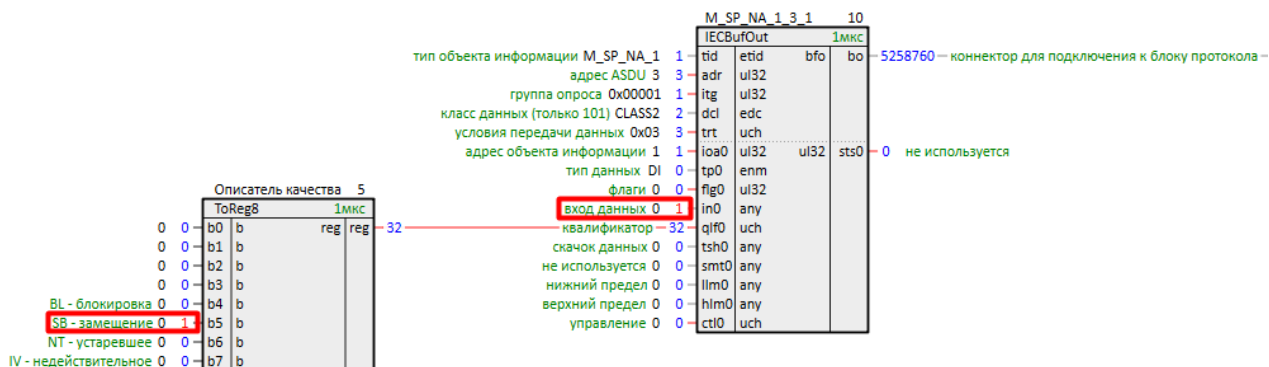


Рисунок 3.5 – Корректный обмен. Сервер

Идентификатор	Регион	Адрес в р...	Значение	Качество	Время	Тип в с...	Доступ	Комментарий
IEC104CLIENT.iec104client.Value	3	1	true	LOCAL_OVERRIDE	2024-04-12 11:51:30.678	bool	ReadOnly	

Рисунок 3.6 – Корректный обмен. Клиент

Другой пример – отправка объекта информации с типом M\_ME\_TF\_1 (ТИТ с описателем качества и меткой времени). Следует выполнить действия:

1. Добавить выходной буфер *IECBufOut*.
2. Установить  $tid = M\_ME\_TF\_1$ , адрес ASDU  $adr = 3$ , адрес объекта информации  $ioa = 2$ .  
Тип данных  $tp$  установить равным **AI**, так как  $M\_ME\_TF\_1$  – величина с плавающей запятой.
3. Условия передачи данных  $trt$  установить равным **0x11**: **бит 0** – спорадическая передача (причина передачи **3**), **бит 4** – принудительная отправка при активации **бита 1** на входе  $ctl$ , причина отправки описывается **6 битом** на входе  $flg$  – при установке **0** используется циклическая передача (причина передачи **1**).
4. Принудительная передача активируется **битом 1** входа  $ctl$  – заведем на него выход блока *ToReg8* из библиотеки *paCore* для удобства.  
На вход  $qlf$  заведем выход с блока *ToReg8* из библиотеки *paCore* – формирователь битовой маски описателя качества. Здесь в младшем бите добавляется еще один атрибут **OV** – переполнение.
5. На входе  $flg$  активировать **бит 0** – учет мертвой зоны на входе  $tsh$ . Установим  $tsh = 5$ .
6. Установить также нижний и верхний пределы равными, соответственно,  $llm = 50$  и  $hlm = 200$ .



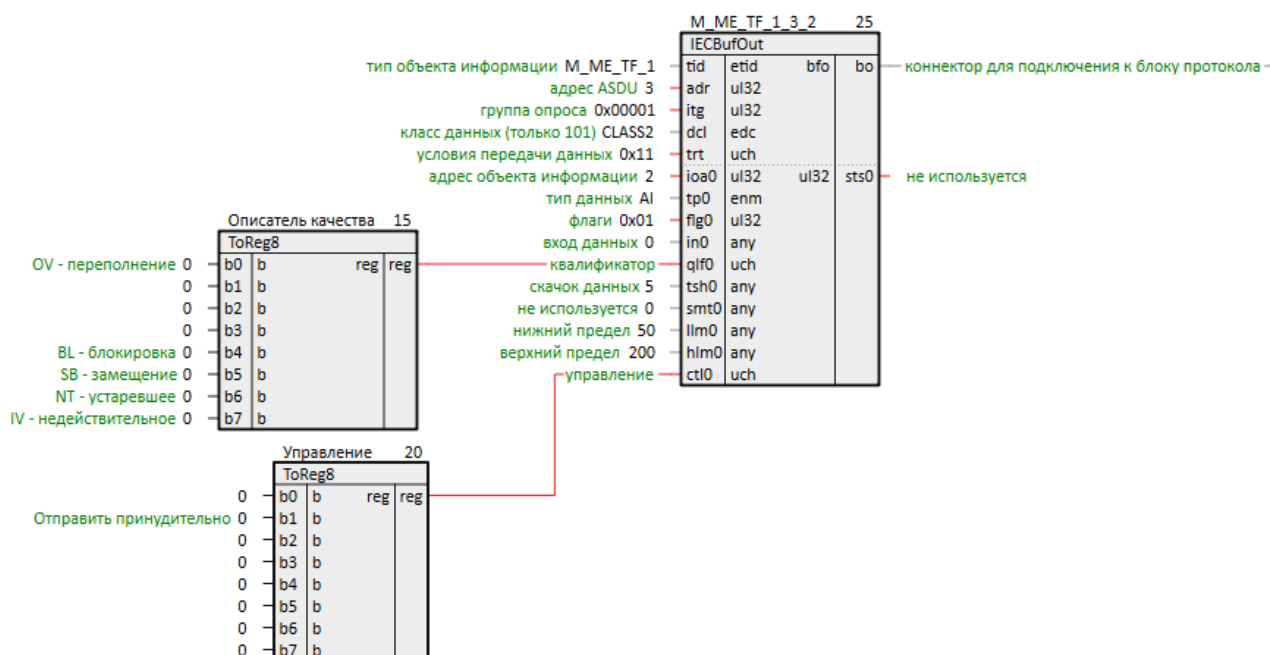


Рисунок 3.7 – Настройка IECBufOut. M\_ME\_TF\_1

7. Добавить коннектор для подключения выходного буфера *IECBufOut* у блока *IEC104Server*.
8. Подключить *IECBufOut* к коннектору блока *IEC104Server*.
9. Настроить прием ТИТ в клиенте:

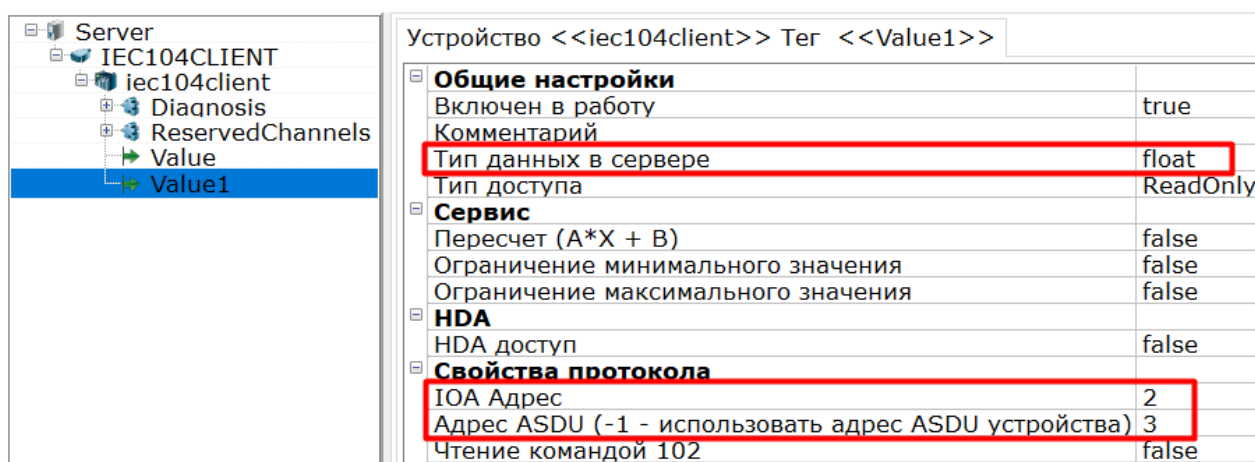


Рисунок 3.8 – Настройки OPC-сервера протокола 5-104

10. Запустить программы на ПЛК и OPC-сервер. Корректный обмен отображен на рисунках ниже.



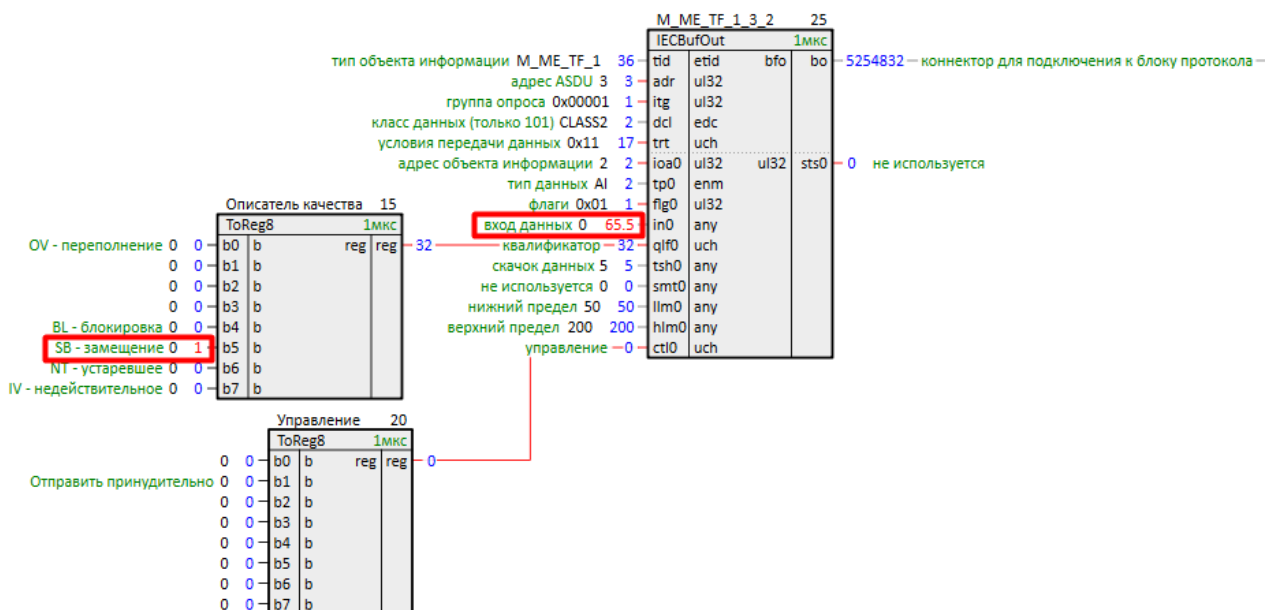


Рисунок 3.9 – Корректный обмен. Сервер

Идентификатор	Регион	Адрес в р...	Значение	Качество	Время	Тип в с...	Доступ	Комментарий
IEC104CLIENT.iec104client.Value1	3	2	65.50000	LOCAL_OVERRIDE	2024-04-12 12:00:13.931	float	ReadOnly	

Рисунок 3.10 – Корректный обмен. Клиент

11. Если после этого передать на вход значение 70 – значение клиенту не передается, так как не была преодолена мертвая зона tsh = 5.

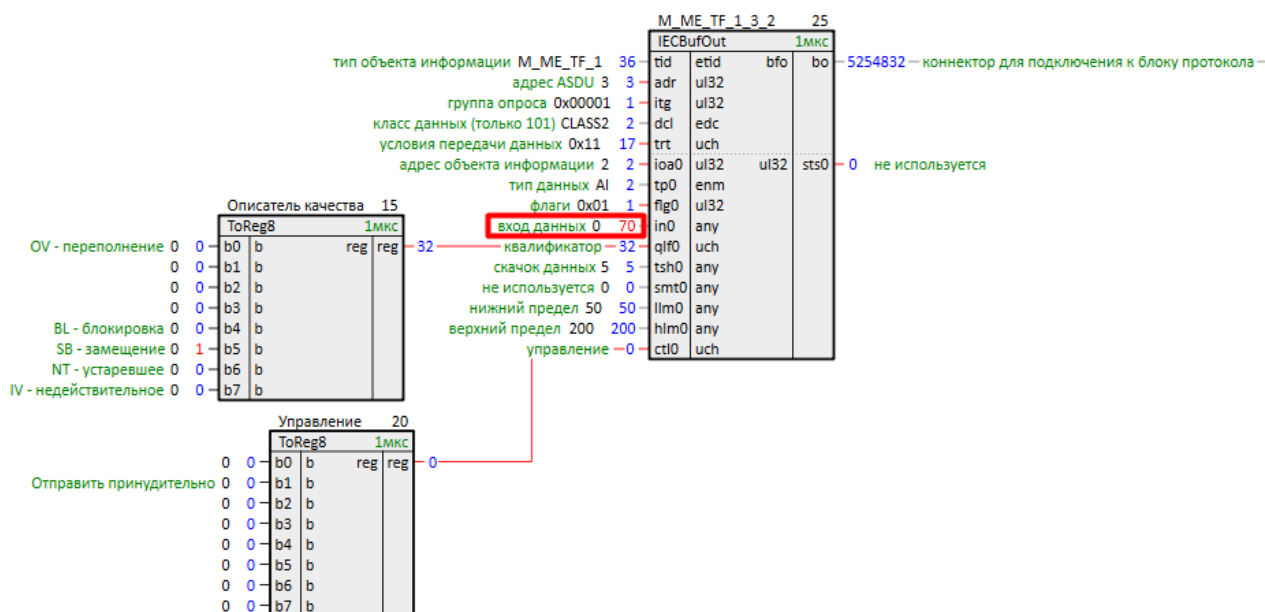


Рисунок 3.11 – Влияние параметра tsh. Сервер

Идентификатор	Регион	Адрес в р...	Значение	Качество	Время	Тип в с...	Доступ	Комментарий
IEC104CLIENT.iec104client.Value1	3	2	65.50000	LOCAL_OVERRIDE	2024-04-12 12:00:13.931	float	ReadOnly	

Рисунок 3.12 – Влияние параметра tsh. Клиент

Для принудительной отправки следует установить бит 1 на входе ctl.

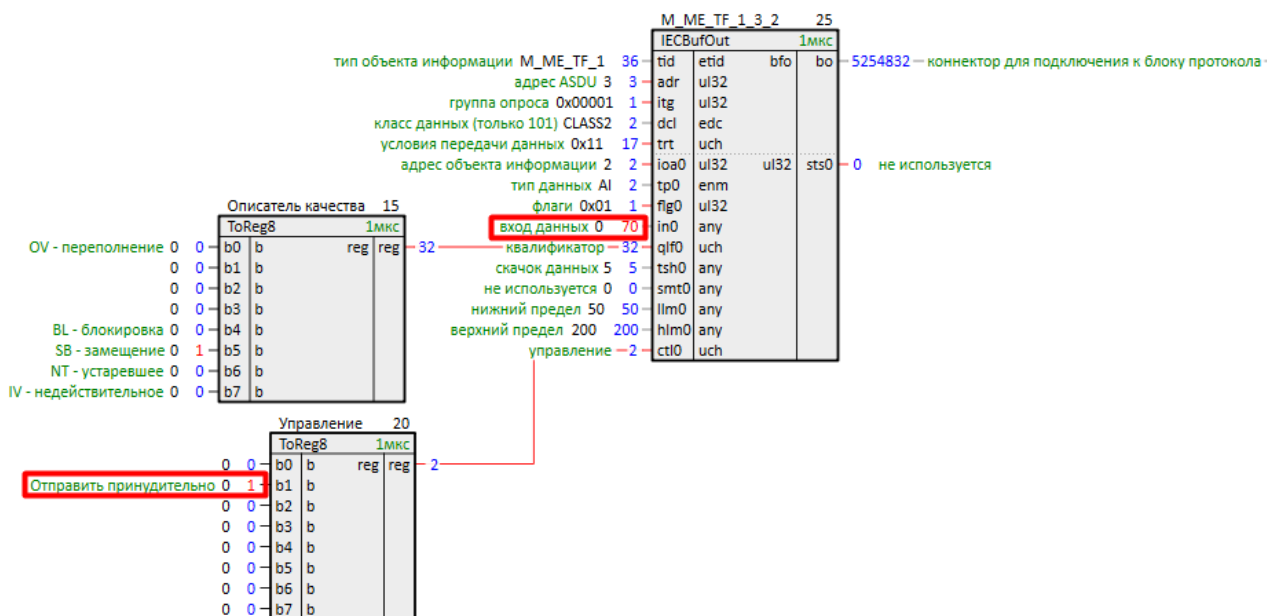


Рисунок 3.13 – Принудительная отправка. Сервер

Идентификатор	Регион	Адрес в р...	Значение	Качество	Время	Тип в с...	Доступ	Комментарий
IEC104CLIENT.Iec104client.Value1	3	2	70	LOCAL_OVERRIDE	2024-04-12 12:07:18.262	float	ReadOnly	

Рисунок 3.14 – Принудительная отправка. Клиент

Значение передалось клиенту, но без метки времени, что видно в логе клиента (**M\_ME\_NC\_1** – без метки времени), так как стандарт предписывает не передавать метку времени при циклической передаче.

```
2024-04-12 12:07:18.262 IEC104CLIENT.iec104client:--> 014: 0d 01 01 00 03 00 02 00 00 00 00 8c 42 20
2024-04-12 12:07:18.261 IEC104CLIENT.iec104client:--> CA 00003 TypeID M_ME_NC_1(013) NUM 1
```

Рисунок 3.15 – Лог клиента

- Для того, чтобы метка времени передавалась при циклической передаче следует установить **бит 5** в параметре **cfg** блока **IEC104Server** – передавать метку времени при отправке данных с причиной передачи: циклические.

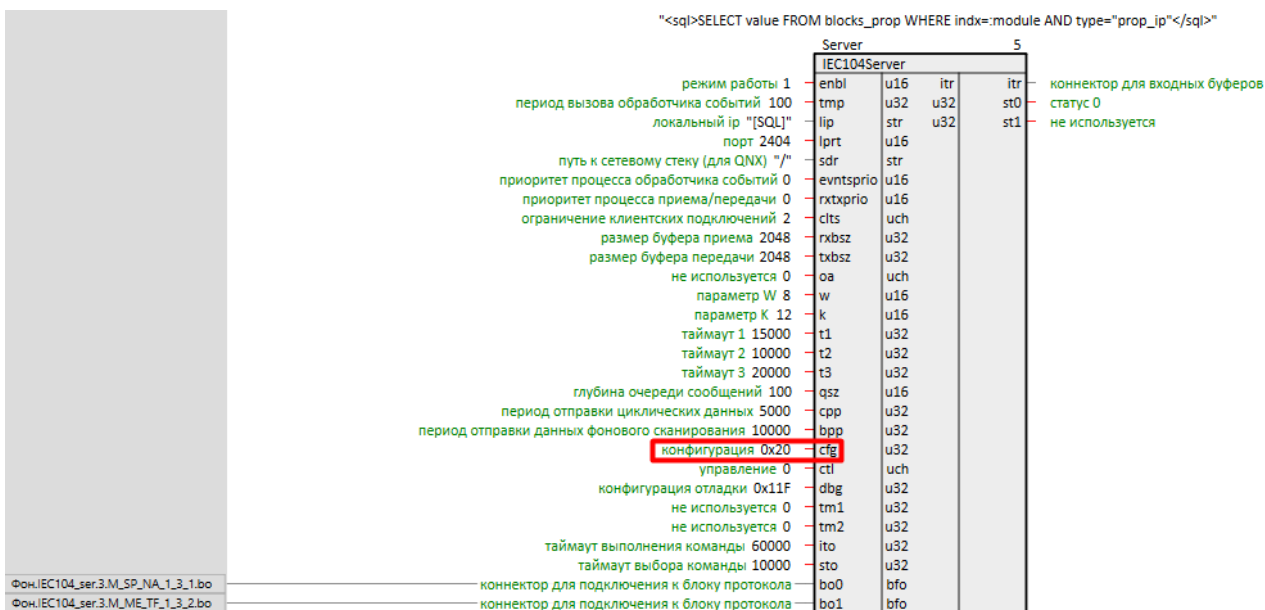


Рисунок 3.16 – Передача метки времени при циклической передаче

Теперь при принудительной отправке метка времени также будет передаваться (**M\_ME\_TF\_1** – с меткой времени).

2024-04-12 13:04:39.247 IEC104CLIENT.iec104client:--> 021: 24 01 01 00 03 00 02 00 00 00 8c 42 00 54 9d 84 0d 0c 04 18  
 2024-04-12 13:04:39.247 IEC104CLIENT.iec104client:--> CA 00003 TypeID M\_ME\_TF\_1(036) NUM 1

Рисунок 3.17 – Лог клиента

13. Для учета параметров выходного буфера верхний и нижний порог – **llm** и **hlm**, следует установить бит 2 и бит 3 на входе **fig**.

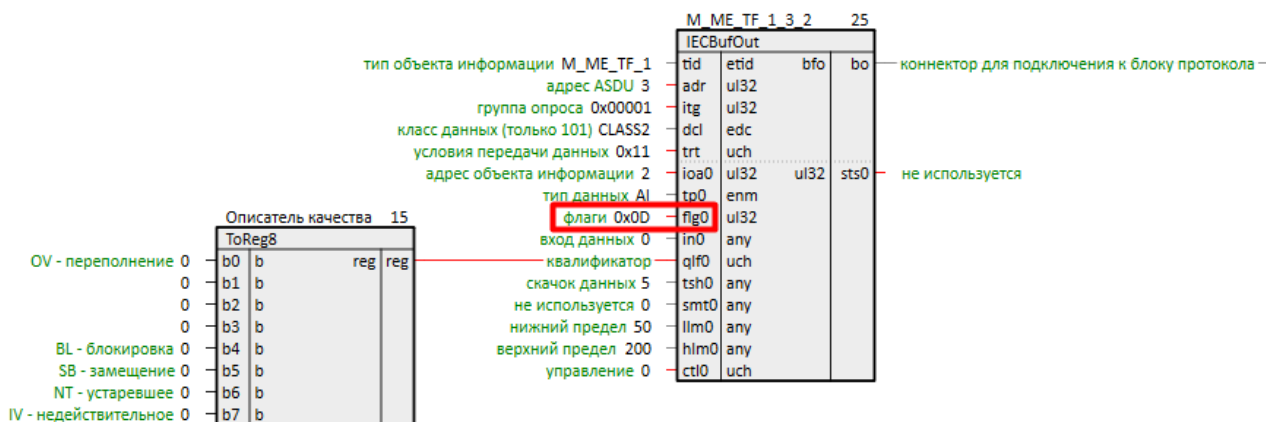


Рисунок 3.18 – Учет нижнего и верхнего порогов

Теперь при выходе значения за установленный входами **llm** и **hlm** диапазон оно будет передаваться без учета мертвой зоны, установленной на входе **tsh**.

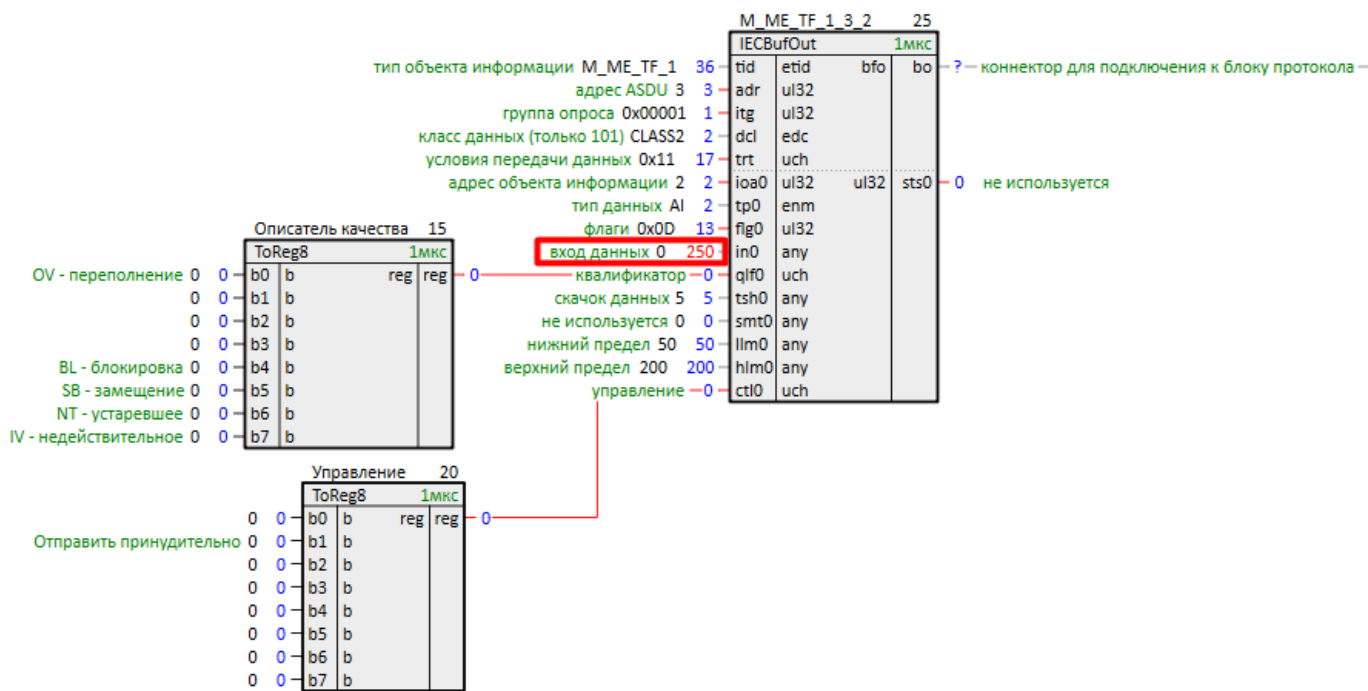


Рисунок 3.19 – – Передача при превышении hlm. Сервер

Идентификатор	Регион	Адрес в р...	Значение	Качество	Время	Тип в с...	Доступ	Комментарий
IEC104CLIENT.iec104client.Value1	3	2	250	GOOD	2024-04-12 13:07:30.190	float	ReadOnly	

Рисунок 3.20 – Передача при превышении hlm. Клиент

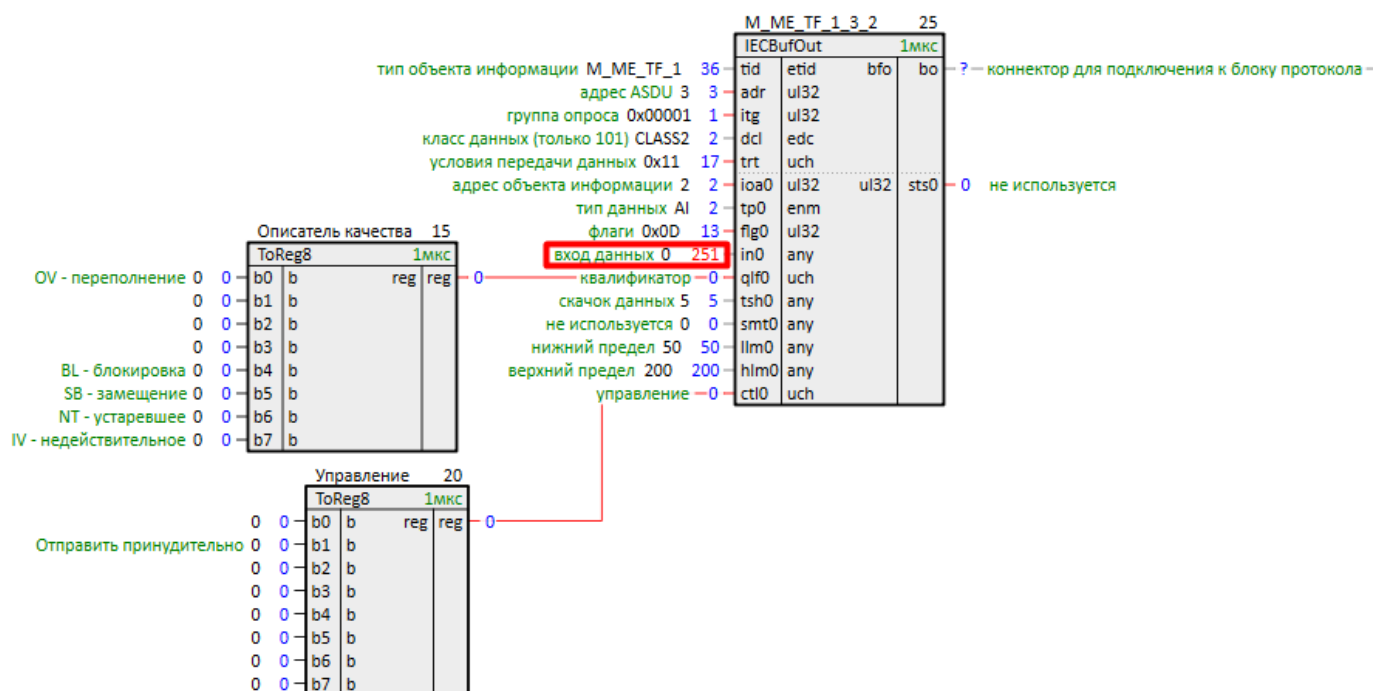


Рисунок 3.21 – Передача при превышении hlm. Сервер

Идентификатор	Регион	Адрес в р...	Значение	Качество	Время	Тип в с...	Доступ	Комментарий
IEC104CLIENT.Iec104client.Value1	3	2	251	GOOD	2024-04-12 13:09:38.715	float	ReadOnly	

Рисунок 3.22 – Передача при превышении hlm. Клиент

### 3.1.2 Прием информации о процессе в направлении управления (С)

Следующий пример – обмен данными в направлении управления (метка **С** – см. Приложение А).

Рассмотрим прием объекта информации с типом **C\_SC\_NA\_1** – однопозиционная команда ТУ.

1. Добавить на любую страницу места работы **Фон** входной буфер **IECBufIn**.
2. Установить **tid = C\_SC\_NA\_1**, адрес ASDU **adr = 3**, адрес объекта информации **ioa = 3**.

Тип данных **tp** установим равным **DO**, так как **C\_SC\_NA\_1** передает значение «включить/выключить» в младшем бите.

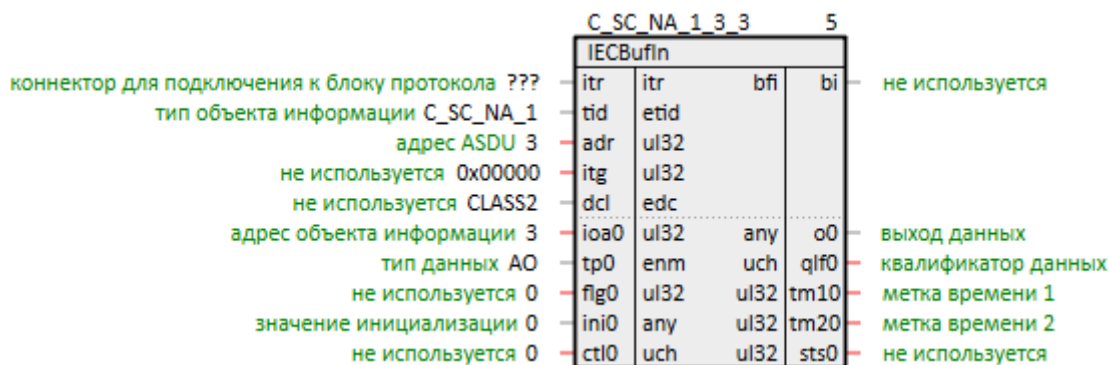


Рисунок 3.23 – Настройка IECBufIn. C\_SC\_NA\_1

3. Подключить **IECBufIn** к коннектору блока **IEC104Server**.
4. Настроить команду в **Multi-Protocol MasterOPC**. Можно также запустить в качестве клиента проект, рассмотренный в разделе 3.2, на виртуальном контроллере с помощью панели отладки.

Устройство <<iec104client>> Тер <<Command>>	
<b>Общие настройки</b>	
Включен в работу	true
Комментарий	
Тип данных в сервере	bool
Тип доступа	WriteOnly
<b>HDA</b>	
HDA доступ	false
<b>Дополнительные настройки</b>	
Чтение сразу после записи	false
<b>Свойства протокола</b>	
IOA Адрес	3
Адрес ASDU (-1 - использовать адрес ASDU устройства)	3
Тип команды	45:C SC NA 1 Single
Длительность команды	0 = no additional
Select(true)/Execute(false)	false

Рисунок 3.24 – Настройки команды Multi-Protocol MasterOPC

5. Запустить программы на ПЛК и OPC-сервере. Корректный обмен изображен на рисунках ниже.

Идентификатор	Регион	Адрес в р...	Значение	Качество	Время	Тип в с...	Доступ	Комментарий
IEC104CLIENT.Iec104client.Comm...	3	3	true	GOOD	2024-04-12 13:18:55.758	bool	WriteOnly	

Рисунок 3.25 – Корректный обмен. Клиент

IECBufIn	1	3	3	5
itr	itr	bfi	bi	не используется
тип объекта информации C_SC_NA_1	45	tid	etid	
адрес ASDU 3	3	adr	ul32	
не используется 0x000000	0	itg	ul32	
не используется CLASS2	2	dcl	edc	
адрес объекта информации 3	3	ioa0	ul32 any o0	1 выход данных
тип данных DO	1	tp0	enm uch qlf0	0 квалификатор данных
не используется 0	0	flg0	ul32 ul32 tm10	0 метка времени 1
значение инициализации 0	0	ini0	any ul32 tm20	0 метка времени 2
не используется 0	0	cti0	uch ul32 sts0	0 не используется

Рисунок 3.26 – Корректный обмен. Сервер

Для приема объекта информации с типом **C\_SE\_TC\_1** (команда уставки с меткой времени) следует:

1. Рассмотреть прием объекта информации с типом **C\_SE\_TC\_1** – команда уставки с меткой времени.
2. Добавить входной буфер **IECBufIn**.
3. Установить **tid = C\_SE\_TC\_1**, адрес ASDU **adr = 3**, адрес объекта информации **ioa = 4**.  
Тип данных **tp** установим равным **AO**, так как **C\_SE\_TC\_1** имеет формат с плавающей запятой.
4. Для расшифровки метки времени соединить выходы **tm1** и **tm2** с входами блока **IECTransTime**.

C_SE_TC_1 3 4 10					
коннектор для подключения к блоку протокола ???	itr	itr	bfi	bi	не используется
тип объекта информации C_SE_TC_1	tid	etid			
адрес ASDU 3	adr	ul32			
не используется 0x000000	itg	ul32			
не используется CLASS2	dcl	edc			
адрес объекта информации 4	ioa0	ul32 any o0			выход данных
тип данных AO	tp0	enm uch qlf0			квалификатор данных
не используется 0	flg0	ul32 ul32 tm10			метка времени 1
значение инициализации 0	ini0	any ul32 tm20			метка времени 2
не используется 0	cti0	uch ul32 sts0			не используется

Метка времени 15	
tm1	ul32 u64 msec
tm2	ul32 b IV str

Рисунок 3.27 – Настройка IECBufIn. C\_SE\_TC\_1

5. Подключить **IECBufIn** к коннектору блока **IEC104Server**.
6. Настроить команду в Multi-Protocol MasterOPC:

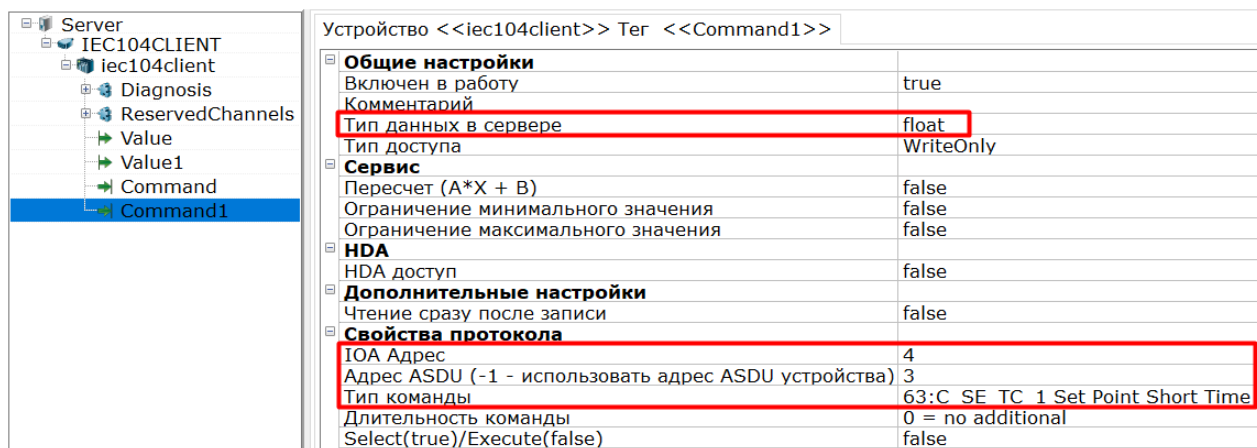


Рисунок 3.28 – Настройки команды Multi-Protocol MasterOPC

7. Запустить программы на ПЛК и OPC-сервере. Корректный обмен показан на рисунках ниже.

Идентификатор	Регион	Адрес в р...	Значение	Качество	Время	Тип в с...	Доступ	Комментарий
IEC104CLIENT.Iec104client.Comm...	3	4	25.50000	GOOD	2024-04-12 13:31:56.389	float	WriteOnly	

Рисунок 3.29 – Корректный обмен. Клиент

C	SE	TC	1	3	4	10	IECBufin	1мкс
итр	итр	бfn	bi	?	?	не используется		
tid	etid							
adr	ul32							
itg	ul32							
dcl	edc							
ioa0	ul32	any	o0			25.5	выход данных	
tp0	enm	uch	qlf0			0	квалификатор данных	
flg0	ul32	tm10				3520932581	метка времени 1	
ini0	any	ul32	tm20			398	метка времени 2	
ctl0	uch	ul32	sts0			0	не используется	

Рисунок 3.30 – Корректный обмен. Сервер

### 3.1.3 Передача упакованных данных (M\_EP\_\*)

Особенность типа информации **M\_EP\_TE\_1** (упакованное сообщение с меткой времени **CP56Время2a**) состоит в том, что в сообщении передается структура, состоящая из битовой маски и интервала времени.

Для передачи объекта информации с типом **M\_EP\_TE\_1** следует:

1. Добавить на любую страницу места работы **Фон** выходной буфер **IECBufOut**.
2. Установить **tid = M\_EP\_TE\_1**, адрес ASDU **adr = 3**, адрес объекта информации **ioa = 5**.
3. Тип данных **tp** установить равным **uLX**, т.к. входные данные для данного типа ASDU структурированы.

Условия передачи данных **trt** установить равным **0x01**: бит **0** – спорадическая передача (причина передачи **3**) (см. Приложение Б).

4. Для формирования структуры завести на вход **in** выход **struct** блока **IECIntFromEP**. Установка бита **0** на входе **cfg** этого блока определяет, что спонтанная передача происходит только при изменении значения на входе **state**.
5. Для формирования информационного байта на вход **IECIntFromEP** завести выход блока **ToReg8** из библиотеки **paCore**.

На вход **qlf** также завести выход с блока **ToReg8** – формирование битовой маски описателя качества. Здесь добавляется атрибут **EI** – действительность интервала длительности.



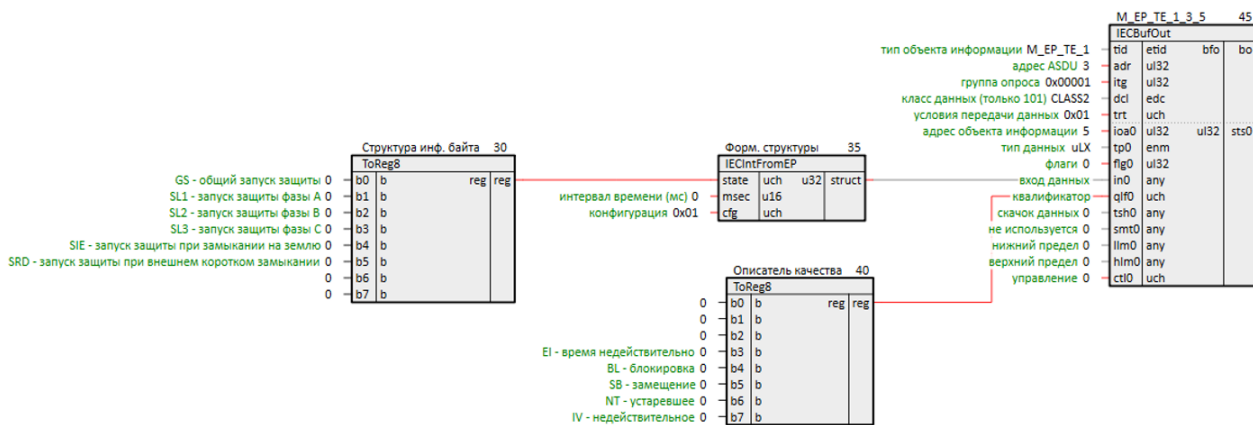


Рисунок 3.31 – Настройка IECBufOut. M\_EP\_TE\_1

6. Добавить коннектор для подключения выходного буфера *IECBufOut* у блока *IEC104Server*.
7. Подключить *IECBufOut* к коннектору блока *IEC104Server*.

В качестве клиента настроить виртуальный контроллер **Полигон**. Можно также запустить в качестве клиента проект, рассмотренный в [разделе 3.2](#), на виртуальном контроллере с помощью панели отладки.

8. Для расшифровки структуры использовать блок *IECEPFromInt*.

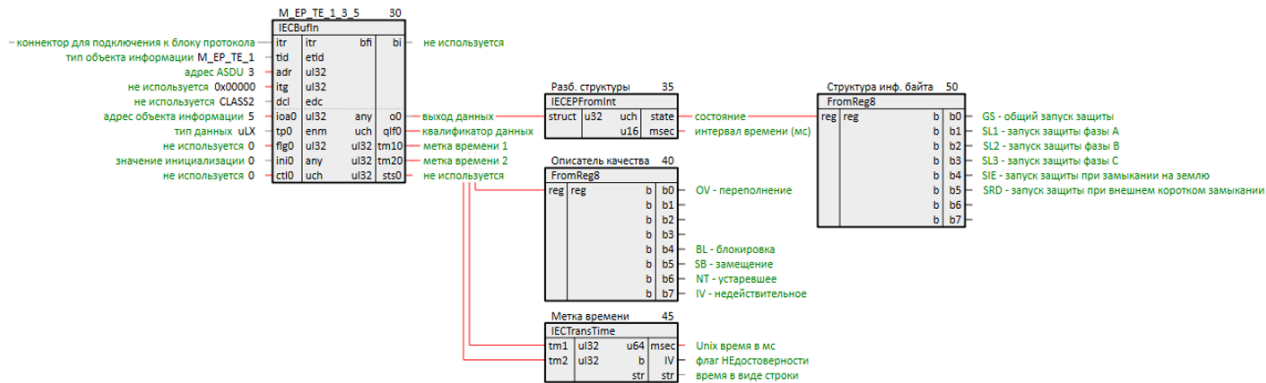


Рисунок 3.32 – Настройки клиента ПА

9. Запустить программы на ПЛК и виртуальном контроллере. Корректный обмен изображен на рисунках ниже.

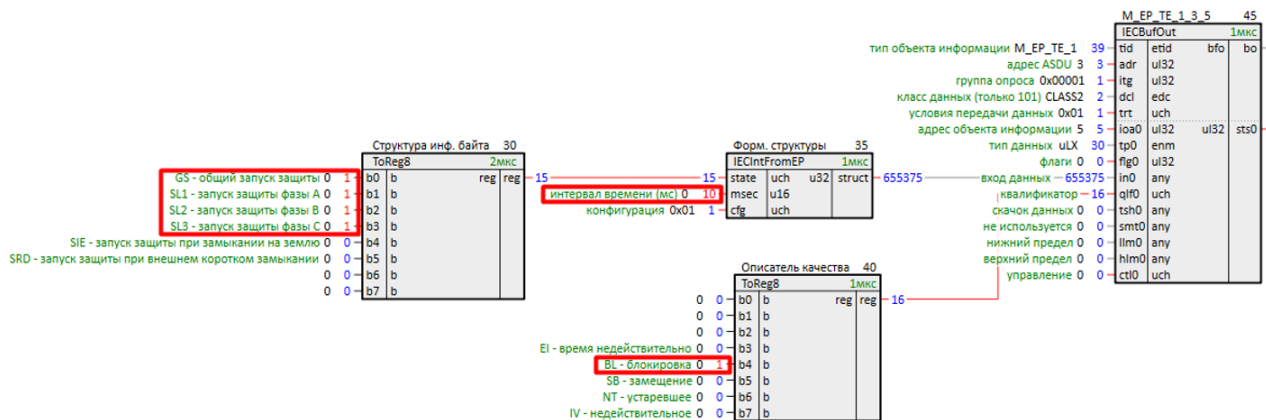
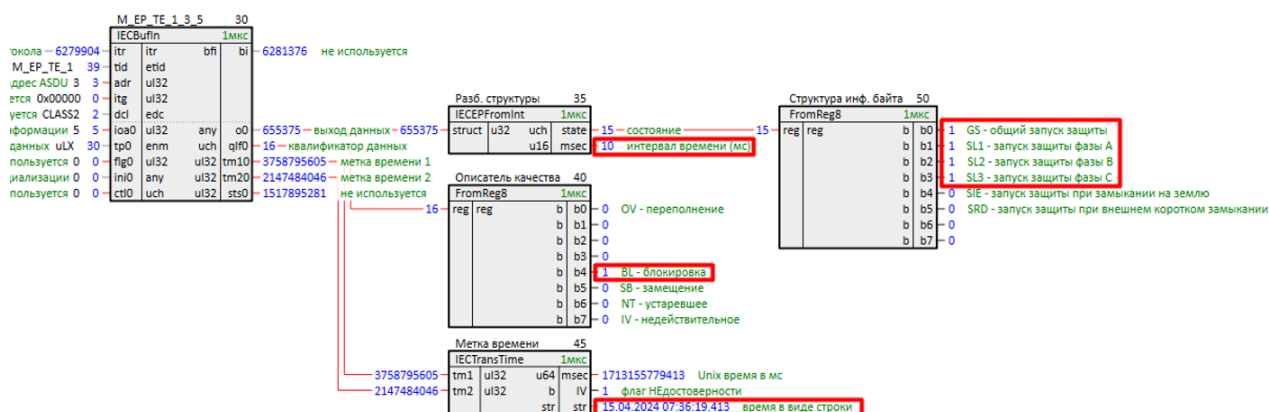


Рисунок 3.33 – Корректный обмен. Сервер



### 3.1.4 Передача интегральных сумм (M\_IT\_\*)

Особенность типа информации **M\_IT\_TB\_1** (интегральная сумма с меткой времени **CP56Время2а**) состоит в том, что в сообщении передается знаковое целое (в 4-х байтах).

Для передачи объекта информации с типом **M\_IT\_TB\_1** следует:

1. Добавить на любую страницу места работы **Фон** выходной буфер **IECBufOut**.
2. Установить **tid = M\_IT\_TB\_1**, адрес ASDU **adr = 3**, адрес объекта информации **ioa = 6**.
3. Тип данных **tp** установить равным **LX**.

Условия передачи данных **trt** установить равным **0x01**: бит 0 – спорадическая передача (причина передачи 3) (см. Приложение Б).

4. Для формирования структуры соединить связью вход **qlf** и выход **out** блока **IECITSQOut**.



5. Добавить коннектор для подключения выходного буфера **IECBufOut** у блока **IEC104Server**.
6. Подключить **IECBufOut** к коннектору блока **IEC104Server**.
7. В качестве клиента настроить виртуальный контроллер **Полигон**. Можно также запустить в качестве клиента проект, рассмотренный в разделе 3.2, на виртуальном контроллере с помощью панели отладки.
8. Для расшифровки структуры используется блок **IECITSQIn**.



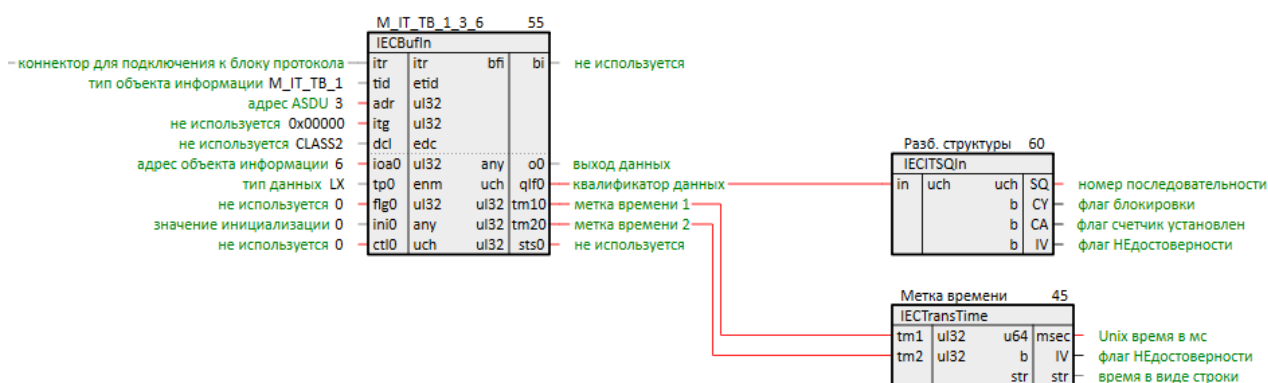


Рисунок 3.36 – Настройки клиента ПА

9. Запустить программы на ПЛК и виртуальном контроллере. Корректный обмен показан на рисунках ниже.

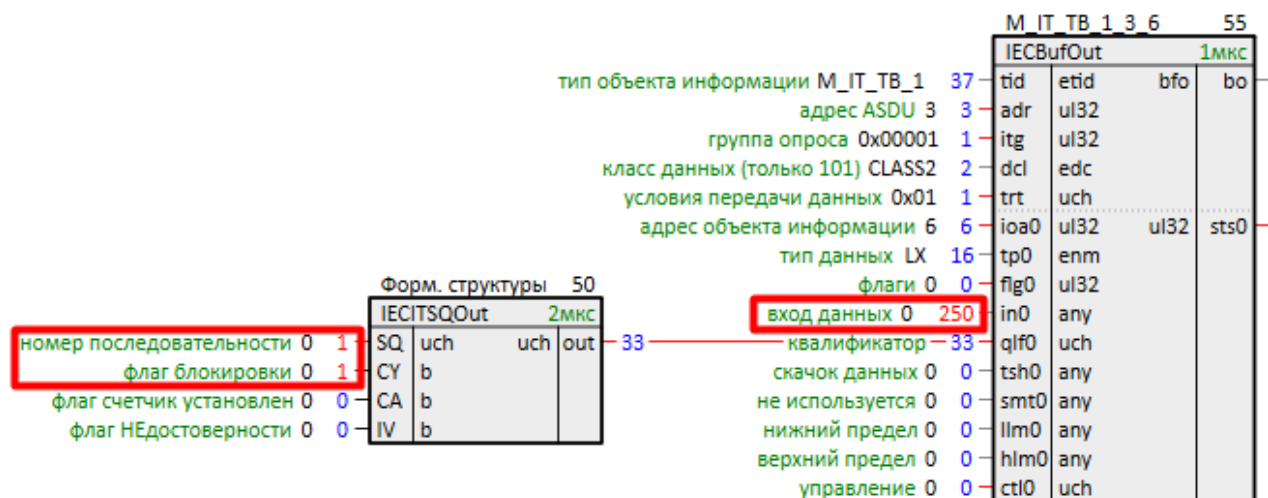


Рисунок 3.37 – Корректный обмен. Сервер

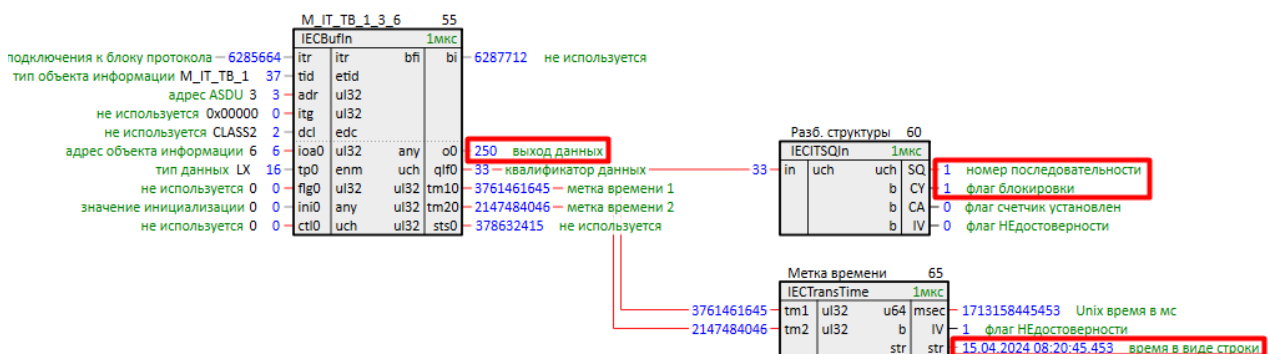


Рисунок 3.38 – Корректный обмен. Клиент

### 3.2 Настройка обмена в режиме клиента протокола 5-104

В данном примере ПЛК210 будет выступать в роли контролирующей станции (клиента).

Пример доступен для скачивания по [ссылке](#). Пароль для доступа к отладчику – 1.

Для настройки обмена с ПЛК в режиме клиента следует:

1. Добавить на любую страницу места работы **Фон** блок **IEC104uni**.
2. Соединить его с блоком **TcpIpCIA** из библиотеки **paCore**.
3. На вход **lip** подать IP-адрес контроллера в виде SQL-запроса. Запрос IP-адреса (prop\_ip):  
 "<sql>SELECT value FROM blocks\_prop WHERE indx=:module AND type="prop\_ip"</sql>"
4. На вход **ip** подать IP адрес подчиненного контроллера в виде SQL-запроса к свойству модуля **Пользовательское свойство 00**. Запрос IP-адреса (prop\_0):

```
"<sql>SELECT value FROM blocks_prop WHERE indx=:module AND type="prop_0"</sql>"
```

### 5. На входе **prm** задать **IEC\_MASTER**.

```
"<sql>SELECT value FROM blocks_prop WHERE indx=:module AND type="prop_ip"</sql>"
```

```
"<sql>SELECT value FROM blocks_prop WHERE indx=:module AND type="prop_0"</sql>"
```

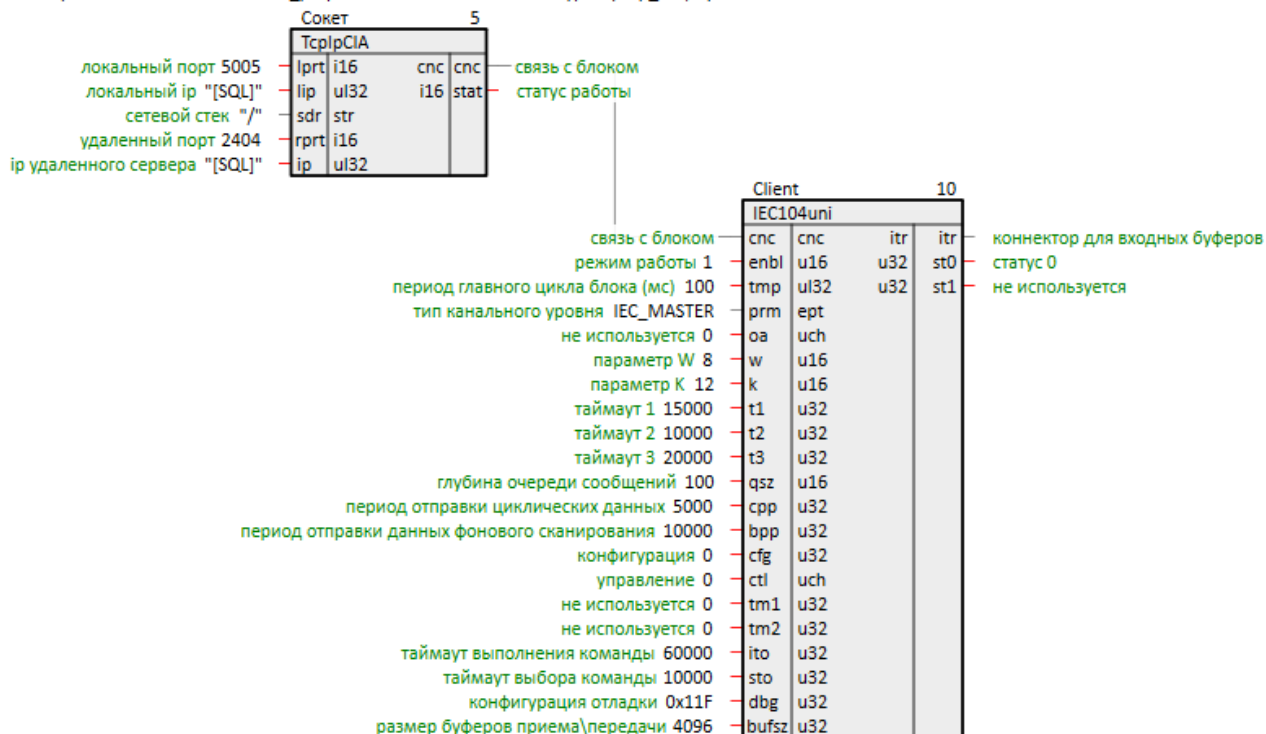


Рисунок 3.39 – Настройка IEC104uni

### 3.2.1 Прием информации о процессе в направлении контроля (M)

В этом подразделе показан обмен данными в направлении контроля (метка **M** – см. Приложение А).

Рассмотрим прием объекта информации с типом **M\_SP\_NA\_1** – одноэлементный объект информации с описателем качества.

Для приема объекта информации с типом **M\_SP\_NA\_1** (одноэлементный объект информации с описателем качества) следует:

1. Добавить на любую страницу места работы **Фон** входной буфер **IECBufIn**.



#### ПРИМЕЧАНИЕ

В данном примере настраивается передача данных в прямом направлении – от сервера к клиенту. Аналогично настраивается обмен в обратном направлении – от клиента к серверу.

2. Установить **tid = M\_SP\_NA\_1**, адрес ASDU **adr = 3**, адрес объекта информации **ioa = 1**.

Тип данных **tp** установить равным **DO**, так как у **M\_SP\_NA\_1** в младшем разряде передается **1 бит** ТС.

3. Для расшифровки описателя качества соединить связью выход **qlf** и вход блока **FromReg8** из библиотеки **paCore**.

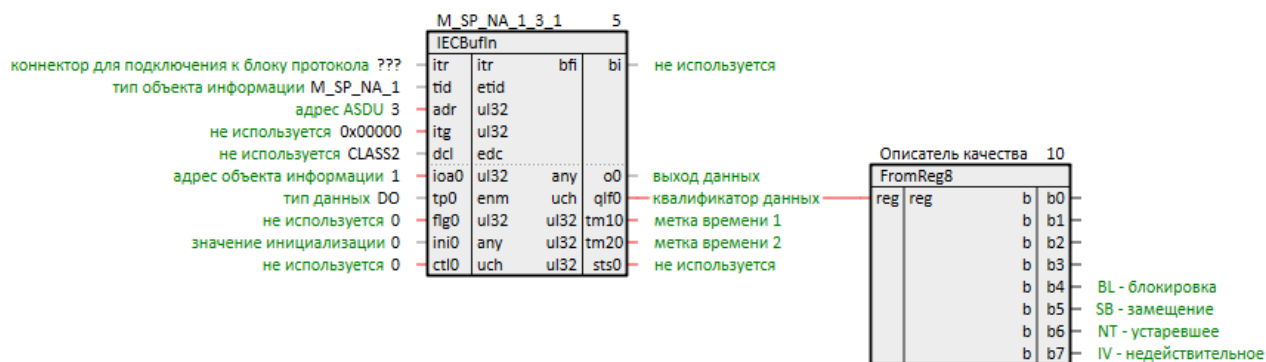


Рисунок 3.40 – Настройка IECBufin. M\_SP\_NA\_1

4. Подключить **IECBufin** к коннектору блока **IEC104uni**.

В качестве сервера использовать **MasterSCADA 4D**. Можно также запустить в качестве сервера проект, рассмотренный в [разделе 3.1](#), на виртуальном контроллере с помощью панели отладки.

5. Настройка сервера APM в MasterSCADA 4D:

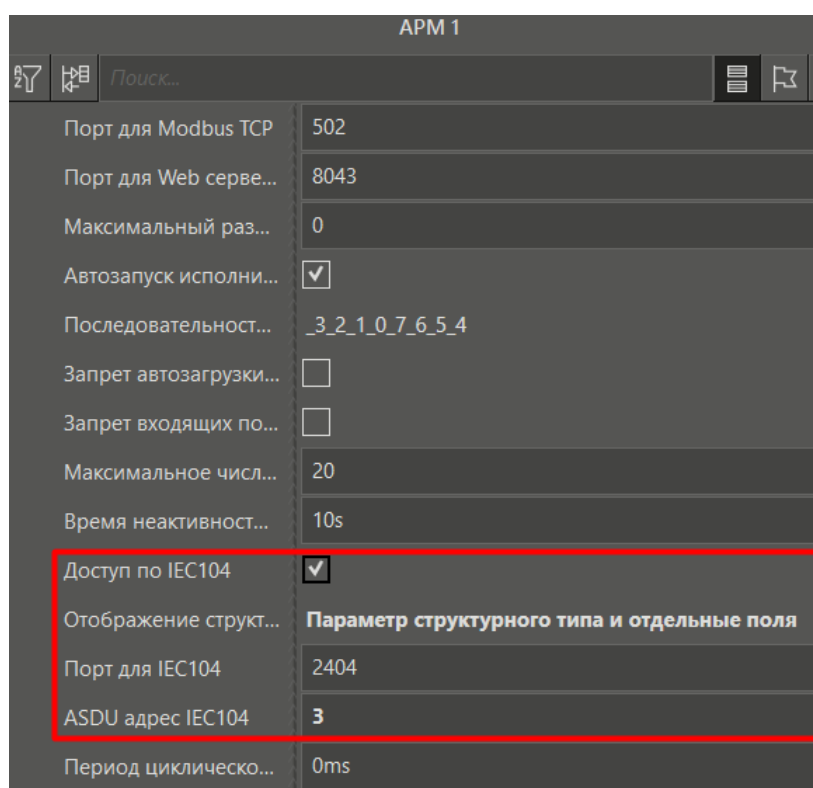


Рисунок 3.41 – Настройки сервера МЭК 104 APM

6. Добавить **Канал** во **Внешние каналы** и привязать к нему параметр.

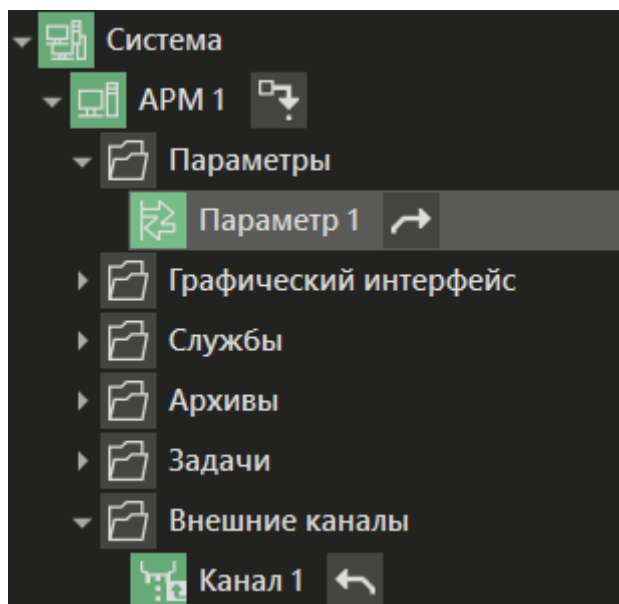


Рисунок 3.42 – Настройки канала АРМ

7. Настройки канала АРМ:

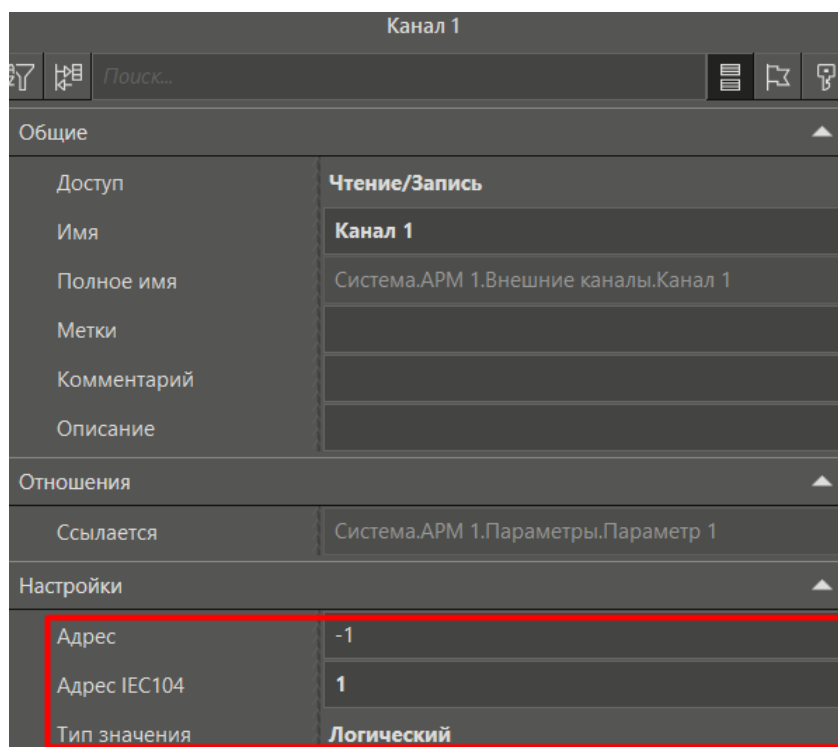


Рисунок 3.43 – Настройки канала АРМ

8. Запустить программы на ПЛК и АРМ. Корректный обмен показан на рисунках ниже.

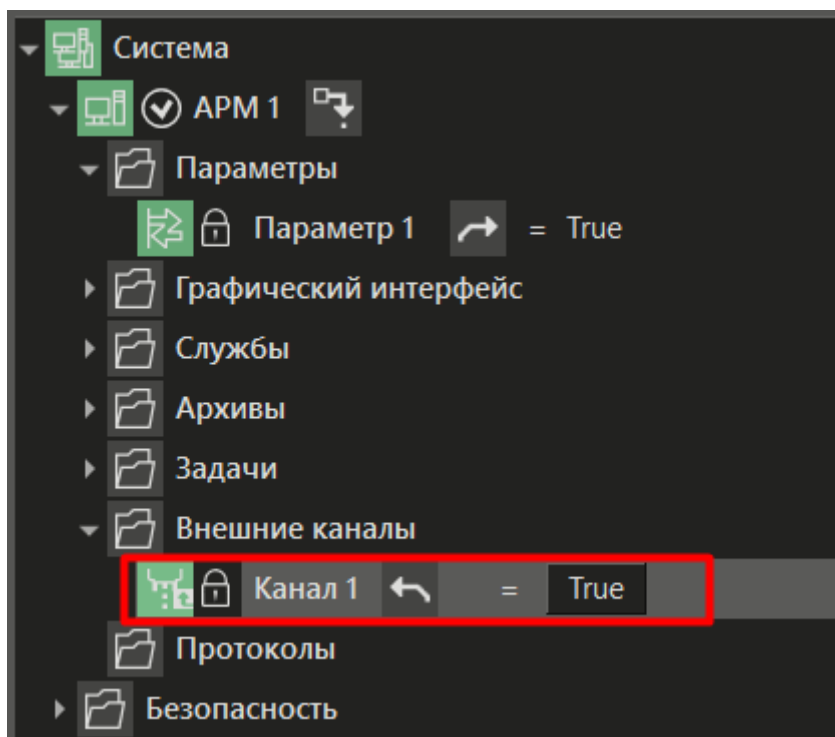


Рисунок 3.44 – Корректный обмен. Сервер

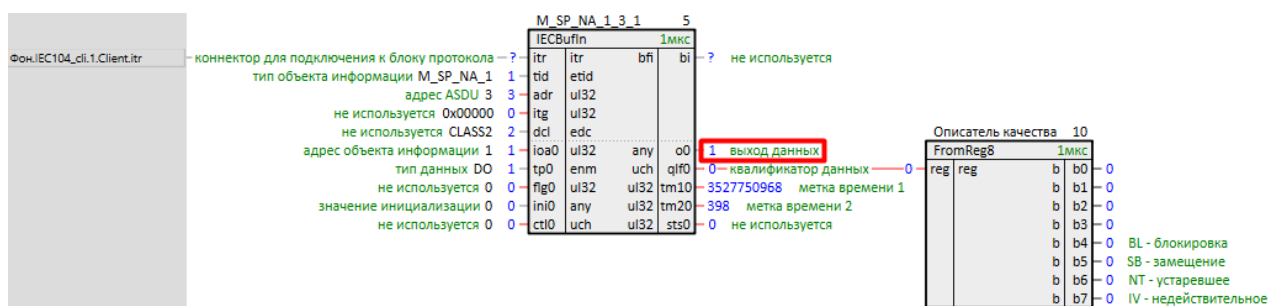


Рисунок 3.45 – Корректный обмен. Клиент

Рассмотрим прием объекта информации с типом **M\_ME\_TF\_1** – ТИ с описателем качества и меткой времени.

Для приема объекта информации с типом **M\_ME\_TF\_1** (ТИТ с описателем качества и меткой времени) следует выполнить действия:

1. Добавить выходной буфер **IECBufln**.
2. Установить **tid = M\_ME\_TF\_1**, адрес ASDU **adr = 3**, адрес объекта информации **ioa = 2**.  
Тип данных **tp** установить равным **AO**, так как **M\_ME\_TF\_1** – величина с плавающей запятой.
3. Для расшифровки описателя качества соединить связью выход **qlf** со входом блока **FromReg8** из библиотеки **paCore**. Здесь в младшем бите добавляется еще один атрибут **OV** – переполнение.
4. Для расшифровки метки времени соединим выходы **tm1** и **tm2** с входами блока **IECTransTime**.

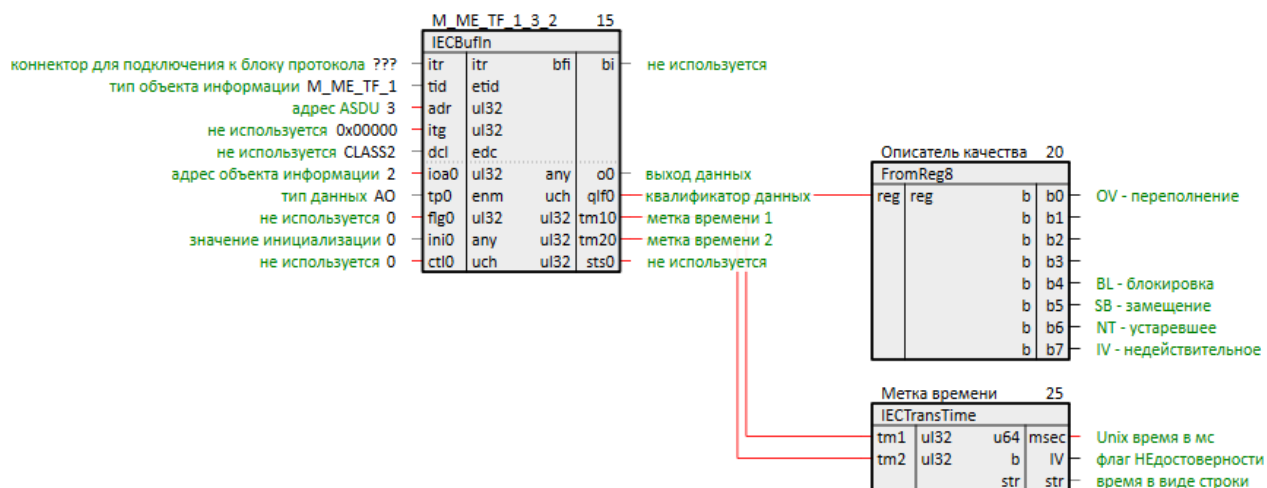


Рисунок 3.46 – Настройка IECBufin. M\_ME\_TF\_1

5. Подключить **IECBufOut** к коннектору блока **IEC104Server**.

Настройка канала APM в MasterSCADA 4D:

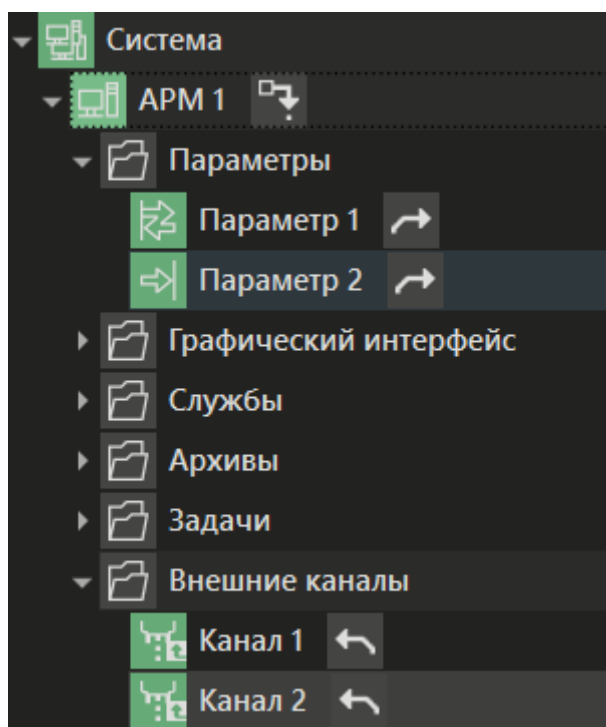


Рисунок 3.47 – Настройки канала APM

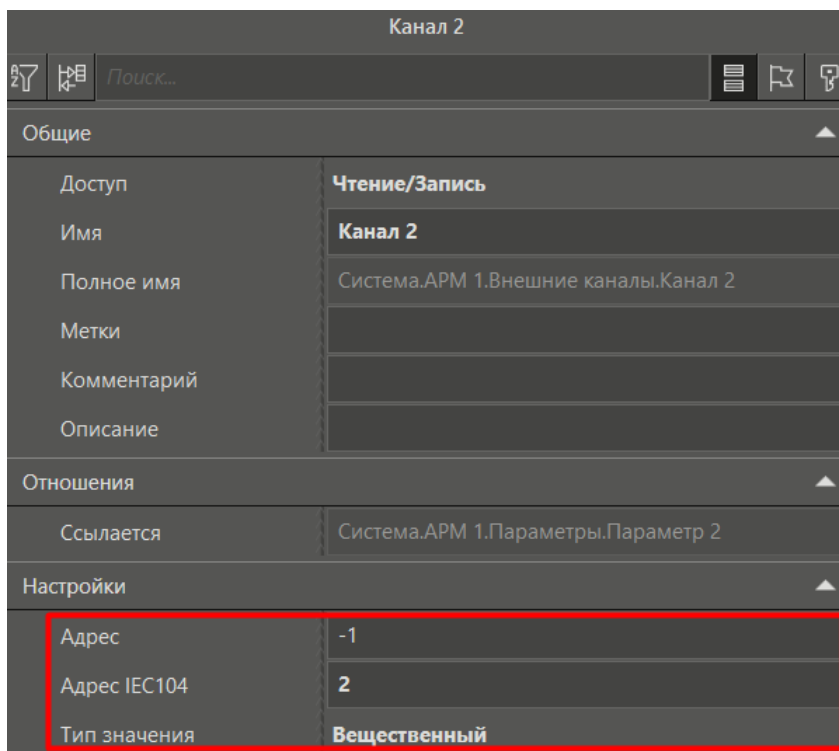


Рисунок 3.48 – Настройки канала АРМ

6. Запустить программы на ПЛК и АРМ. Корректный обмен изображен на рисунках ниже.

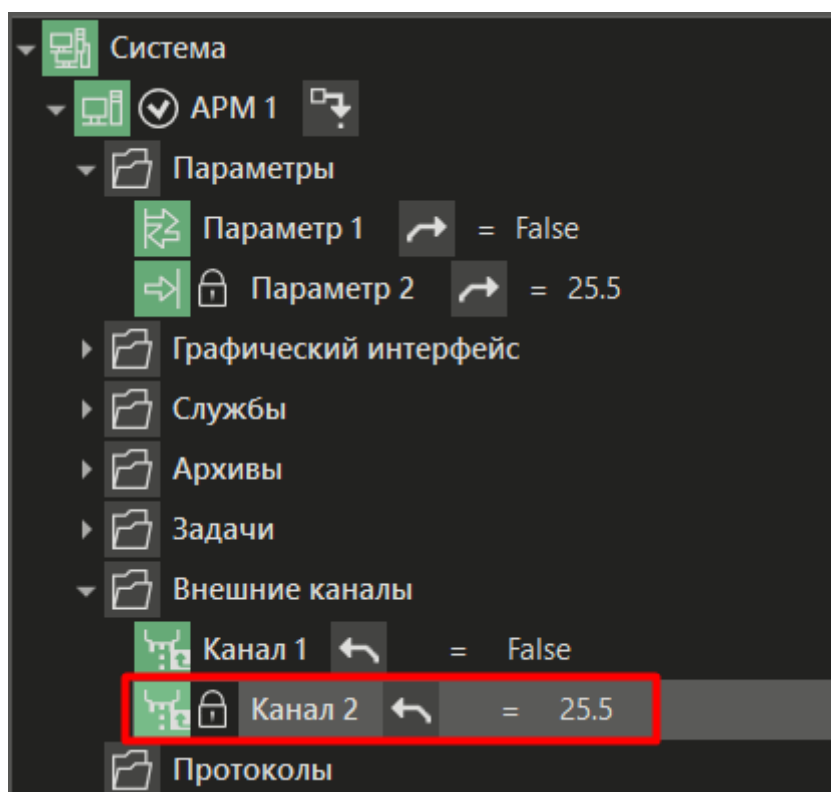


Рисунок 3.49 – Корректный обмен. Сервер

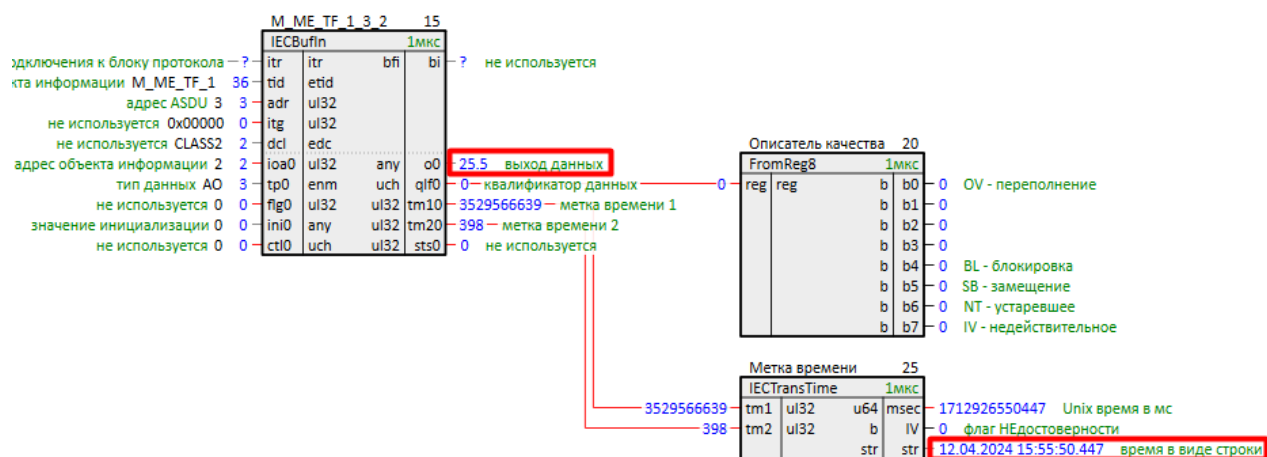


Рисунок 3.50 – Корректный обмен. Клиент

### 3.2.2 Передача информации о процессе в направлении управления (C)

Для отправки объекта информации с типом C\_SC\_NA\_1 (однопозиционная команда ТУ) в направлении управления (см. Приложение А), следует:

1. Добавить на любую страницу места работы **Фон** выходной буфер **IECBufOut**.
2. Установить **tid = C\_SC\_NA\_1**, адрес ASDU **adr = 3**, адрес объекта информации **ioa = 3**.  
Установить тип данных **tp = DI**, так как **C\_SC\_NA\_1** передает значение «включить/выключить» в младшем бите.
3. Установить условия передачи данных **trt = 0x08**: бит 3 – команда (причины передачи 6, 8, 10) (см. Приложение Б).

Передача команды активируется битом 0 входа **ctl**. Для удобства нужно связать вход **ctl** и выход блока **ToReg8** из библиотеки **paCore**.

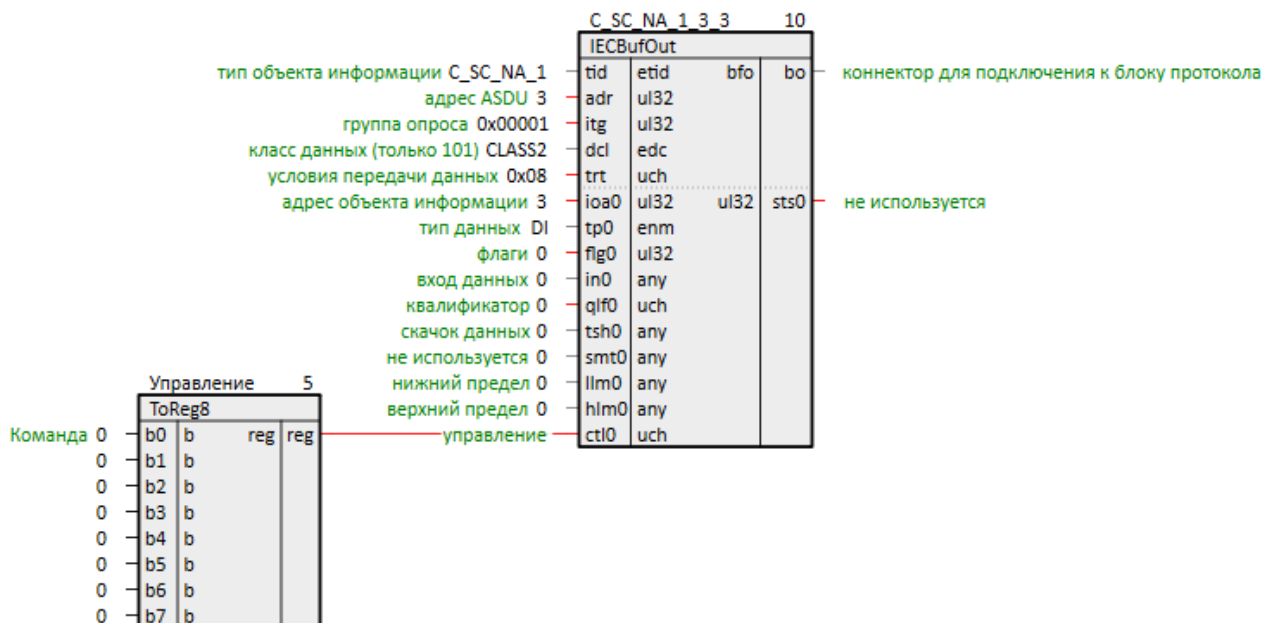


Рисунок 3.51 – Настройка IECBufOut. C\_SC\_NA\_1

4. Добавить коннектор для подключения выходного буфера **IECBufOut** у блока **IEC104uni**.
5. Подключить **IECBufOut** к коннектору блока **IEC104uni**.

В качестве сервера использовать **MasterSCADA 4D**. Можно также запустить в качестве сервера проект, рассмотренный в разделе 3.2, на виртуальном контроллере с помощью панели отладки.

6. Настройки канала АРМ:



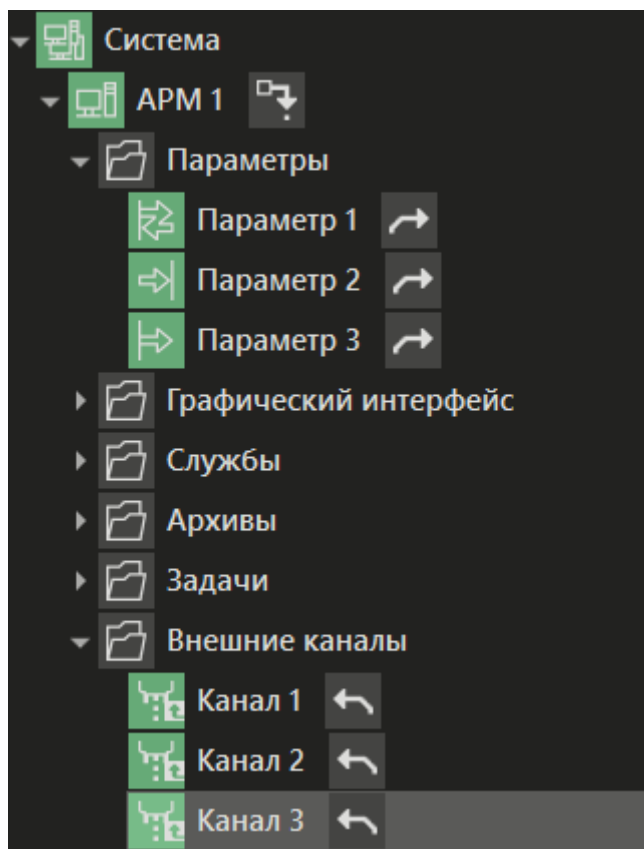


Рисунок 3.52 – Настройки канала АРМ

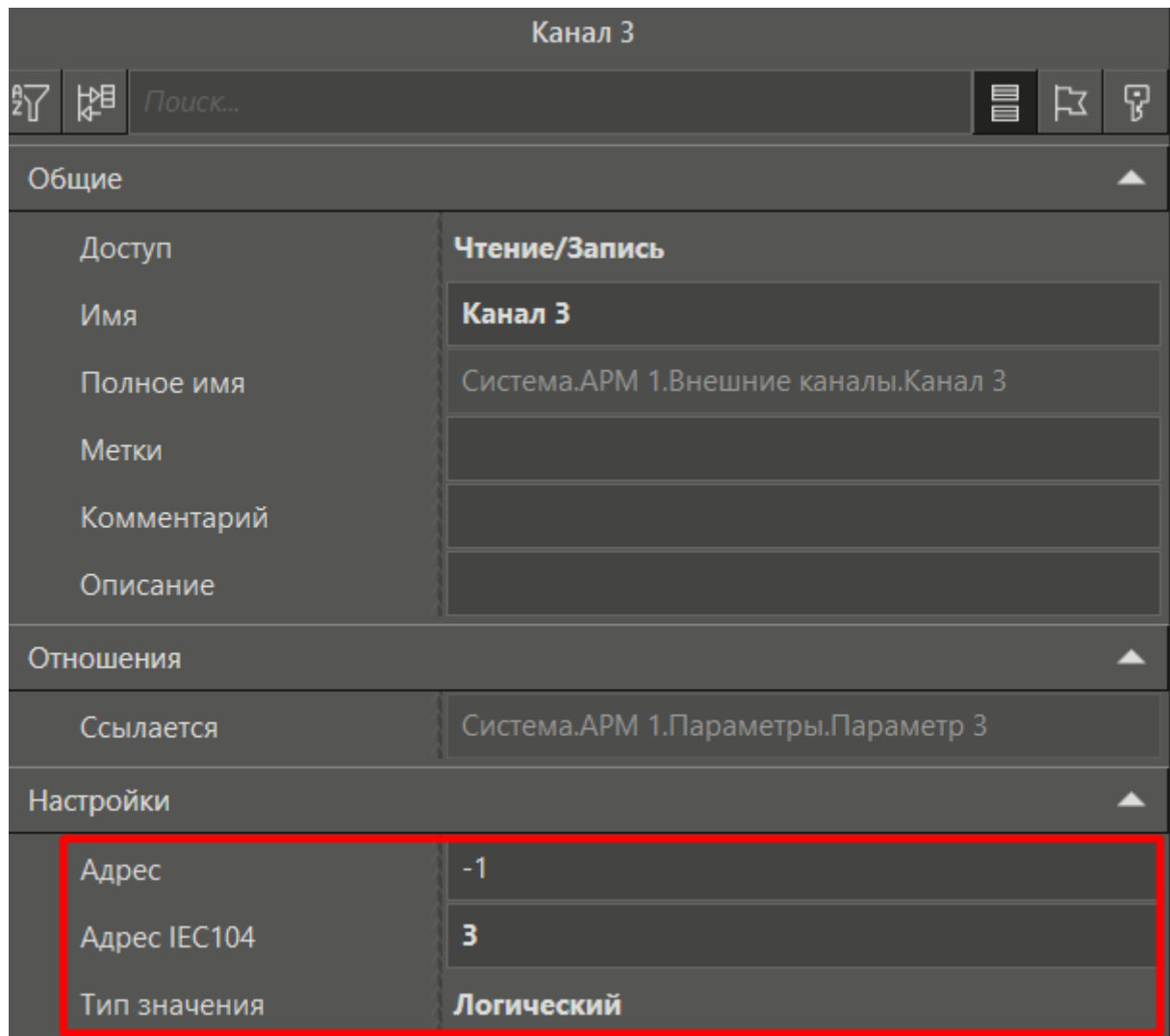


Рисунок 3.53 – Настройки канала АРМ

7. Запустить программы на ПЛК и АРМ. Корректный обмен отображен на рисунках ниже.

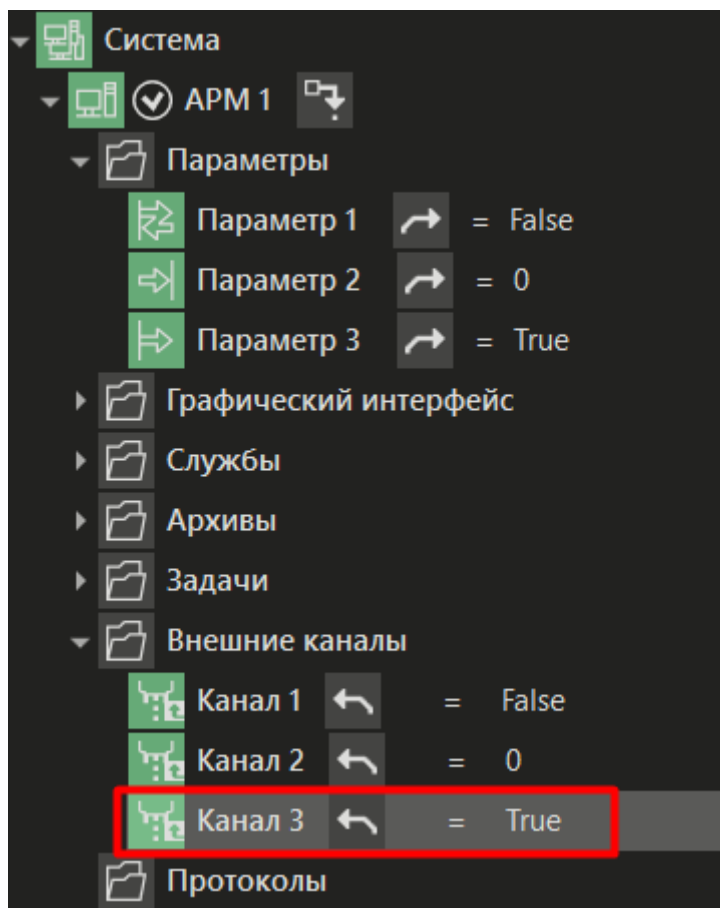


Рисунок 3.54 – Корректный обмен. Сервер

		C	SC	NA	1	3	3	10	
								IECBufOut	1мкс
тип объекта информации	C_SC_NA_1	45	tid	etid	bfo	bo	52		
адрес ASDU	3	3	adr	ul32					
группа опроса	0x00001	0	itg	ul32					
класс данных (только 101)	CLASS2	2	dcl	edc					
условия передачи данных	0x08	8	trt	uch					
адрес объекта информации	3	3	ioa0	ul32	ul32	sts0	0		
тип данных	DI	0	tp0	enm					
флаги	0	0	flg0	ul32					
<b>вход данных</b>	<b>0</b>	<b>1</b>	in0	any					
квалификатор	0	0	qlf0	uch					
скачок данных	0	0	tsh0	any					
не используется	0	0	smt0	any					
нижний предел	0	0	llm0	any					
верхний предел	0	0	hlm0	any					
управление	1	1	ctl0	uch					
Управление		5							
ToReg8		1мкс							
Команда	0	1	b0	b	reg	reg	1		
	0	0	b1	b					
	0	0	b2	b					
	0	0	b3	b					
	0	0	b4	b					
	0	0	b5	b					
	0	0	b6	b					
	0	0	b7	b					

Рисунок 3.55 – Корректный обмен. Клиент

Для отправки объекта информации с типом **C\_SE\_TC\_1** (команда уставки с меткой времени) следует:

1. Добавить выходной буфер **IECBufOut**.
2. Задать **tid = C\_SE\_TC\_1**, адрес ASDU **adr = 3**, адрес объекта информации **ioa = 4**.  
Установить тип данных **tp = AI**, так как **C\_SE\_TC\_1** имеет формат с плавающей запятой.

3. На входе **flg** установить **4 бит** – посылать команду при изменении значения на входе. При такой настройке для отправки команды не нужно устанавливать бит на входе **ctl**.

		C_SE_TC_1_3_4		10	
		IECBufOut			
тип объекта информации C_SE_TC_1	tid	etid	bfo	bo	коннектор для подключения к блоку протокола
адрес ASDU 3	adr	ul32			
группа опроса 0x00001	itg	ul32			
класс данных (только 101) CLASS2	dcl	edc			
условия передачи данных 0x08	trt	uch			
адрес объекта информации 4	ioa0	ul32	ul32	sts0	не используется
тип данных AI	tp0	enm			
флаги 0x10	flg0	ul32			
вход данных 0	in0	any			
квалификатор 0	qlf0	uch			
скачок данных 0	tsh0	any			
не используется 0	smt0	any			
нижний предел 0	llm0	any			
верхний предел 0	hlm0	any			
управление 0	ctl0	uch			

Рисунок 3.56 – Настройка IECBufIn. C\_SE\_TC\_1

4. Добавить коннектор для подключения выходного буфера **IECBufOut** у блока **IEC104uni**.  
5. Подключить **IECBufOut** к коннектору блока **IEC104uni**.

Настройка канала АРМ в MasterSCADA 4D:

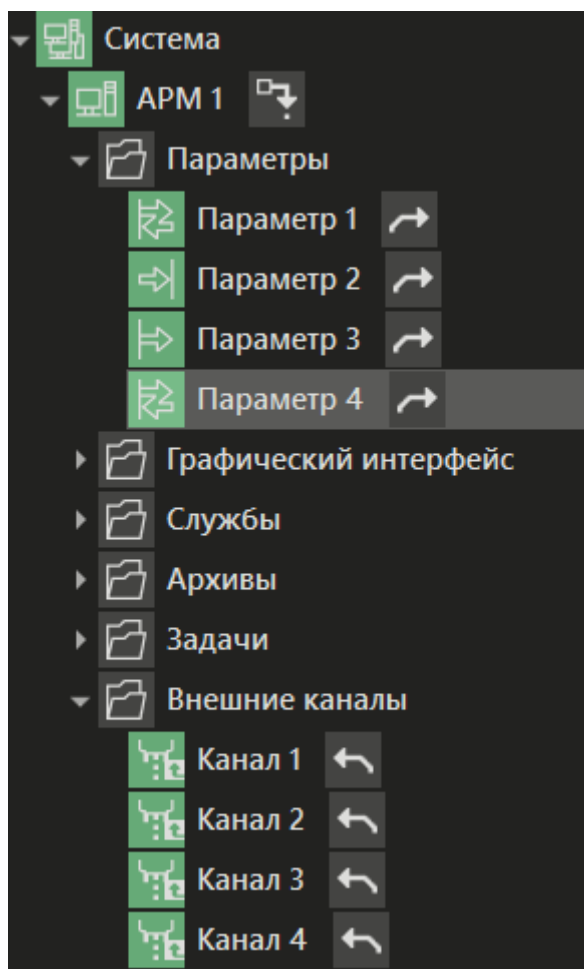


Рисунок 3.57 – Настройки канала АРМ

Канал 4

Поиск...

**Общие**

Доступ	Чтение/Запись
Имя	<b>Канал 4</b>
Полное имя	Система.АРМ 1.Внешние каналы.Канал 4
Метки	
Комментарий	
Описание	

**Отношения**

Ссылается	Система.АРМ 1.Параметры.Параметр 4
-----------	------------------------------------

**Настройки**

Адрес	-1
Адрес IEC104	<b>4</b>
Тип значения	<b>Вещественный</b>

Рисунок 3.58 – Настройки канала АРМ

6. Запустить программы на ПЛК и АРМ. Корректный обмен изображен на рисунках ниже.

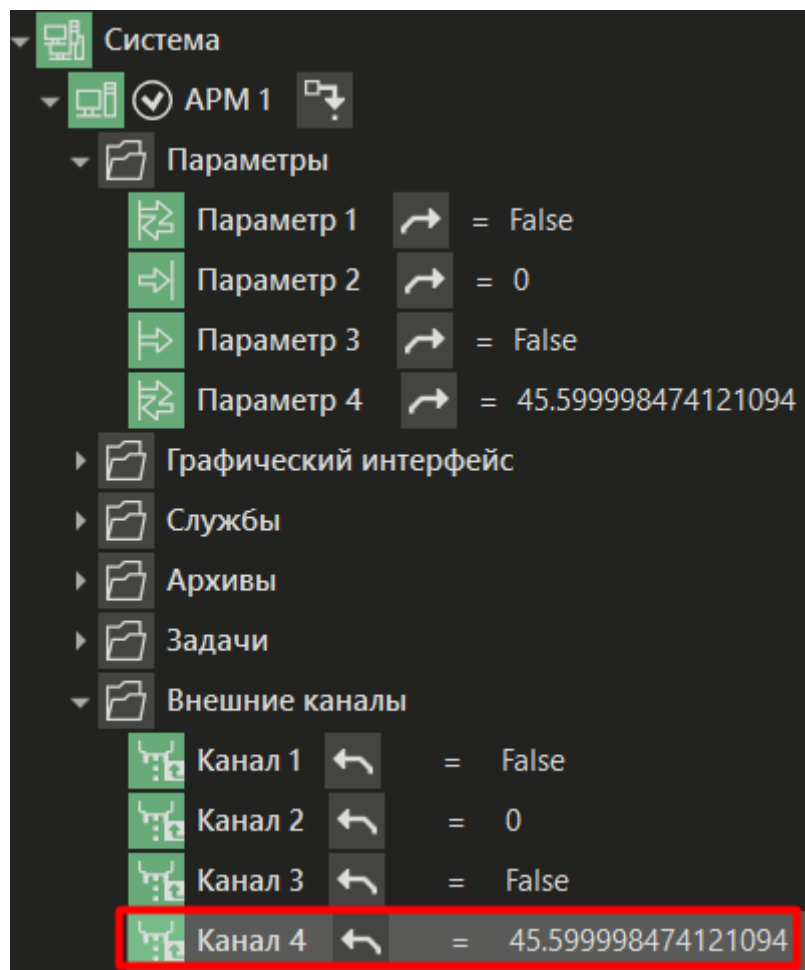


Рисунок 3.59 – Корректный обмен. Сервер

		C_SE_TC_1_3_4				10
		IECBufOut				1мкс
тип объекта информации C_SE_TC_1	63	tid	etid	bfo	bo	? – коннектор для подключения к блоку протокола –
адрес ASDU	3	adr	ul32			
группа опроса 0x00001	0	itg	ul32			
класс данных (только 101) CLASS2	2	dcl	edc			
условия передачи данных 0x08	8	trt	uch			
адрес объекта информации	4	ioa0	ul32	ul32	sts0	0 не используется
тип данных AI	2	tp0	enm			
флаги 0x10	16	flg0	ul32			
вход данных 0	45.6	in0	any			
квалификатор	0	qlf0	uch			
скачок данных	0	tsh0	any			
не используется	0	smt0	any			
нижний предел	0	llm0	any			
верхний предел	0	hlm0	any			
управление	0	ctl0	uch			

Рисунок 3.60 – Корректный обмен. Клиент

### 3.2.3 Прием упакованных данных (M\_EP\_\*)

В этом подразделе показан прием объекта информации с типом **M\_EP\_TE\_1** – упакованное сообщение с меткой времени **CP56Время2a**. Особенность данного типа состоит в том, что в сообщении передается структура, состоящая из битовой маски и интервала времени.

Для приема объекта информации с типом **M\_EP\_TE\_1** следует выполнить действия:

1. Добавить на любую страницу места работы **Фон** выходной буфер **IECBufIn**.
2. Задать **tid = M\_EP\_TE\_1**, адрес ASDU **adr = 3**, адрес объекта информации **ioa = 5**.

Установить тип данных **tp = uLX**, т.к. входные данные для данного типа ASDU структурированы.

- Для расшифровки структуры на выходе **o** использовать блок **IECEPFromInt**. Для удобства подключим к нему блок **ToReg8** из библиотеки **paCore**.
- Для расшифровки описателя качества с выхода **qIf** подсоединим его также к входу блока **FromReg8**. Здесь добавляется атрибут **EI** – действительность интервала длительности.
- Для расшифровки метки времени выход **qIf** и вход блока **FromReg8**. **IECTransTime**.

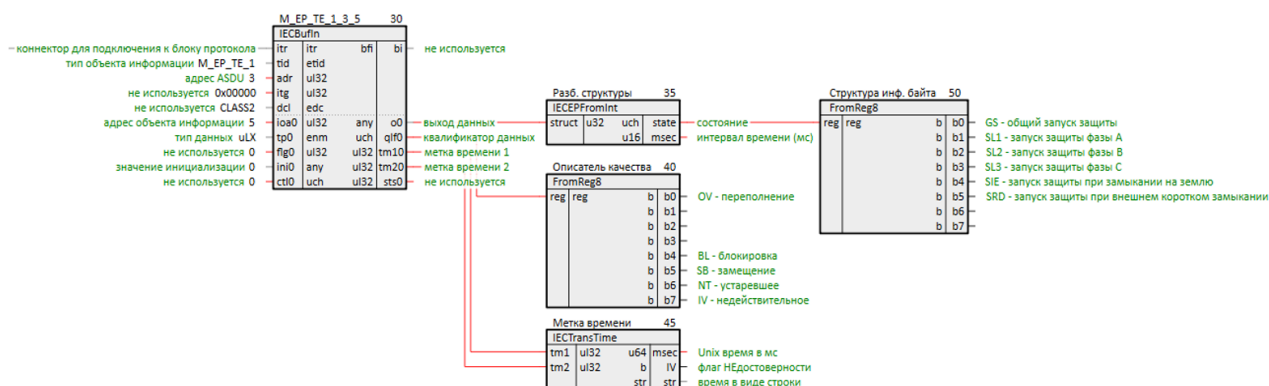


Рисунок 3.61 – Настройка IECBufIn. M\_EP\_TE\_1

- Подключить **IECBufIn** к коннектору блока **IEC104uni**.  
В качестве сервера настроим виртуальный контроллер Полигон. Можно также запустить в качестве клиента проект, рассмотренный в разделе 3.2, на виртуальном контроллере с помощью панели отладки.
- Для формирования структуры использовать блок **IECIntFromEP**.

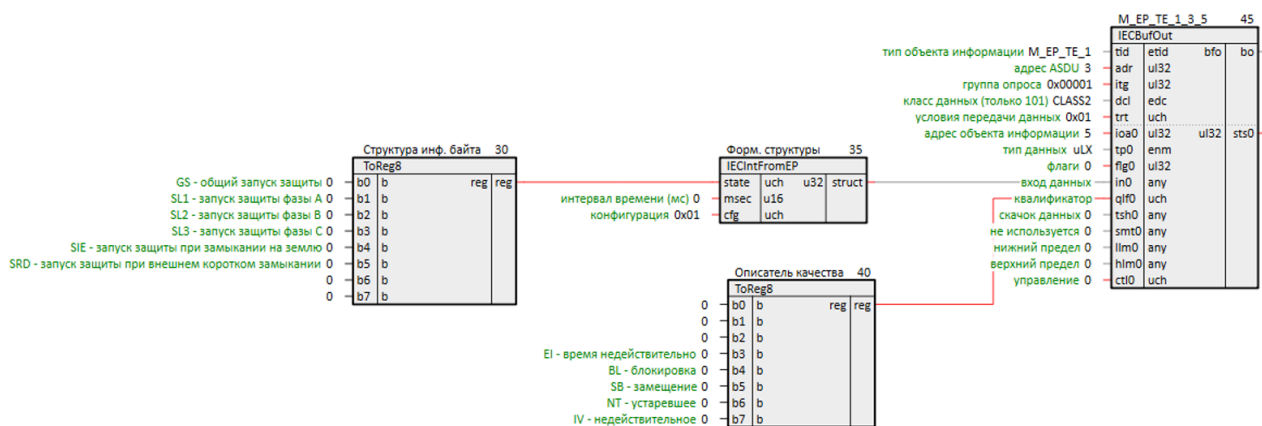


Рисунок 3.62 – Настройки сервера ПА

- Запустить программы на ПЛК и виртуальном контроллере. Корректный обмен показан на рисунках ниже.

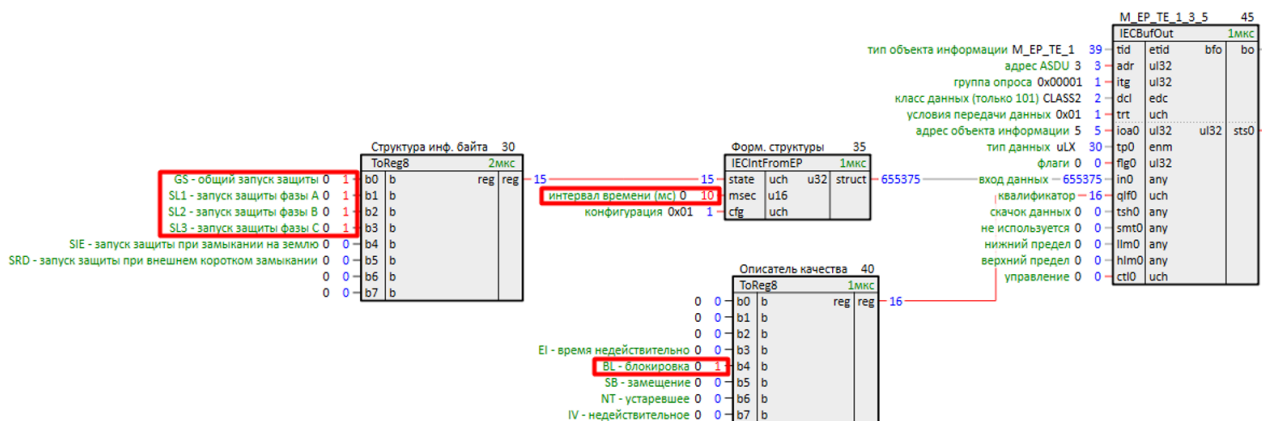
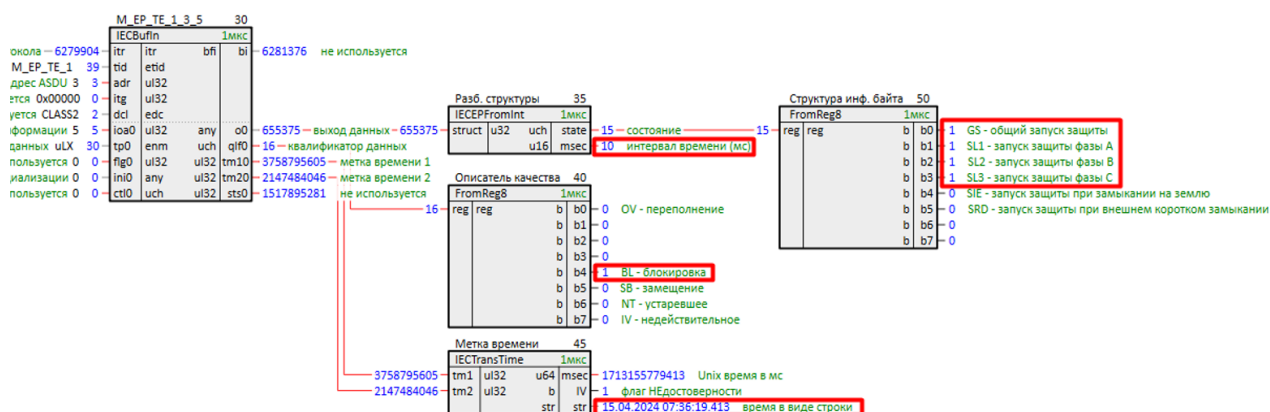


Рисунок 3.63 – Корректный обмен. Сервер





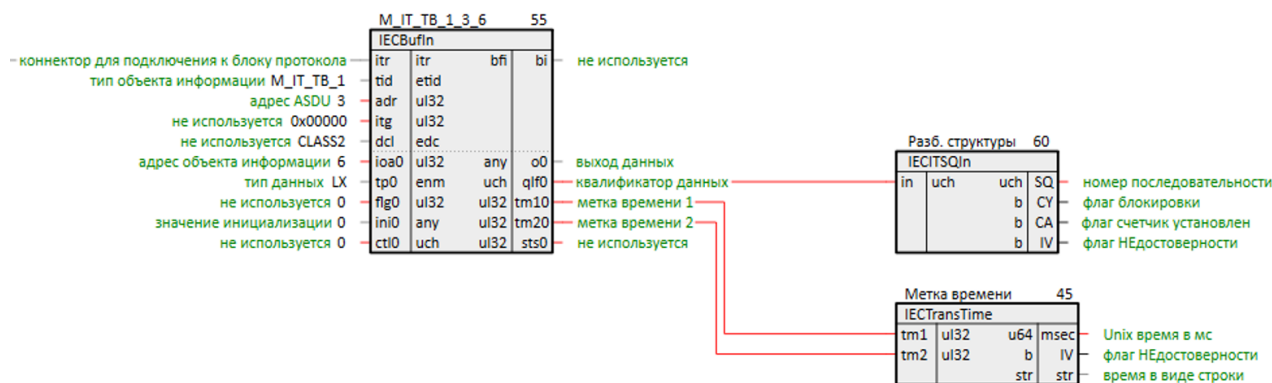
### 3.2.4 Прием интегральных сумм (M\_IT\_\*)

Здесь показан прием объекта информации с типом **M\_IT\_TB\_1** – интегральная сумма с меткой времени **CP56Время2а**. Особенность данного типа состоит в том, что в сообщении передается знаковое целое (в 4-х байтах).

Для приема объекта информации с типом **M\_IT\_TB\_1** следует:

1. Добавить на любую страницу места работы **Фон** выходной буфер **IECBufIn**.
2. Задать **tid = M\_IT\_TB\_1**, адрес ASDU **adr = 3**, адрес объекта информации **ioa = 6**.  
Установить тип данных **tp = LX**.
3. Для расшифровки структуры на выходе **qlf** использовать блок **IECITSQIn**.

Для расшифровки метки времени соединить выходы **tm1** и **tm2** с входами блока **IECTransTime**.



4. Подключить **IECBufOut** к коннектору блока **IEC104Server**.

В качестве сервера настроить виртуальный контроллер **Полигон**. Можно также запустить в качестве сервера проект, рассмотренный в [разделе 3.1](#), на виртуальном контроллере с помощью панели отладки.

5. Для формирования структуры использовать блок **IECITSQOut**.

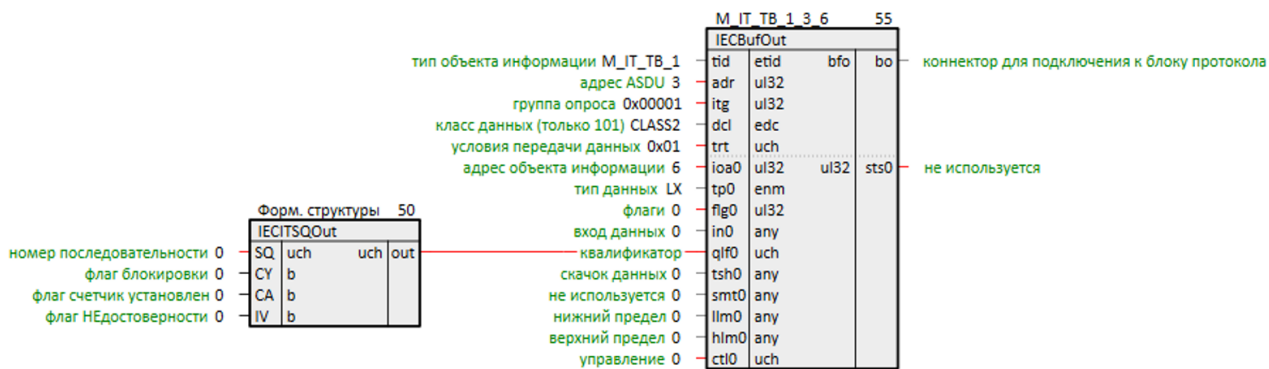


Рисунок 3.66 – Настройки сервера ПА

6. Запустим программы на ПЛК и виртуальном контроллере. Корректный обмен показан на рисунках ниже.

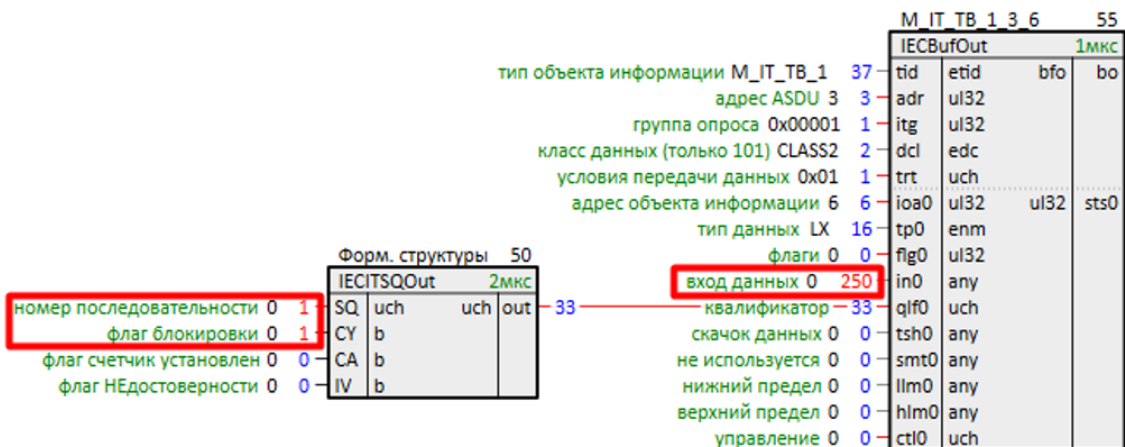


Рисунок 3.67 – Корректный обмен. Сервер

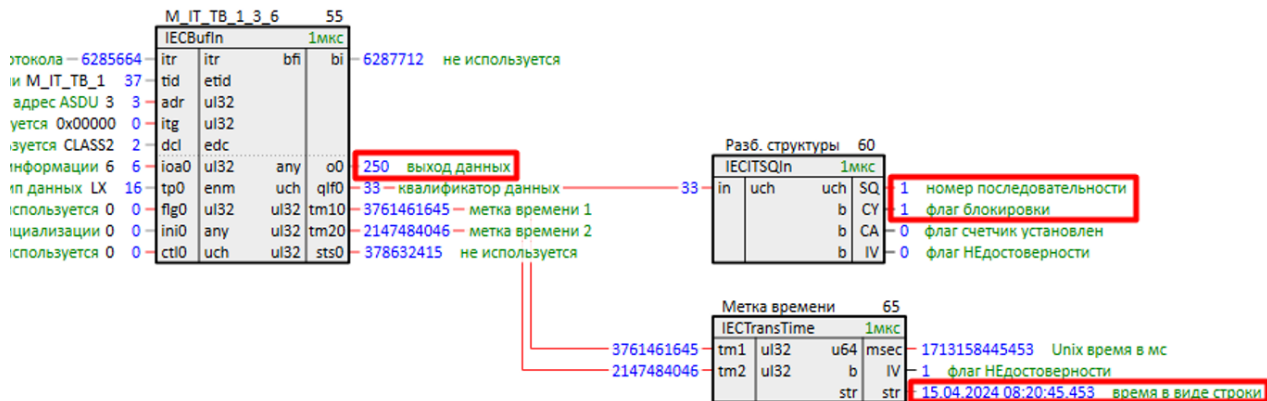


Рисунок 3.68 – Корректный обмен. Клиент

## 4 Примеры настройки обмена по протоколу МЭК 60870-5-101

### 4.1 Настройка обмена в режиме сервера протокола 5-101

Здесь и далее в примерах настраивается обмен по протоколу **5-101**.

В данном примере ПЛК210 будет выступать в роли контролируемой станции (сервера).

Пример доступен для скачивания по [ссылке](#). Пароль для доступа к отладчику – 1.

Для настройки обмена данными с ПЛК в режиме сервера следует:

1. Добавить на любую страницу места работы **Фон** блок **IEC101uni**.
2. Соединить вход блока **cnc** и аналогичный выход блока работы с COM-портом. Для контроллеров ОВЕН блоки работы с COM-портами **210-RS485** и **210-RS232** находятся в библиотеке **paOwenIO**.
3. На входе **prm** задать режим **IEC\_SLAVE**.
4. Добавить коннектор для выходного буфера **IECBufOut**.

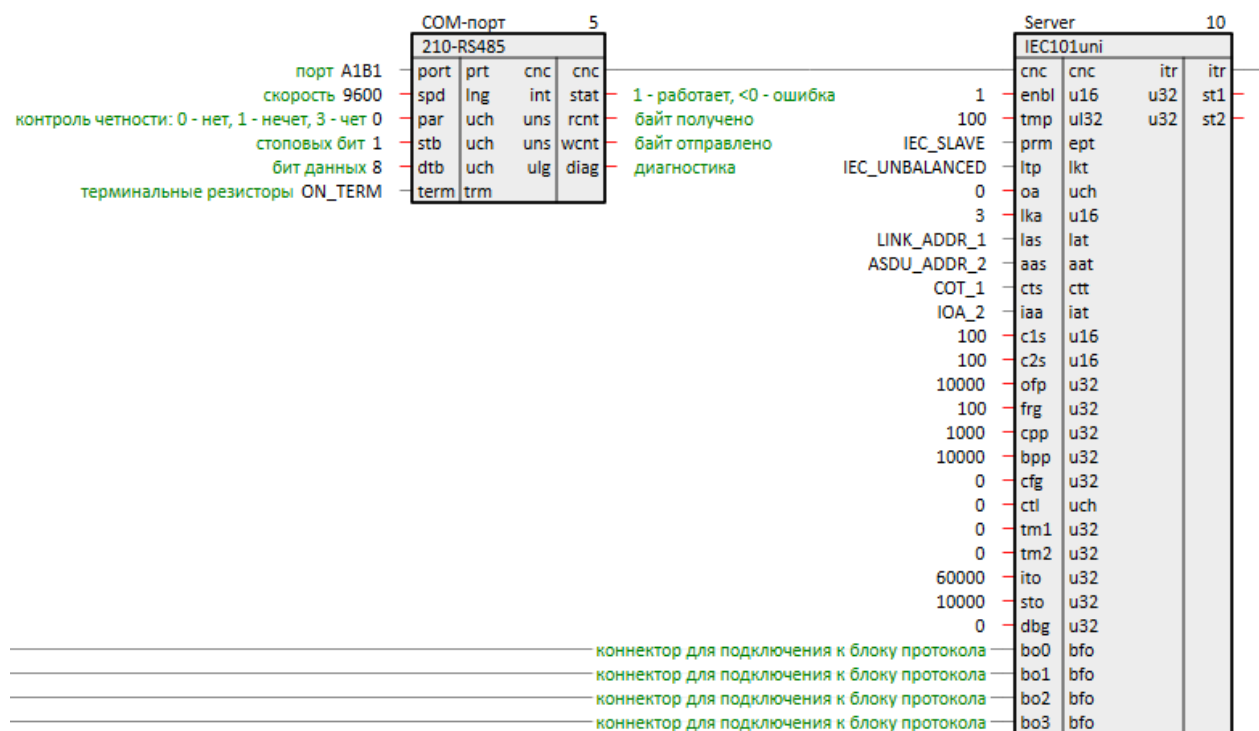


Рисунок 4.1 – Настройка IEC101uni в режиме IEC\_SLAVE



#### ПРИМЕЧАНИЕ

Прием и передача ASDU для протокола **5-101** аналогичны рассмотренным в [разделах 3.1 и 3.2](#). Поддерживаемые типы ASDU приведены в [Приложении А](#) и в справке среды программирования.

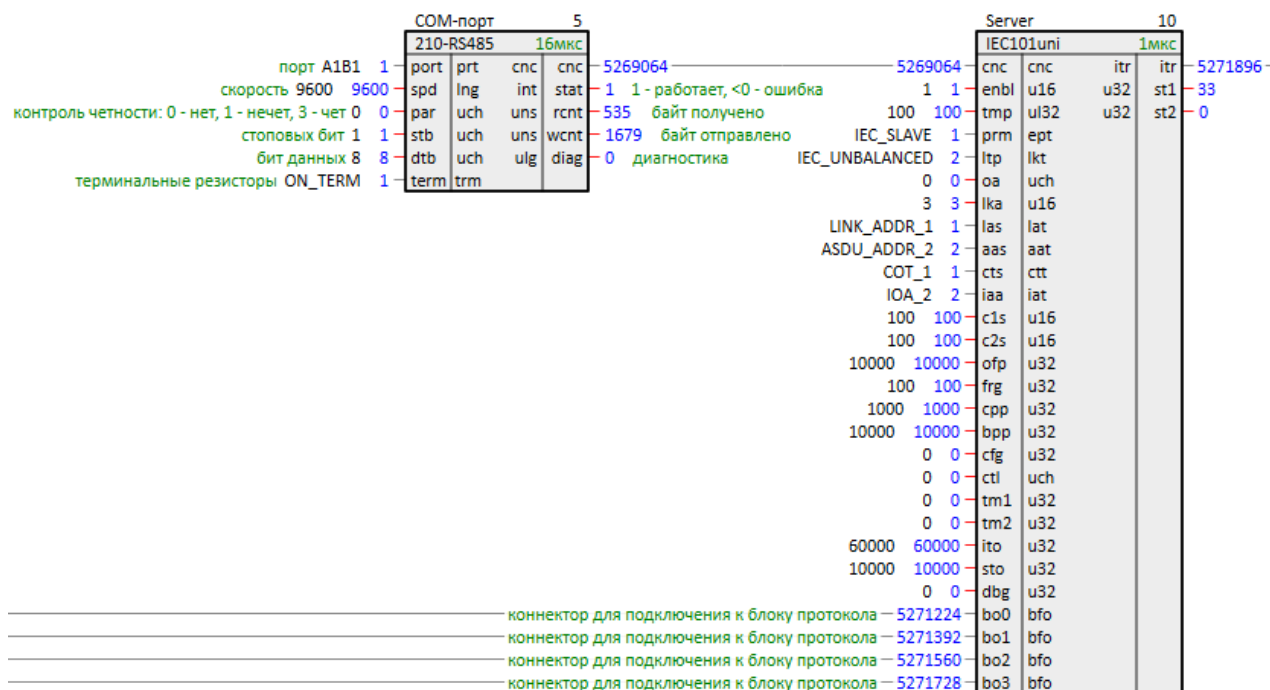


Рисунок 4.2 – Корректный обмен

## 4.2 Настройка обмена в режиме клиента протокола 5-101

В данном примере ПЛК210 будет выступать в роли контролирующей станции (клиента).

Пример доступен для скачивания по [ссылке](#). Пароль для доступа к отладчику – 1.

Для настройки обмена данными с ПЛК в режиме клиента следует:

1. Добавить на любую страницу места работы **Фон** блок **IEC101uni**.
2. Соединить вход блока **cnc** и аналогичный выход блока работы с COM-портом. Для контроллеров ОВЕН блоки работы с COM-портами **210-RS485** и **210-RS232** находятся в библиотеке **paOwenIO**.
3. На входе **prm** задать режим **IEC\_MASTER**.
4. Добавить коннектор для выходного буфера **IECBufOut**.

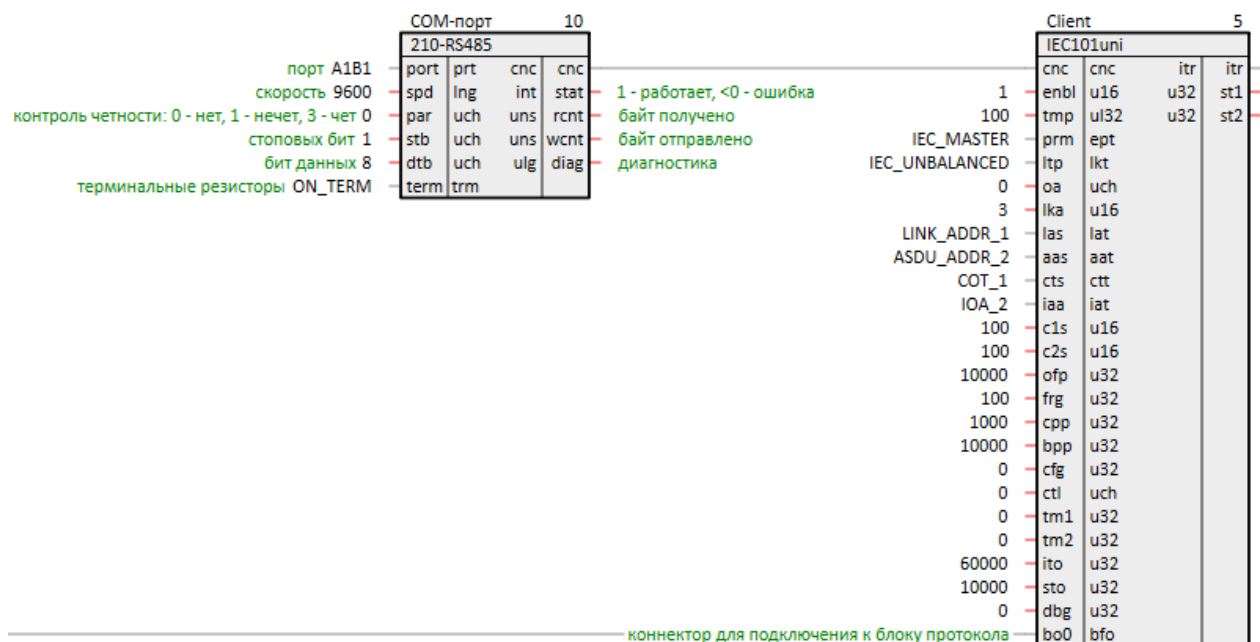


Рисунок 4.3 – Настройка IEC101uni в режиме IEC\_MASTER

**ПРИМЕЧАНИЕ**

Прием и передача ASDU для протокола **5-101** аналогичны рассмотренным в [разделах 3.1 и 3.2](#). Поддерживаемые типы ASDU приведены в [Приложении А](#) и в справке среды программирования.

		COM-порт				10	Client				5
		210-RS485		17мкс		5270744	IEC101uni		1мкс		5269264
порт A1B1	1	port	prt	cnc	cnc	1	cnc	cnc	itr	itr	16
скорость	9600	spd	lng	int	stat	1	enbl	u16	u32	st1	0
контроль четности:	0 - нет, 1 - нечет, 3 - чет	par	uch	uns	rcnt	100	tmp	u16	u32	st2	0
стоповых бит	1	stb	uch	uns	wcnt	2	prn	ept			
бит данных	8	dtb	uch	ulg	diag	2	IEC_MASTER	ltp	lkt		
терминальные резисторы	ON_TERM	term	trm			0	IEC_UNBALANCED	oa	uch		
						3		lka	u16		
						1	LINK_ADDR_1	las	fat		
						2	ASDU_ADDR_2	aas	aat		
						1	COT_1	cts	ctt		
						2	IOA_2	iaa	iat		
						100	100	c1s	u16		
						100	100	c2s	u16		
						10000	10000	ofp	u32		
						100	100	frg	u32		
						1000	1000	cpp	u32		
						10000	10000	bpp	u32		
						0	0	cfg	u32		
						0	0	ctl	uch		
						0	0	tm1	u32		
						0	0	tm2	u32		
						60000	60000	ito	u32		
						10000	10000	sto	u32		
						0	0	dbg	u32		
						5269096	bo0	bfo			

коннектор для подключения к блоку протокола — 5269096

Рисунок 4.4 – Корректный обмен

## 5 Отладка и диагностика обмена

Для отладки и диагностики обмена по протоколам МЭК 60870 в среде реализован вывод диагностических сообщений.

Конфигурация типов выводимых сообщений в лог протокола и место вывода определяется входом **dbg** блоков протоколов **IEC101uni**, **IEC104uni**, **IEC104Server**.

Если задать **0** на данном входе, в консоль выводятся только сообщения с **типом 0** (фатальная ошибка, статусная информация и т.п.).



### ПРИМЕЧАНИЕ

Сообщения с **типом 0** выводятся всегда, независимо от установленного значения на входе **dbg**.



### ПРИМЕЧАНИЕ

Так как формирование лога диагностических сообщений значительно нагружает процессор ПЛК, при неиспользовании диагностических сообщений рекомендуется устанавливать **dbg = 0**.

```

root@plc210rk_12_polygon: ~

Initialization complete.
Starting background loop...
08:59:45.175 -0- IEC104 Slave Registered: MaxClients(2). (CIEC104::iRegisterSlave)
08:59:45.175 -0- CIEC104(Port: 2404, OA: 0) Main Thread priority was set to 25 (CIEC104Server::start)
BlockThread starting new thread with priority=25 block=0x0x2c0be3d0
BlockThread thread id=-1144688192 started
OK set affinity to CBaseBlockThread thread=-1144688192 cpu=0
08:59:45.212 -0- CIEC104(Port: 2404, OA: 0) Start listening on 10.2.12.12:2404. (CTCPSocket::_listen_socket)
08:59:45.213 -0- CIEC104(Port: 2404, OA: 0) RxTx Thread priority was set to 25 (CTCPServer::start)
BlockThread starting new thread with priority=25 block=0x0x2c0be310
BlockThread thread id=-1153080896 started
08:59:45.250 -0- CIEC104(Port: 2404, OA: 0) Server started. (CTCPServer::bkgrWork)
OK set affinity to CBaseBlockThread thread=-1153080896 cpu=2
08:59:45.311 -0- CIEC104(Port: 2404, OA: 0) New connection from 10.2.10.10:55997, sockfd 5. (CTCPServer::bkgrWork)
08:59:45.311 -0- CIEC104(Port: 2404, OA: 0) Socket read value SO_SNDBUF buflen = 87040. (CTCPServer::bkgrWork)
08:59:45.311 -0- CIEC104(Port: 2404, OA: 0) Node(0): Connected. (IEC10xBase.cpp:ProtoCB)
08:59:45.435 -0- CIEC104(Port: 2404, OA: 0) Node(0): Activated. (IEC10xBase.cpp:ProtoCB)

```

Рисунок 5.1 – Вывод сообщений при **dbg = 0**

Вывод сообщений осуществляется двумя различными способами. Способ вывода задается соответствующим битом на входе **dbg** (см. **биты 8 и 9**):

- Вывод в консоль – при запуске проекта в консоли;
- Вывод через блок **RamLog** из библиотеки **paCore** (см. справку среды).

Первый способ удобен при тестировании и отладке программы, когда есть возможность запустить ее через консоль.



```

root@plc210rk_12_polygon: ~
10:00:49.985 -8- CIEC104(Port: 2404, OA: 0) Slot(0 10.2.10.10:57157): Sent bytes
 16 of 16: 68 0E 08 00 02 00 01 01 01 00 03 00 01 00 00 00; (CTCPServer::_transm
it_task)
10:00:50.637 -2- CIEC104(Port: 2404, OA: 0) Slave Slot(0): T3 Expired, Sending T
ESTACT. (CIEC104::T3timerCB)
10:00:50.652 -8- CIEC104(Port: 2404, OA: 0) Trying to send: Slot(0) Data(6): 68
04 43 00 00 00; (CIEC104Server::cbCheckForData)
10:00:50.653 -8- CIEC104(Port: 2404, OA: 0) Slot(0 10.2.10.10:57157): Sent bytes
 6 of 6: 68 04 43 00 00 00; (CTCPServer::_transmit_task)
10:00:50.774 -7- CIEC104(Port: 2404, OA: 0) Slot(0 10.2.10.10:57157): Received D
ata(6): 68 04 83 00 00 00; (CTCPServer::_receive_task)
10:00:50.774 -7- CIEC104(Port: 2404, OA: 0) Data Received: Slot(0) Data(6): 68 0
4 83 00 00 00; (CIEC104Server::cbDataReadyInd)
10:00:50.774 -7- CIEC104(Port: 2404, OA: 0) Received Slave Slot(0) Data[6] = 68
04 83 00 00 00; (CIEC104::DataReadyCB)
10:00:55.038 -3- CIEC104(Port: 2404, OA: 0) Queueing Message Node[0] (E1 1 Of 10
0) TypeID(M_SP_NA_1) Data(0) COT(COT_Per_Cyc) IOA(1) PN(0) T(0) to ASDU(3). (CIE
C104::iQueueMessage)
10:00:55.043 -8- CIEC104(Port: 2404, OA: 0) Trying to send: Slot(0) Data(16): 68
0E 0A 00 02 00 01 01 01 00 03 00 01 00 00 00; (CIEC104Server::cbCheckForData)
10:00:55.043 -8- CIEC104(Port: 2404, OA: 0) Slot(0 10.2.10.10:57157): Sent bytes
 16 of 16: 68 0E 0A 00 02 00 01 01 01 00 03 00 01 00 00 00; (CTCPServer::_transm
it_task)

```

Рисунок 5.2 – Вывод сообщений в консоль при dbg = 0x1C7

Второй способ позволяет записывать лог из оперативной памяти ПЛК в файл на диске контроллера.

Пример настройки записи лога через **RamLog** в файл на USB-накопителе:

Запись в ОЗУ		10		
RamLog		5мкс		
1	1- enb	b	u32 sts	0 Статус, 0 - ОК, >0 -ошибки
Размер буфера	1024 1024- bsize	u32	rlog	log
Количество буферов	8 8- bnum	u32	s100	cds
Создать все буферы при инициализации	0 0- init	b	s100	indxs
Таймаут неполного буфера (мс)	100 100- tmout	u64		
Уровень ошибок	16 16- glvl	u16		
Уровень отладки	10 10- dbglvl	u16		
Сбросить коды и индексы	0 0- dscred	b		
Код отладки (-d)	0 0- code	u32		
Добавить код	0 0- addcd	b		
Удалить код	0 0- delcd	b		
Индекс блока (b)	0 0- indx	u32		
Добавить индекс	0 0- addix	b		
Удалить индекс	0 0- delix	b		
Печатать в консоль	0 0- print	b		

Запись на USB		15			
RamLogToFile		1мкс			
?	log	rlog	i32 sts	1 Статус: 1-пишет,0-ждёт,<0-ошибка	
Формат имени файла	"/sda1/RamLog-%y-%m-%d-%H-%_"	fname	s50 s100	cfname	"/sda1/RamLog-24-04-15-10-0 Текущее имя файла
	Приписывать 0	app	b u64	fsize	5993 Размер текущего файла
Частота смены файла (минут)	60 60- rtime	u64	u32	fnum	0 Количество превышений размера
Максимальный размер файла (МБ)	10 10- rsize	flt			

Рисунок 5.3 – Настройка вывода лога через RamLog



```

/mnt/ufs/media/sda1/RamLog-24-04-15-10-0 - root@10.2.12.12 - Редактор - WinSCP
Кодировка Цвет
10:39:22.723 -0- IEC104 Slave Registered: MaxClients(2). (CIEC104::iRegisterSlave)
10:39:22.723 -0- CIEC104(Port: 2404, OA: 0) Main Thread priority was set to 25 (CIEC104Server::start)
10:39:22.760 -1- CIEC104(Port: 2404, OA: 0) Address 10.2.12.12:2404 binded to socket(5). (CTCPSocket::_listen_socket)
10:39:22.761 -0- CIEC104(Port: 2404, OA: 0) Start listening on 10.2.12.12:2404. (CTCPSocket::_listen_socket)
10:39:22.761 -0- CIEC104(Port: 2404, OA: 0) RxTx Thread priority was set to 25 (CTCPServer::start)
10:39:22.798 -0- CIEC104(Port: 2404, OA: 0) Server started. (CTCPServer::bkgrWork)
10:39:23.071 -0- CIEC104(Port: 2404, OA: 0) New connection from 10.2.10.10:58052, sockfd 6. (CTCPServer::bkgrWork)
10:39:23.071 -0- CIEC104(Port: 2404, OA: 0) Socket read value SO_SNDBUF buflen == 87040. (CTCPServer::bkgrWork)
10:39:23.071 -1- CIEC104(Port: 2404, OA: 0) Trying To Connect: Slot 0, Socket 6, Address 10.2.10.10:58052... (CIEC104Server::NewConnectionCB)
10:39:23.071 -1- CIEC104(Port: 2404, OA: 0) Slave Slot(0): New Client Added. (CIEC104::NewConnectionCB)
10:39:23.071 -0- CIEC104(Port: 2404, OA: 0) Node(0): Connected. (IEC10xBase.cpp::ProtoCB)
10:39:23.071 -1- CIEC104(Port: 2404, OA: 0) Connection for 10.2.10.10:58052 added to slot 0. (CTCPServer::_add_conn)
10:39:23.202 -7- CIEC104(Port: 2404, OA: 0) Slot(0 10.2.10.10:58052): Received Data(6): 68 04 07 00 00 00; (CTCPServer::DataReadyCB)
10:39:23.203 -7- CIEC104(Port: 2404, OA: 0) Data Received: Slot(0) Data(6): 68 04 07 00 00 00; (CIEC104Server::cbDataReadyCB)
10:39:23.203 -7- CIEC104(Port: 2404, OA: 0) Received Slave Slot(0) Data[6] = 68 04 07 00 00 00; (CIEC104::DataReadyCB)
10:39:23.203 -0- CIEC104(Port: 2404, OA: 0) Node(0): Activated. (IEC10xBase.cpp::ProtoCB)
10:39:23.203 -3- CIEC104(Port: 2404, OA: 0) Queueing Reply Node[0] (El 1 Of 100) TypeID(M_EI_NA_1) Data(Local Power Sw
10:39:23.445 -7- CIEC104(Port: 2404, OA: 0) Slot(0 10.2.10.10:58052): Received Data(16): 68 0E 00 00 02 00 46 01 04 00 00 00 00 00 00 00
10:39:23.445 -7- CIEC104(Port: 2404, OA: 0) Data Received: Slot(0) Data(16): 68 0E 00 00 02 00 46 01 04 00 03 00 00 00 00 00
10:39:23.445 -7- CIEC104(Port: 2404, OA: 0) Received Slave Slot(0) Data[16] = 68 0E 00 00 02 00 46 01 04 00 03 00 00 00 00 00
10:39:23.445 -2- CIEC104(Port: 2404, OA: 0) Node(0): ASDU Received: TypeID(M_EI_NA_1), COT(COT_Init), AddrASDU(3), OA(0)
10:39:27.829 -3- CIEC104(Port: 2404, OA: 0) Queueing Message Node[0] (El 1 Of 100) TypeID(M_SP_NA_1) Data(0) COT(COT_F
10:39:32.896 -3- CIEC104(Port: 2404, OA: 0) Queueing Message Node[0] (El 1 Of 100) TypeID(M_SP_NA_1) Data(0) COT(COT_F
10:39:37.943 -3- CIEC104(Port: 2404, OA: 0) Queueing Message Node[0] (El 1 Of 100) TypeID(M_SP_NA_1) Data(0) COT(COT_F
10:39:42.990 -3- CIEC104(Port: 2404, OA: 0) Queueing Message Node[0] (El 1 Of 100) TypeID(M_SP_NA_1) Data(0) COT(COT_F
10:39:43.453 -2- CIEC104(Port: 2404, OA: 0) Slave Slot(0): T3 Expired, Sending TESTACT. (CIEC104::T3timerCB)

```

Рисунок 5.4 – Вывод сообщений в файл при dbg = 0x2C7

Для отслеживания времен выполнения подпрограмм, размера очереди сообщений, переполнения очереди и др. можно использовать блок *IECInfo* (описание выходов см. в разделе 2.11).



**ПРИМЕЧАНИЕ**

Блок *IECInfo* совместим только с блоком протокола *IEC104Server*.

"<sql>SELECT value FROM blocks\_prop WHERE indx=:module AND type="prop\_ip"</sql>"

Server		5		46мкс	
режим работы	1	enbl	u16	itr	itr
период вызова обработчика событий	100	100	u32	st0	1
локальный ip "[SQL]"	10.2.12.12	lip	str	st1	0
порт	2404	lprt	u16		
путь к сетевому стеку (для QNX) "/"		sdr	str		
приоритет процесса обработчика событий	0	evntsprio	u16		
приоритет процесса приема/передачи	0	rxtprio	u16		
ограничение клиентских подключений	2	clts	uch		
размер буфера приема	2048	rxbsz	u32		
размер буфера передачи	2048	txbsz	u32		
не используется	0	oa	uch		
параметр W	8	w	u16		
параметр K	12	k	u16		
таймаут 1	15000	t1	u32		
таймаут 2	10000	t2	u32		
таймаут 3	20000	t3	u32		
глубина очереди сообщений	100	qsz	u16		
период отправки циклических данных	5000	cpp	u32		
период отправки данных фонового сканирования	10000	bpp	u32		
конфигурация 0x20	32	cfg	u32		
управление	0	ctl	uch		
конфигурация отладки 0x4C7	1223	dbg	u32		
не используется	0	tm1	u32		
не используется	0	tm2	u32		
таймаут выполнения команды	60000	ito	u32		
таймаут выбора команды	10000	sto	u32		
коннектор для подключения к блоку протокола	?	bo0	bfo		
коннектор для подключения к блоку протокола	?	bo1	bfo		
коннектор для подключения к блоку протокола	?	bo2	bfo		
коннектор для подключения к блоку протокола	?	bo3	bfo		

b98		20		1мкс	
itr	itr	fit	maintmmin	0.040834	
ctl	u32	fit	maintmmax	0.044625	
tmout	1000	fit	maintmavrg	0.0430675	
rst	0	fit	maintmnum	50	
		fit	maintmabsmax	39.9187	
		fit	evntstmmmin	0.010208	
		fit	evntstmmmax	0.023916	
		fit	evntstmavrg	0.0130884	
		u32	evntstmmnum	8	
		fit	evntstmabsmax	0.098875	
		fit	auxtmmin	0.070292	
		fit	auxtmmax	0.144666	
		fit	auxtmavrg	0.077875	
		u32	auxtmnum	50	
		fit	auxtmabsmax	2.97354	
		fit	txrxtmmin	0.014291	
		fit	txrxtmmax	0.077	
		fit	txrxtmavrg	0.0173249	
		u32	txrxtmnum	50	
		fit	txrxtmabsmax	2.51679	
		u32	msgQmax0	0	
		u32	msgQavrg0	0	
		u32	msgQnum0	50	
		u32	msgQabsmax0	0	
		u32	msgQfull0	0	
		u32	rpIQfull0	0	
		u32	reachedK0	0	
		u32	txbufmax0	0	
		u32	txbufavrg0	0	
		u32	txbufnum0	50	
		u32	txbufabsmax0	38	
		u32	txbufReachedLim0	0	

Рисунок 5.5 – Работа блока IECInfo

## Приложение А. Описание типов информации ASDU

Таблица А.1 – Описание типов информации ASDU

Код	Вид информации	Метка	Реализация в раIEC104
<b>Информация о процессе в направлении контроля (M)</b>			
1	Одноэлементная	M_SP_NA_1	Да
2	Одноэлементная с временной меткой	M_SP_TA_1	Да (только 5-101)
3	Двухэлементная	M_DP_NA_1	Да
4	Двухэлементная с временной меткой	M_DP_TA_1	Да (только 5-101)
5	Информация о положении отпаяк	M_ST_NA_1	Нет
6	Информация о положении отпаяк с временной меткой	M_ST_TA_1	Нет
7	Строка из 32 битов	M_BO_NA_1	Да
8	Строка из 32 битов с временной меткой	M_BO_TA_1	Да (только 5-101)
9	Измерение	M_ME_NA_1	Нет
10	Измерение с временной меткой	M_ME_TA_1	Нет
11	Измерение, масштабируемое	M_ME_NB_1	Да
12	Измерение, масштабируемое, с временной меткой	M_ME_TB_1	Да (только 5-101)
13	Измерение, короткий формат с плавающей запятой	M_ME_NC_1	Да
14	Измерение, короткий формат с плавающей запятой, с временной меткой	M_ME_TC_1	Да (только 5-101)
15	Интегральная сумма (нарастающий итог)	M_IT_NA_1	Да
16	Нарастающий итог с временной меткой	M_IT_TA_1	Да (только 5-101)
17	Информация о релейной защите с временной меткой	M_EP_TA_1	Да (только 5-101)
18	Упакованная информация о срабатывании ПОЗ с временной меткой	M_EP_TB_1	Да (только 5-101)
19	Упакованная информация о срабатывании ВЦЗ с временной меткой	M_EP_TC_1	Да (только 5-101)
20	Упакованная одноэлементная информация с определением изменения состояния	M_PS_NA_1	Нет
21	Значение измеряемой величины, нормализованное, без описателя качества	M_ME_ND_1	Нет
22...29	Резерв для дальнейших совместимых определений	-	-
30	Одноэлементная информация с меткой времени <b>CP56Время2а</b> <sup>1)</sup>	M_SP_TB_1	Да
31	Двухэлементная информация с меткой времени <b>CP56Время2а</b>	M_DP_TB_1	Да
32	Информация о положении отпаяк с меткой времени <b>CP56Время2а</b>	M_ST_TB_1	Нет
33	Строка из 32 бит с меткой времени <b>CP56Время2а</b>	M_BO_TB_1	Да
34	Значение измеряемой величины, нормализованное, с меткой времени <b>CP56Время2а</b>	M_ME_TD_1	Нет
35	Значение измеряемой величины, масштабированное, с меткой времени <b>CP56Время2а</b>	M_ME_TE_1	Да

## Продолжение таблицы А.1

Код	Вид информации	Метка	Реализация в раЕС104
36	Значение измеряемой величины, короткий формат с плавающей запятой с меткой времени <b>CP56Время2а</b>	M_ME_TF_1	Да
37	Интегральная сумма с меткой времени <b>CP56Время2а</b>	M_IT_TB_1	Да
38	Информация о работе релейной защиты с меткой времени <b>CP56Время2а</b>	M_EP_TD_1	Да
39	Упакованная информация о срабатывании пусковых органов защиты с меткой времени <b>CP56Время2а</b>	M_EP_TE_1	Да
40	Упакованная информация о срабатывании выходных цепей защиты с меткой времени <b>CP56Время2а</b>	M_EP_TF_1	Да
41...44	Резерв для дальнейших совместимых определений	-	-
<b>Информация о процессе в направлении управления (С)</b>			
45	<b>CON</b> <sup>2)</sup> Команда однопозиционная	C_SC_NA_1	Да
46	<b>CON</b> Команда двухпозиционная	C_DC_NA_1	Да
47	<b>CON</b> Команда пошагового регулирования	C_RC_NA_1	Нет
48	<b>CON</b> Команда уставки, нормализованное значение	C_SE_NA_1	Нет
49	<b>CON</b> Команда уставки, масштабируемое значение	C_SE_NB_1	Да
50	<b>CON</b> Команда уставки, короткое число с плавающей запятой	C_SE_NC_1	Да
51	<b>CON</b> Строка из 32 битов	C_BO_NA_1	Да
52...57	Резерв для дальнейших совместимых определений	-	-
58	Однопозиционная команда с меткой времени <b>CP56Время2а</b>	C_SC_TA_1	Да (только 5-104)
59	Двухпозиционная команда с меткой времени <b>CP56Время2а</b>	C_DC_TA_1	Да (только 5-104)
60	Команда пошагового регулирования с меткой времени <b>CP56Время2а</b>	C_RC_TA_1	Нет
61	Команда уставки, нормализованное значение с меткой времени <b>CP56Время2а</b>	C_SE_TA_1	Нет
62	Команда уставки, масштабированное значение с меткой времени <b>CP56Время2а</b>	C_SE_TB_1	Да (только 5-104)
63	Команда уставки, короткий формат с плавающей запятой с меткой времени <b>CP56Время2а</b>	C_SE_TC_1	Да (только 5-104)
64	Строка из 32 бит с меткой времени <b>CP56Время2а</b>	C_BO_TA_1	Да (только 5-104)
65...69	Резерв для дальнейших совместимых определений	-	-
<b>Системная информация в направлении контроля (М)</b>			
70	Конец инициализации	M_EI_NA_1	Да
71...99	Резерв для дальнейших совместимых определений	-	-
<b>Системная информация в направлении управления (С)</b>			
100	<b>CON</b> Команда опроса	C_IC_NA_1	Да
101	<b>CON</b> Команда опроса счетчика	C_CI_NA_1	Да
102	<b>CON</b> Команда считывания	C_RD_NA_1	Да
103	<b>CON</b> Команда синхронизации часов	C_CS_NA_1	Да
104	<b>CON</b> Команда тестирования	C_TS_NA_1	Нет
105	<b>CON</b> Команда возврата процесса в исходное состояние	C_RP_NA_1	Нет
106	<b>CON</b> Команда передачи задержки	C_CD_NA_1	Нет

## Продолжение таблицы А.1

Код	Вид информации	Метка	Реализация в раЕC104
107	Команда тестирования с меткой времени <b>CP56Время2a</b>	C_TS_TA_1	Нет
108... 109	Резерв для дальнейших совместимых определений	-	-
<b>Параметры в направлении управления (P)</b>			
110	<b>CON</b> Параметр измеряемой величины, нормализованное значение	P_ME_NA_1	Нет
111	<b>CON</b> Параметр измеряемой величины, масштабируемое значение	P_ME_NB_1	Нет
112	<b>CON</b> Параметр измеряемой величины, короткое число с плавающей запятой	P_ME_NC_1	Нет
113	<b>CON</b> Параметр активации	P_AC_NC_1	Нет
114... 119	Резерв для дальнейших совместимых определений	-	-
<b>Передача файлов (F)</b>			
120	Файл готов	F_FR_NA_1	Нет
121	Секция готова	F_SR_NA_1	Нет
122	Вызов директории, выбор файла, вызов файла, секции	F_SC_NA_1	Нет
123	Последняя секция, последний сегмент	F_LS_NA_1	Нет
124	Подтверждение файла, подтверждение секции	F_AF_NA_1	Нет
125	Сегмент	F_SG_NA_1	Нет
126	Директория	F_DR_TA_1	Нет
127	Резерв для дальнейших совместимых определений	-	-

**ПРИМЕЧАНИЕ**

<sup>1)</sup> ASDU с метками времени **CP56Время2a** используются, если пункт управления не может добавить время от часов до лет однозначно к получаемым ASDU с метками от миллисекунд до минут. Это может случиться при использовании сетей с неопределенными задержками или когда возникает временный сбой в сети

<sup>2)</sup> ASDU с меткой **CON**, передаваемые в направлении управления, подтверждаются прикладным уровнем и могут возвращаться в направлении контроля при различных причинах передачи. Эти отраженные ASDU используются для положительного/отрицательного квитирования (проверки)

Более подробно типы описаны в ГОСТ Р МЭК 60870-5 «Устройства и системы телемеханики. Часть 5. Протоколы передачи» ([раздел 101](#), [раздел 104](#)).

## Приложение Б. Причины передачи (COT)

Таблица Б.1 – Причины передачи (COT)

Код	Описание
0	Не используется
1	Периодическая передача
2	Фоновое сканирование ( <b>Background Scan</b> )
3	Спорадическая передача при возникновении события ( <b>Spontaneous</b> )
4	Сообщение об инициализации ( <b>Initialized</b> )
5	Запрос или запрашиваемые данные ( <b>Request Or Requested</b> )
6	Активация ( <b>Activation</b> )
7	Подтверждение активации ( <b>Activation Confirm</b> )
8	Деактивация ( <b>Deactivation</b> )
9	Подтверждение деактивации ( <b>Deactivation Confirm</b> )
10	Завершение активации ( <b>Activation Termination</b> )
11	Обратная информация, вызванная удаленной командой ( <b>Return Information Caused By A Remote Command</b> )
12	Обратная информация, вызванная местной командой ( <b>Return Information Caused By A Local Command</b> )
13	Передача файлов ( <b>File Transfer</b> )
14...19	Резерв
20	Общий опрос ( <b>Interrogated By Station Interrogation</b> )
21...36	Опрос группы 1...16 ( <b>Interrogated By Group 1...16 Interrogation</b> )
37	Общий опрос счетчиков
38...41	Запрос счетчиков группы 1...4
42...43	Резерв
44	Неизвестный тип идентификатора
45	Неизвестная причина передачи
46	Неизвестный общий адрес станции
47	Неизвестный адрес объекта информации



Россия, 111024, Москва, 2-я ул. Энтузиастов, д. 5, корп. 5  
тел.: +7 (495) 641-11-56, факс: (495) 728-41-45  
тех. поддержка 24/7: 8-800-775-63-83, support@owen.ru  
отдел продаж: sales@owen.ru  
Веб-сайт ООО "ПромАвтоматика-Софт": www.pa.ru  
рег.:1-RU-135062-1.2