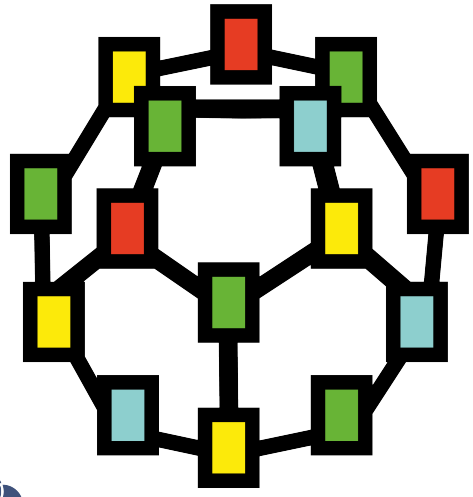
Библиотека palEC104



ПромАвтоматика

Руководство пользователя

05.2024
версия 1.1

Содержание

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, передающая информацию только по запросу контролирующей станции.

Контролирующая станция (Controlling Station, Master, Client) – станция, инициирующая процедуру обмена информацией.

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 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 (**циклическая передача** в Полигон) – 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 (**спонтанная передача** в Полигон) – 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.

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. Подразумевается, 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 paCore 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 ***palEC104***. Данная библиотека доступна для работы при наличии соответствующей лицензии runtime (см. описание лицензионных пакетов на странице среды программирования Полигон).

Документ соответствует версии среды Полигон 2 – **1917**, версии библиотеки ***palEC104*** – **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).

МЭК 60870-5 составляют 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.

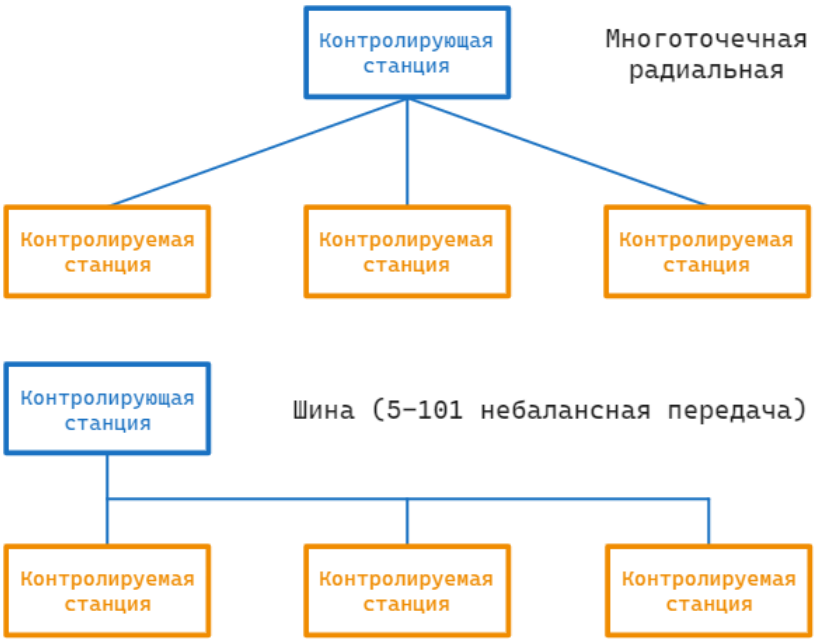


Рисунок 1.1 – Поддерживаемые топологии сети

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

Продолжение таблицы 1.3

Название	Размер	Описание
Общий адрес станции (**Common Adress** of ASDUs)	1…2 байта	Уникальный адрес станции в сети
Объекты информации (один или несколько)		
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**.

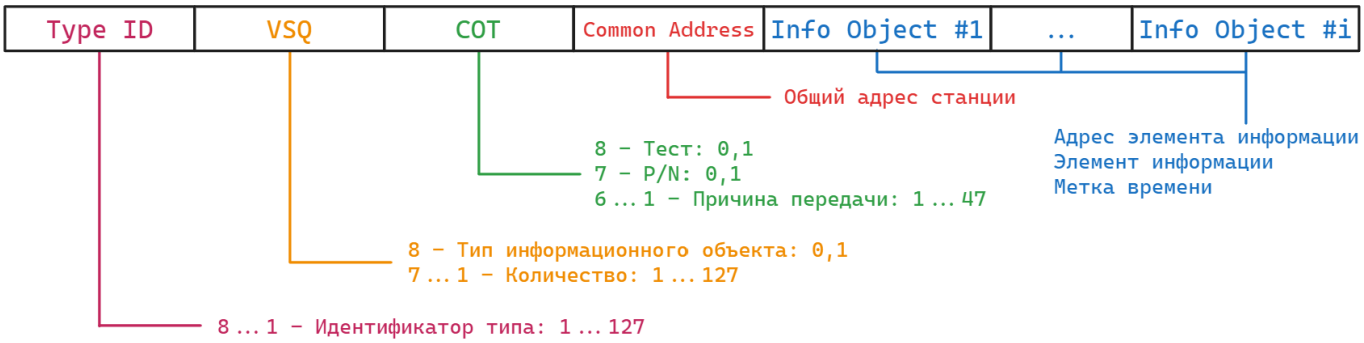


Рисунок 1.2 – Структура ASDU

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** и **5-104**, соответственно, в режимах клиент и сервер. Для работы с этими блоками требуется подключения блоков работы с системными интерфейсами – ***210-RS485***/***210-RS485*** из библиотеки ***paOwenIO*** и ***TcpIpCIA*** 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.

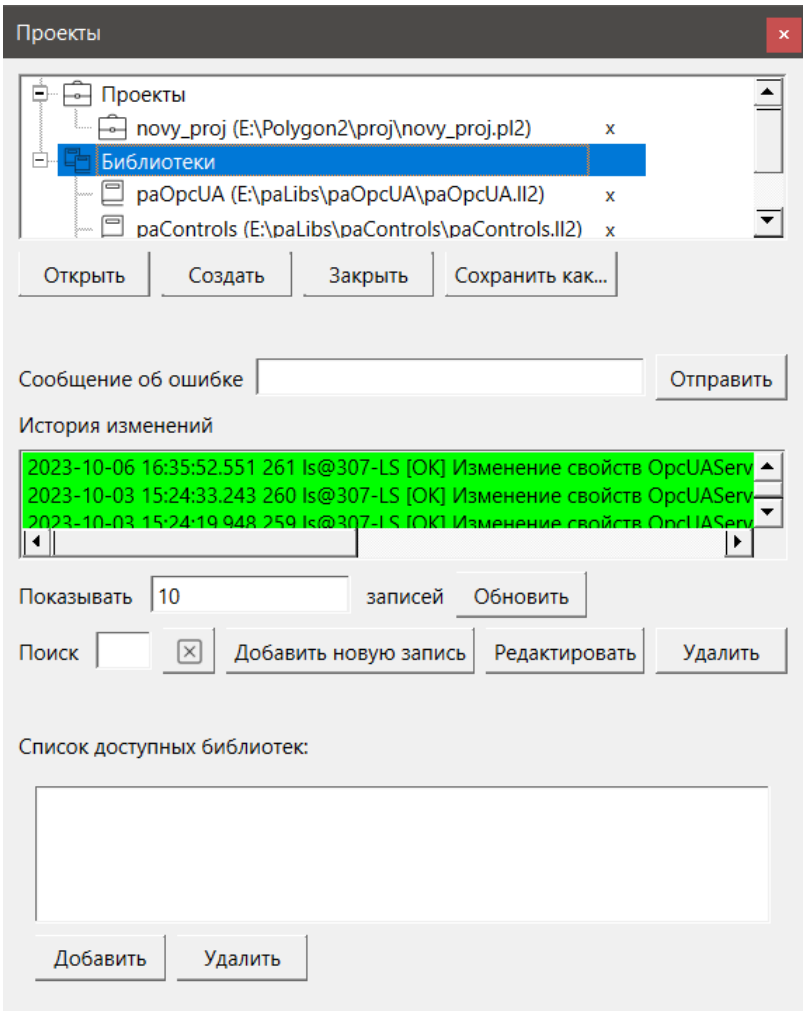


Рисунок 2.1 – Добавление библиотеки paIEC104 в проект

2. Нажать кнопку ***Открыть*** и перейти в папку с файлами библиотеки, которую необходимо добавить. Затем в выпадающем списке выбрать тип файла ***Библиотека Полигон 2 (*.lI2)***.

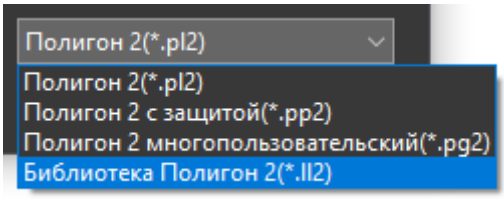


Рисунок 2.2 – Добавление библиотеки palEC104 в проект

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

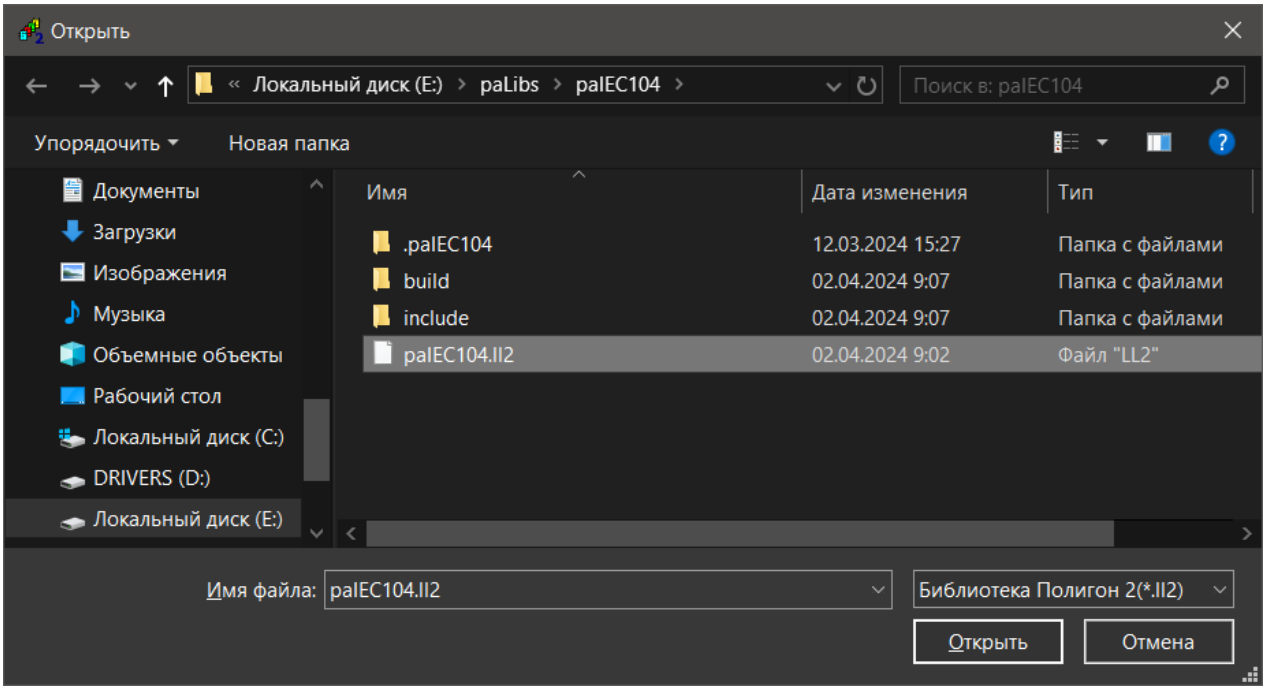


Рисунок 2.3 – Добавление библиотеки palEC104 в проект

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, может быть размещен только в ***Фоне***.

Таблица 2.1 – Назначение входов и выходов IEC101uni

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

Продолжение таблицы 2.1

Элемент	Описание
tm2	Не используется
dbg	Конфигурация отладки. Битовая маска: **Бит 0…7** – битовая маска, определяющая какие типы сообщений выводить: **бит 0** – тип 1, ..., **бит 7** – тип 8 (высший уровень); **Бит 8** – выводить в консоль; **Бит 9** – выводить через блок ***RamLog*** из библиотеки ***paCore***
bo	Входы для подключения ***IECBufOut*** (циклический)
Входы (константные)	
tmp	Период вызова главной функции блока, мс
prm	Тип канального уровня: **IEC_MASTER** – инициирует соединение; **IEC_SLAVE** – ожидает соединение
ltp	Режим канального уровня: **IEC_BALANCED** – балансный; **IEC_UNBALANCED** – небалансный
oa	Не используется
lka	Адрес станции в сети, если используется (**Link Address**)
las	Длина поля адреса станции. Определяется протоколом. Допустимые значения на входе: **LINK_ADDR_NO** – адрес станции не используется; **LINK_ADDR_1** – 1 байт; **LINK_ADDR_2** – 2 байта
aas	Длина поля общего адреса ASDU. Определяется протоколом. Допустимые значения на входе: **ASDU_ADDR_1** – 1 байт; **ASDU_ADDR_2** – 2 байта
cts	Длина поля причины передачи (COT). Определяется протоколом. Допустимые значения на входе: **COT_1** – 1 байт; **COT_2** – 2 байта
iaa	Длина адреса объекта информации. Определяется протоколом. Допустимые значения на входе: **IOA_1** – 1 байт; **IOA_2** – 2 байта; **IOA_3** – 3 байта
c1s	Глубина очереди сообщений класса 1
c2s	Глубина очереди сообщений класса 2
ofp	Если в течении этого времени (в мс) нет обмена по интерфейсу на выход **sts бит 0** блока выдается значение **0** – нет связи
frg	Максимальное межсимвольное расстояние при приеме кадра. Если это время (в мс) истекло – кадр считается недействительным
cpp	Период отправки с причиной передачи циклический, мс. Необходимость передачи данных с этой причиной задается для каждого выходного буфера. Рекомендуется задавать **cpp ≠ bpp** так, чтобы моменты срабатывания таймеров как можно реже пересекались
bpp	Период отправки сообщений с причиной передачи фоновое сканирование, мс. Необходимость передачи данных с этой причиной задается для каждого выходного буфера. Рекомендуется задавать **cpp ≠ bpp** 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: **Бит 0** – синхронизировать время после установления связи (используется системное время); **Бит 1** – послать команду общего опроса после установления связи; **Бит 4** – блокировать команду синхронизации времени; **Бит 5** – включать метку времени при отправке данных с причиной передачи: циклические; **Бит 6** – включать метку времени при отправке данных с причиной передачи: фоновое сканирование; **Бит 8** – использовать структурированный адрес, прибавление стандартного смещения (для ТИТ смещение – **0x2000**, для ТС смещение – **0x1000**); **Бит 12** – подключить протокол отправки диагностических сообщений
ito	Таймаут выполнения команды информация о процессе в направлении управления **C** (см. Приложение А), мс
sto	Таймаут выбора команды информация о процессе в направлении управления **C** (см. Приложение А), мс
Выходы	
itr	Выход для подключения ***IECBufIn***
st1	Выход статуса. Битовая маска: **Бит 0** – есть связь; **Бит 1** – выполнена команда синхронизации времени; **Бит 2** – первичный канальный уровень активен; **Бит 3** – вторичный канальный уровень активен
st2	Не используется

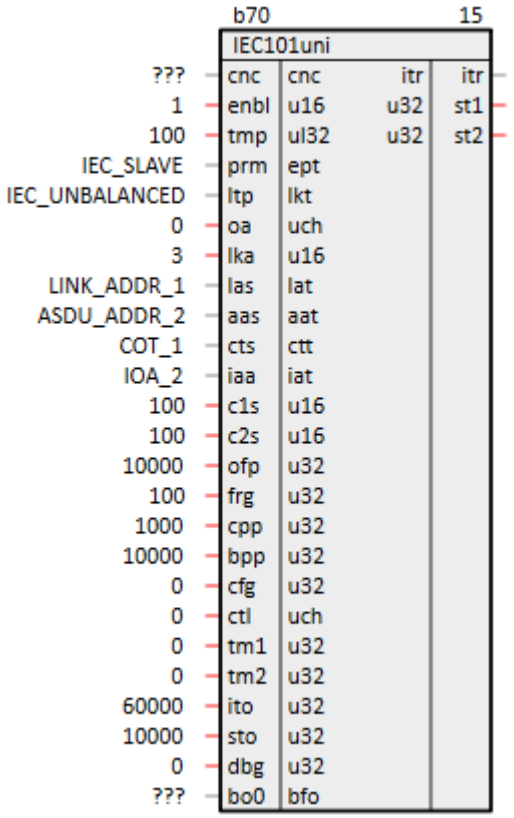


Рисунок 2.4 – Блок IEC101uni

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 ***paIEC104*** – **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.

Так как работа блока занимает значительное время, может быть размещен только в ***Фоне***.

Таблица 2.2 – Назначение входов и выходов IEC104uni

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

Продолжение таблицы 2.2

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

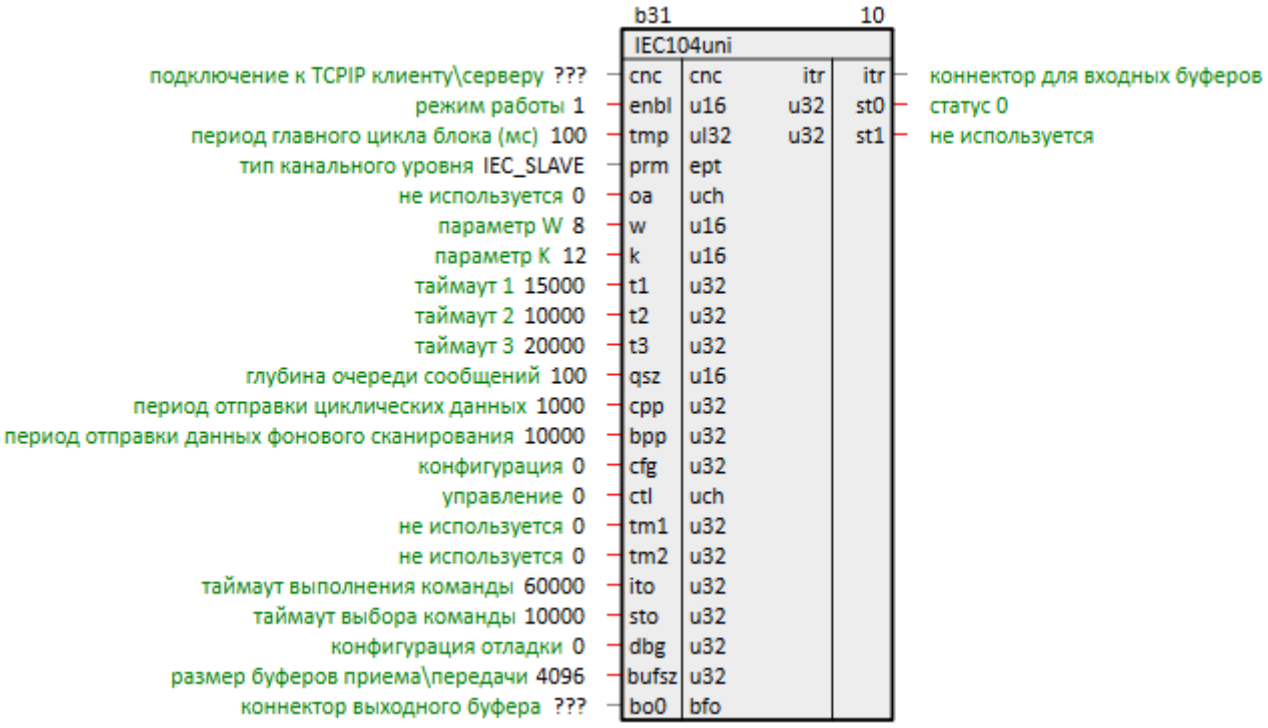


Рисунок 2.5 – Блок IEC104uni

Рассмотрим алгоритм работы связки блоков ***TcpIpClA*** 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 ***TcpIpClA***).

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 (длина задается **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.

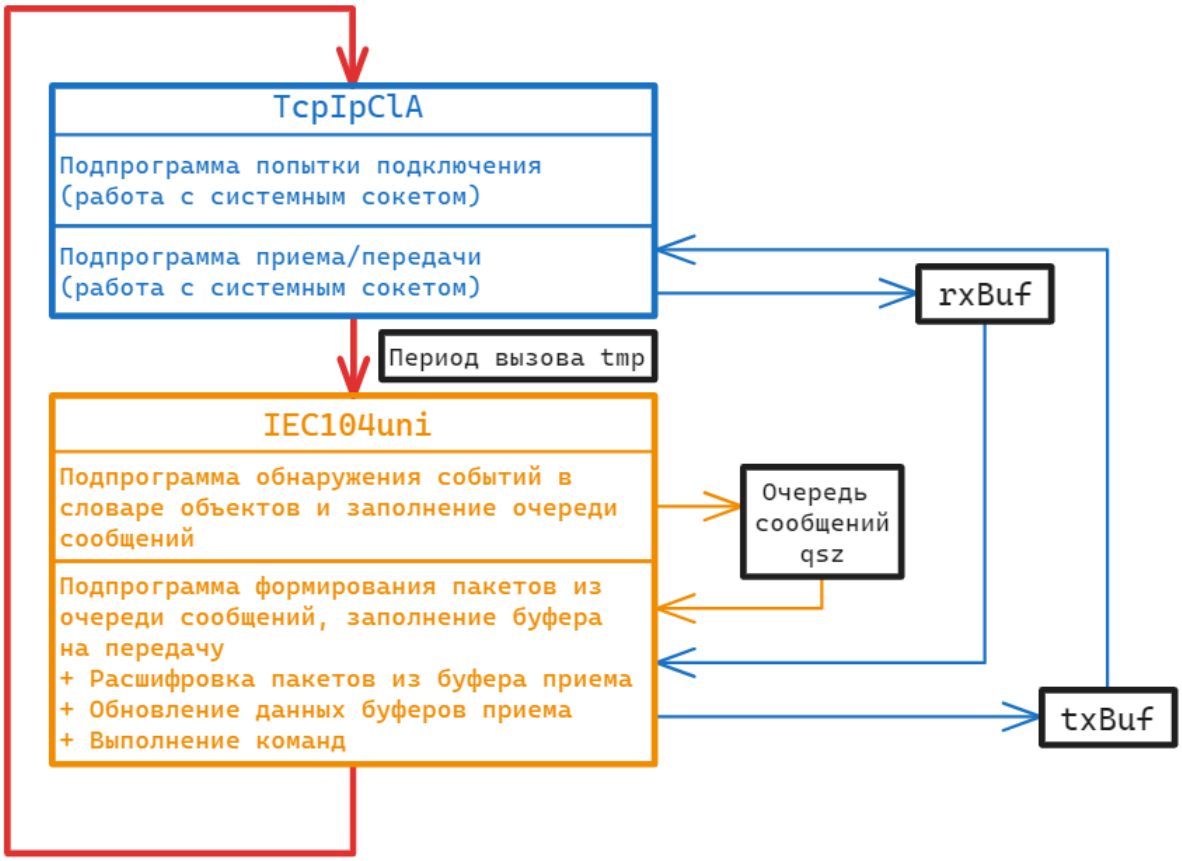


Рисунок 2.6 – Работа IEC104uni: последовательность вызова подпрограмм в фоновом потоке

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 нагруженности 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.

Так как работа блока занимает значительное время, может быть размещен только в ***Фоне***.

Таблица 2.3 – Назначение входов и выходов IEC104Server

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

Продолжение таблицы 2.3

Элемент	Описание
dbg	Конфигурация отладки. Битовая маска: **Бит 0…7** – битовая маска, определяющая какие типы сообщений выводить: **бит 0** – тип 1, ..., **бит 7** – тип 8 (высший уровень); **Бит 8** – выводить в консоль; **Бит 9** – выводить через блок ***RamLog*** из библиотеки ***paCore***; Бит 10 – выводить через протокол отправки диагностических сообщений (для разработчиков); **Бит 11** – выводить в консоль данные таймеров времени выполнения алгоритма и периода вызова алгоритма
bo	Входы для подключения ***IECBufOut*** (циклический)
Входы (константные)	
tmp	Период вызова подпрограммы обнаружения событий, мс (см. рисунок 2.8)
lip	Локальный IP-адрес
lprt	Локальный порт
sdr	Сетевой стек, для ПЛК ОВЕН **"/"**
evntsprio	Приоритет потока обработки событий (см. рисунок 2.8)
rxtxprio	Приоритет потока приема/передачи (см. рисунок 2.8)
slts	Ограничение количества одновременных подключений клиентами
rxbsz	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. **t2**): **w < k**, рекомендуется **w = 2/3k**
k	Число пакетов, которое может отправить узел без подтверждения. Если **k** пакетов отправлено без подтверждения, то передача останавливается до получения подтверждения (см. **t1**): **w < k**, рекомендуется **w = 2/3k**
t1	Таймаут при посылке кадров, мс. Если через время **t1** не было получено подтверждение – соединение закрывается: **t2 < t1**
t2	Таймаут для подтверждения, если число принятых пакетов **w** не достигнуто, мс: **t2 < t1**
t3	Таймаут для посылки блоков тестирования в случае долгого простоя, мс. С периодом **t3** передается тестовый пакет для проверки канала, если ответа на тестовый пакет не последует – соединение закрывается
qsz	Глубина очереди сообщений
cpp	Период отправки с причиной передачи циклический, мс. Необходимость передачи данных с этой причиной задается для каждого выходного буфера. Рекомендуется задавать **cpp ≠ bpp** так, чтобы моменты срабатывания таймеров как можно реже пересекались
bpp	Период отправки сообщений с причиной передачи фоновое сканирование, мс. Необходимость передачи данных с этой причиной задается для каждого выходного буфера. Рекомендуется задавать **cpp ≠ bpp** 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: **Бит 0** – синхронизировать время после установления связи (используется системное время); **Бит 1** – послать команду общего опроса после установления связи; **Бит 2** – использовать глобальное время (UTC) в метках времени; **Бит 3** – принудительно считать временные метки действительными; **Бит 4** – блокировать команду синхронизации времени; **Бит 5** – включать метку времени при отправке данных с причиной передачи: циклические; **Бит 6** – включать метку времени при отправке данных с причиной передачи: фоновое сканирование; **Бит 7** – включать метку времени при отправке данных с причиной передачи: общий опрос; **Бит 8** – использовать структурированный адрес, прибавление стандартного смещения (для ТИТ смещение – **0x2000**, для ТС смещение – **0x1000**); **Бит 9** – отключить замещение старых данных новыми в очереди
ito	Таймаут выполнения команды информация о процессе в направлении управления **С** (см. Приложение А), мс
sto	Таймаут выбора команды информация о процессе в направлении управления **С** (см. Приложение А), мс
Выходы	
itr	Выход для подключения ***IECBufIn***
st0	Выход статуса. Битовая маска: **Бит 0** – есть связь; **Бит 1** – выполнена команда синхронизации времени
st1	Не используется

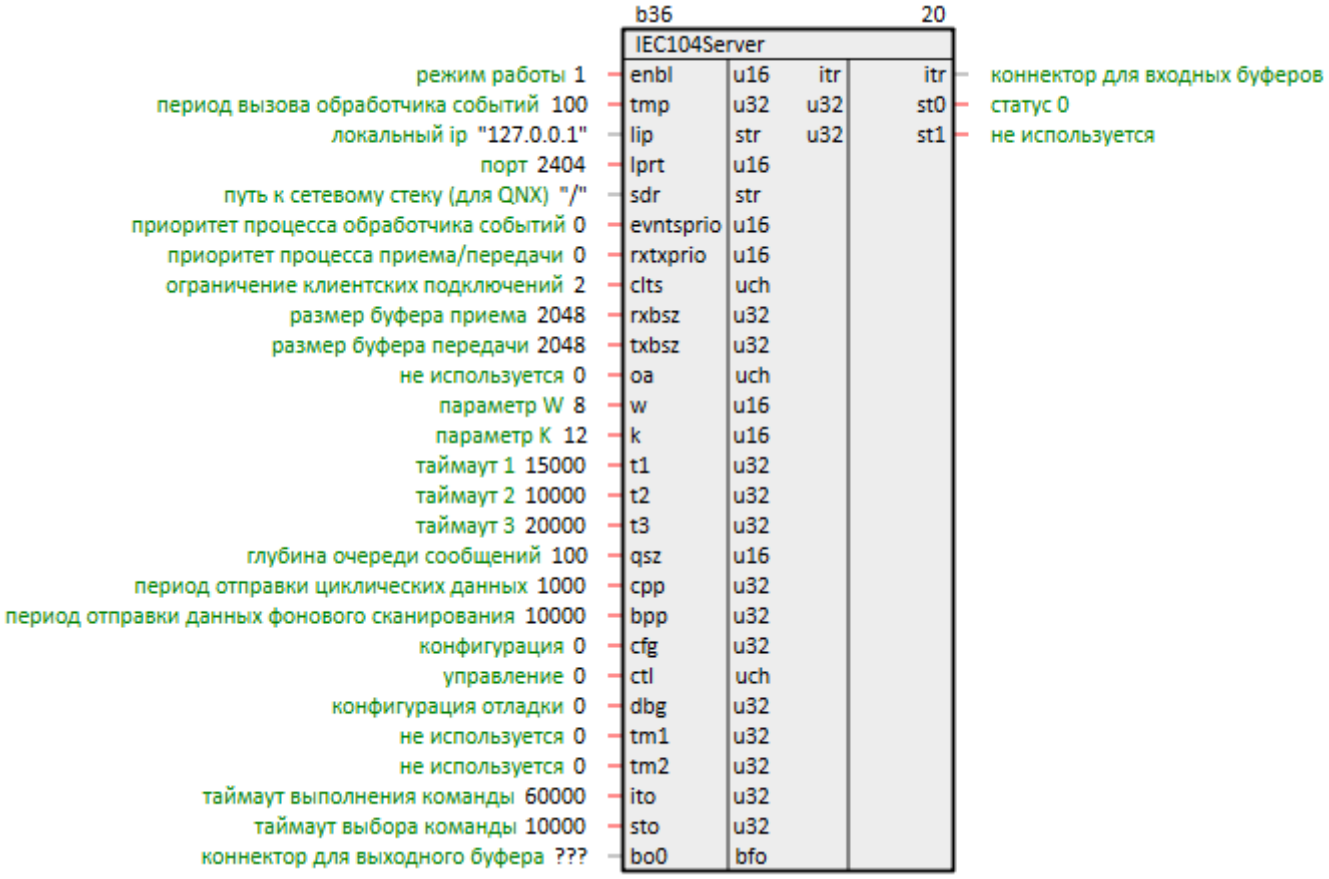


Рисунок 2.7 – Блок IEC104Server

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 (длина задается **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.

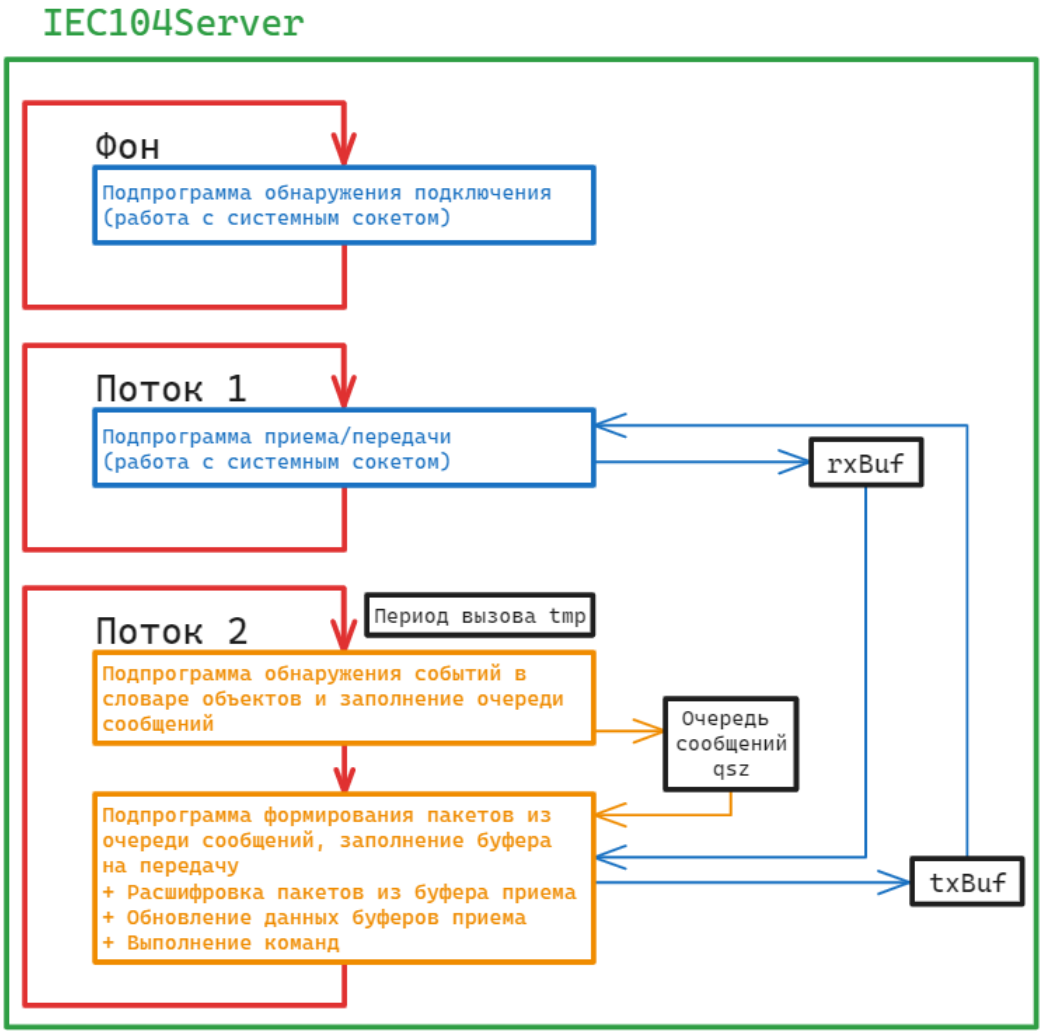


Рисунок 2.8 – Работа IEC104Server: разделение на потоки и последовательность вызова подпрограмм

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.

Может быть размещен только в ***Фоне***.

Таблица 2.4 – Назначение входов и выходов IECBufIn

Элемент	Описание
Входы	
itr	Вход для подключения к блоку протокола
ini	Вход данных для начальной инициализации выхода (циклический)
ctl	Не используется (циклический)
Входы (константные)	
tid	Тип ASDU (**Type ID**) для всех объектов информации в пределах буфера. Определяется протоколом. Допустимые значения на входе описаны в справке среды на библиотеку и в Приложение А
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[1)] (циклический). Определяется средой Полигон. Допустимые значения на входе: **DO** (Char), **uDO** (Unsigned Char) – символьный; **IO** (Short), **uIO** (Unsigned Short) – 16-битное целое; **LO** (Long), **uLO** (Unsigned Long), **LX** (int32_t), **uLX** (uint32_t) – 32-битное целое; **AO** (Float) – число с плавающей запятой
flg	Не используется (циклический)
Выходы	
bi	Не используется
o	Выход данных (циклический)
qlf	Квалификатор (циклический). Выход имеет различные функции, в зависимости от типа объекта информации (см. Приложение А): 1. Для типов **M_*** (кроме **M_IT_***) – как выход значения квалификатора, определяется протоколом (битовая маска): **Бит 0** – **OV**: переполнение (для **M_ME_*_1**); **Бит 4** – **BL**: блокировка (для **M_*_*_1**); **Бит 5** – **SB**: проведено замещение (для **M_*_*_1**); **Бит 6** – **NT**: неактуальное значение (для **M_*_*_1**); **Бит 7** – **IV**: недействительное значение (для **M_*_*_1**) 2. Для типов **M_IT_*** – как выход вспомогательных полей данных[2)], определяется протоколом (битовая маска): **Биты 0…4** – **SQ**: номер передаваемой последовательности; **Бит 5** – **CY**: произошло переполнение счетчика; **Бит 6** – **CA**: счетчик установлен; **Бит 7** – **IV**: показание счетчика недействительно
tm1	Выход времени, младшее слово[3)] (циклический)
tm2	Выход времени, старшее слово[3)] (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 структурированны. 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.

Может быть размещен только в ***Фоне***.

Таблица 2.5 – Назначение входов и выходов IECBufOut

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

Продолжение таблицы 2.5

Элемент	Описание
trt	Определяет причину передачи (см. Приложение А). Определяется протоколом. Допустимые значения на входе (битовая маска): **Бит 0** – передавать данные при изменении (причина передачи **3**). Для типов ASDU **M_ME_xx_1** посредством входа **flg** есть возможность задать дополнительные условия передачи; **Бит 1** – передавать данные циклически (причина передачи **1**) с периодом, указанным на входе **cpp** блока протокола; **Бит 2** – передавать данные фоновым сканированием (причина передачи **2**) с периодом, указанным на входе **bpp** блока протокола; **Бит 3** – отправить при активации **бита 0** на входе **ctl:** команда для типов ASDU **C_*** (причины передачи **6**, **8**, **10**); **Бит 4** – отправить при активации **бита 1** на входе **ctl** (причина передачи определяется **битом 6** на входе **flg**)
ioa	Адрес объекта информации в пределах сектора (циклический)
tp	Тип данных на выходах[1)] (циклический). Определяется средой Полигон. Допустимые значения на входе: **Dl** (Char), **uDl** (Unsigned Char) – символьный; **Il** (Short), **uIl** (Unsigned Short) – 16-битное целое; **Ll** (Long), **uLl** (Unsigned Long), **LX** (int32_t), **uLX** (uint32_t) – 32-битное целое; **Al** (Float) – число с плавающей запятой
flg	Флаги (циклический). Для типов ASDU **M_ME_xx_1** задает условия передачи спонтанных сообщений (см. Приложение А). Определяется протоколом. Допустимые значения на входе (битовая маска): **Бит 0** – передавать данные в случае, если текущее значение на входе **in** блока отличается от предыдущего более, чем на величину, определяемую входом **tsh** (**Threshold**) блока (мертвая зона); Бит 1 – не используется; **Бит 2** – передавать данные в случае, если значение на входе **in** меньше величины, определяемой входом **llm** (**Low Limit**); **Бит 3** – передавать данные в случае, если значение на входе **in** больше величины, определяемой входом **hlm** (**High Limit**); Для команд (тип ASDU **C_***): **Бит 4** – посылать команду при изменении значения на входе; **Бит 5** – использовать предварительный выбор; Для всех, кроме команд: **Бит 6** – причина передачи в принудительном режиме, т.е. при активации бита на входе **ctl**: **0** – причина передачи **1** (циклически), **1** – причина передачи **2** (фоновое сканирование)
Входы	
in	Входы данных (циклический)
qlf	Квалификатор[2)] (циклический). Вход имеет различные функции, в зависимости от типа объекта информации (см. Приложение А): 1. Для типов **M_*** (кроме **M_IT_***) – как вход значения квалификатора, определяется протоколом (битовая маска): **Бит 0** – **OV**: переполнение (для **M_ME_*_1**); **Бит 4** – **BL**: блокировка (для **M_*_*_1**); **Бит 5** – **SB**: проведено замещение (для **M_*_*_1**); **Бит 6** – **NT**: неактуальное значение (для **M_*_*_1**); **Бит 7** – **IV**: недействительное значение (для **M_*_*_1**) 2. Для типов **M_IT_*** – как вход вспомогательных полей данных[2)], определяется протоколом: **Биты 0…4** – **SQ**: номер передаваемой последовательности; **Бит 5** – **CY**: произошло переполнение счетчика; **Бит 6** – **CA**: счетчик установлен; **Бит 7** – **IV**: показание счетчика недействительно

Продолжение таблицы 2.5

Элемент	Описание
tsh	Вход данных **Threshold** (циклический). Имеет значение только для типов ASDU **M_ME_*_1** (см. Приложение А) (мертвая зона)
smt	Не используется (циклический)
llm	Вход данных **Low Limit** (циклический). Имеет значение только для типов ASDU **M_ME_*_1** (см. Приложение А)
hlm	Вход данных **High Limit** (циклический). Имеет значение только для типов ASDU **M_ME_*_1** (см. Приложение А)
ctl	Управление (циклический). Битовая маска: **Бит 0** – выполнить команду (**бит 3** на входе **trt**); **Бит 1** – отправить принудительно (**бит 4** на входе **trt**)
Выходы	
bo	Выход для подключения к блоку протокола
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 формирования структуры используется блок IECIntFromEP.
[2] 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 **lI** (Short), т.к. для инкапсуляции используется структура данных типа int16_t.

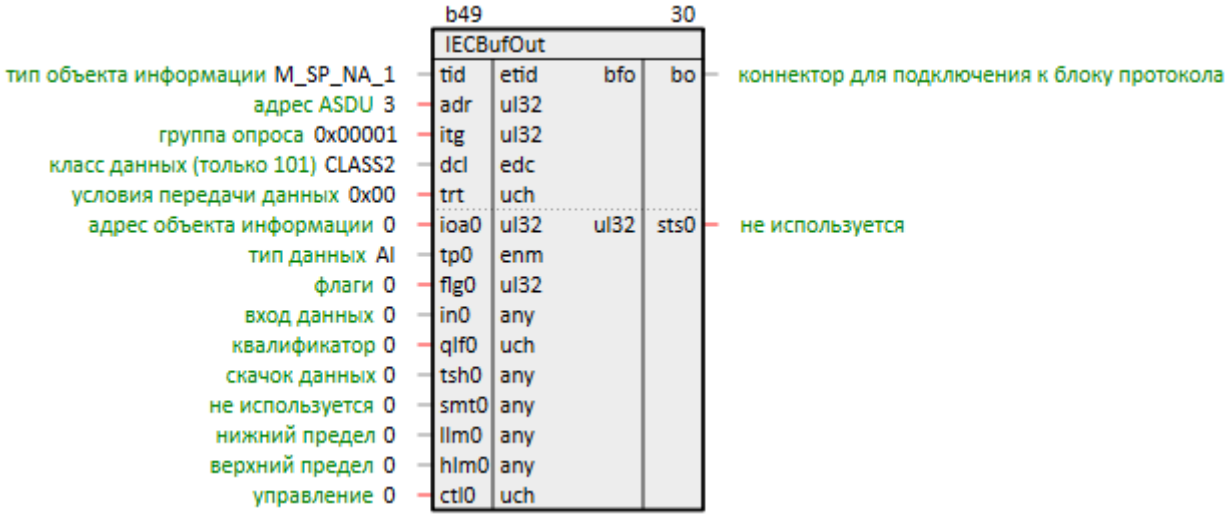


Рисунок 2.10 – Блок IECBufOut

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 А) на отдельные поля данных. Базовый тип выхода буфера (определяется входом **tp** буфера IECBufIn) должен быть **uLX**.

Может быть размещен только в ***Фоне***.

Таблица 2.6 – Назначение входов и выходов IECEPFromInt

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

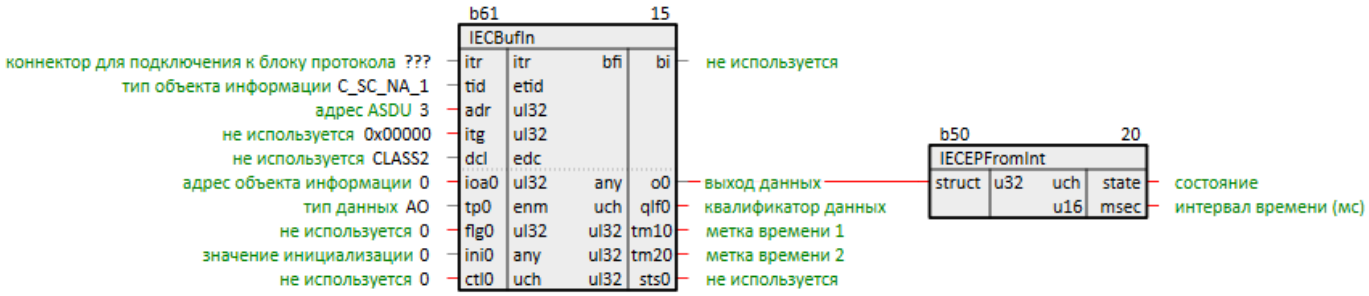


Рисунок 2.11 – Блок IECEPFromInt

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 А) в структурированное слово для использования в выходном буфере. Базовый тип входа буфера (определяется входом **tp** буфера IECBufOut) должен быть **uLX**.

Может быть размещен только в ***Фоне***.

Таблица 2.7 – Назначение входов и выходов IECIntFromEP

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

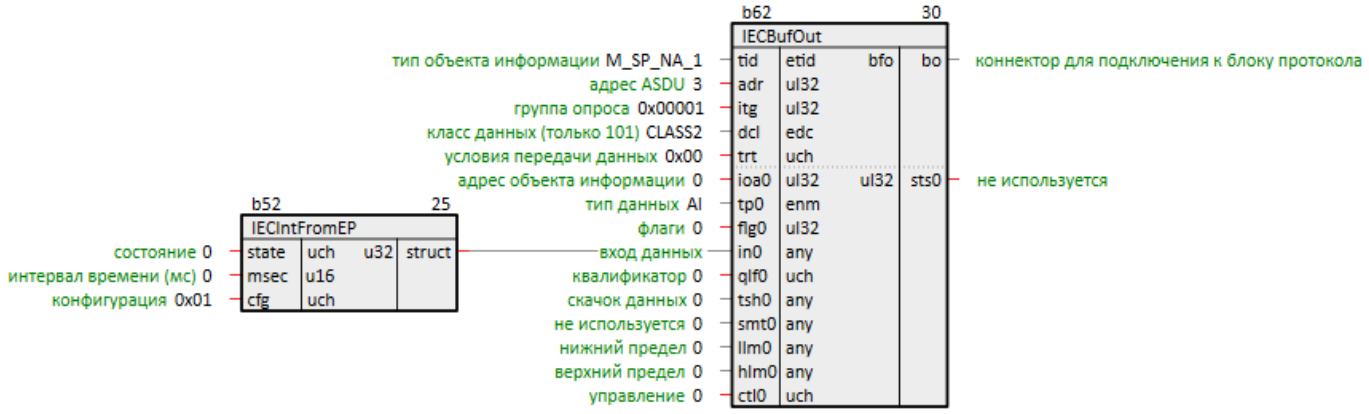


Рисунок 2.12 – Блок IECIntFromEP

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.

Может быть размещен только в ***Фоне***.

Таблица 2.8 – Назначение входов и выходов IECITSQIn

Элемент	Описание
Входы	
in	Вход для подключения к выходу данных **qlf** блока ***IECBufIn***
Выходы	
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

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.

Может быть размещен только в ***Фоне***.

Таблица 2.9 – Назначение входов и выходов IECITSQOut

Элемент	Описание
Входы	
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
Выходы	
out	Выход для подключения к входу **qlf** блока ***IECBufOut***

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.

Может быть размещен только в ***Фоне***.

Таблица 2.10 – Назначение входов и выходов IECTransTime

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

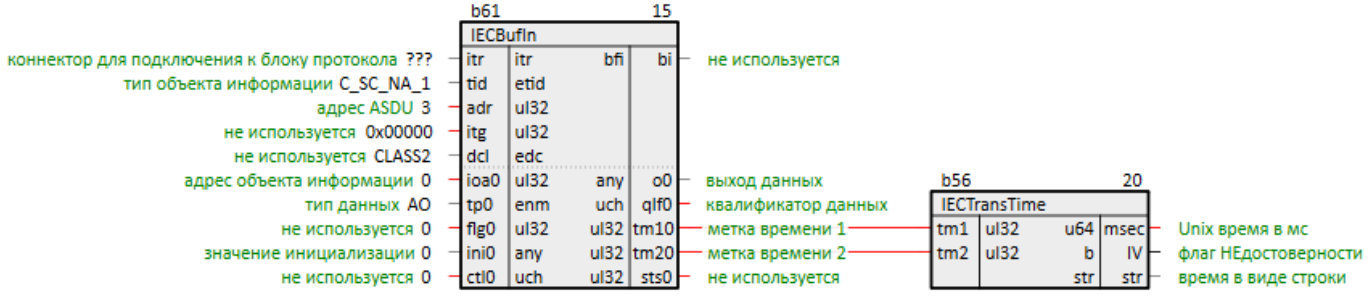


Рисунок 2.15 – Блок IECTransTime

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
Значения **msgQ*** считаются после выполнения подпрограммы 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.

Может быть размещен только в ***Фоне***.

Таблица 2.11 – Назначение входов и выходов IECInfo

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

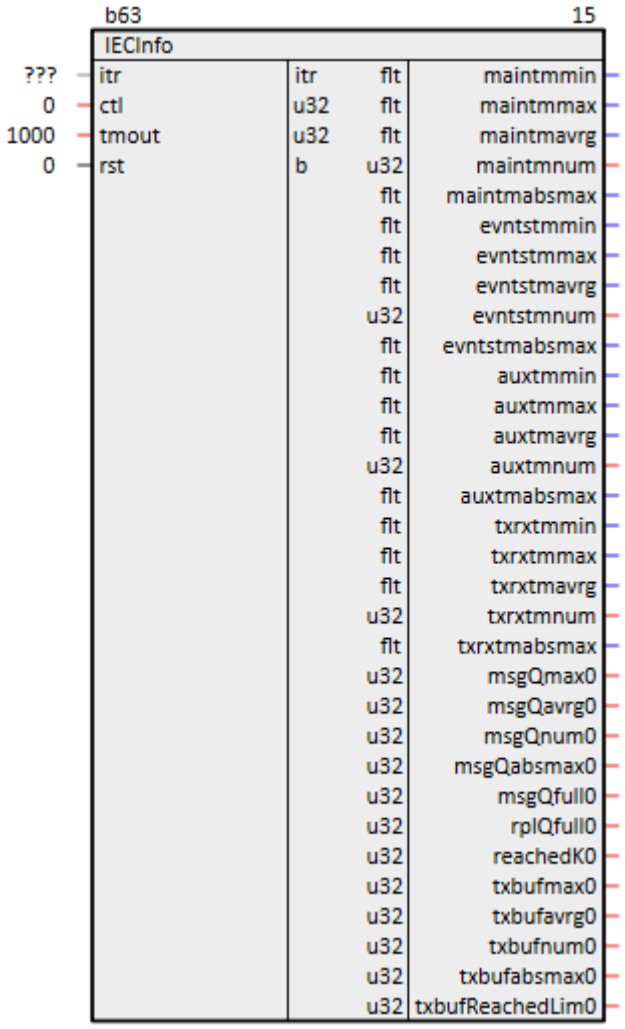


Рисунок 2.16 – Блок IECInfo

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

3.1 Настройка обмена в режиме сервера протокола 5-104

Настроим обмен 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**.

Добавим на любую страницу места работы ***Фон*** блок IEC104Server.

На вход **lip** подадим 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>"`

Добавим коннектор для выходного буфера IECBufOut.

Установим период для отправки данных с причиной **1** (циклический) равным 5 секундам – **cpp = 5000**.

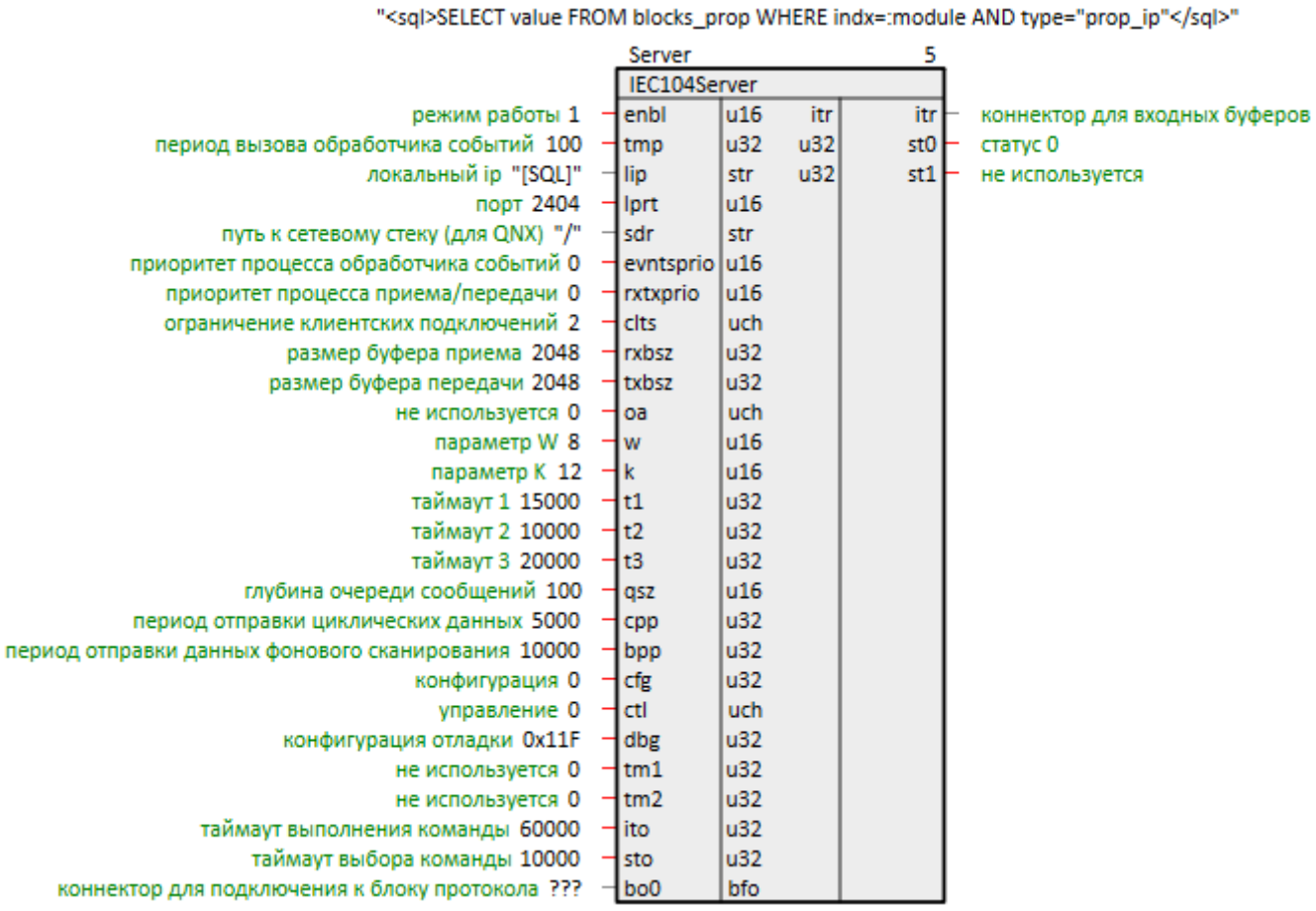


Рисунок 3.1 – Настройка IEC104Server

3.1.1 Передача информации о процессе в направлении контроля (M)

Рассмотрим обмен данными в направлении контроля (метка **M** – см. Приложение А).

Рассмотрим отправку объекта информации с типом **M_SP_NA_1** – одноэлементный объект информации с описателем качества.

1. Добавим 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.

2. Установим **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.

3. Условия передачи данных **trt** установим 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.

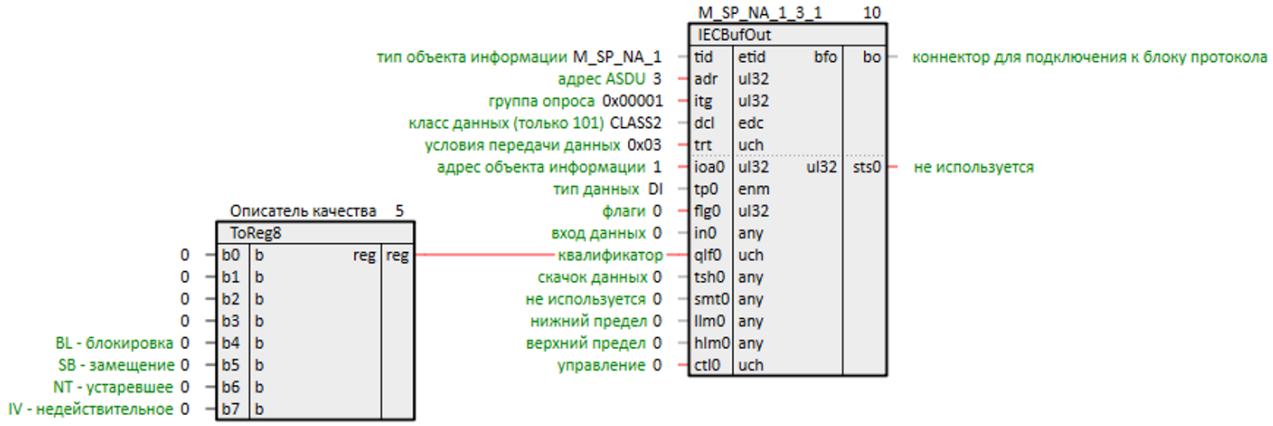


Рисунок 3.2 – Настройка IECBufOut. M_SP_NA_1

4. В качестве клиента будем использовать Multi-Protocol MasterOPC. Можно также запустить в качестве клиента проект, рассмотренный в разделе 3.1, на виртуальном контроллере с помощью панели отладки.

5. Настройка клиента в Multi-Protocol MasterOPC показана на рисунках ниже.

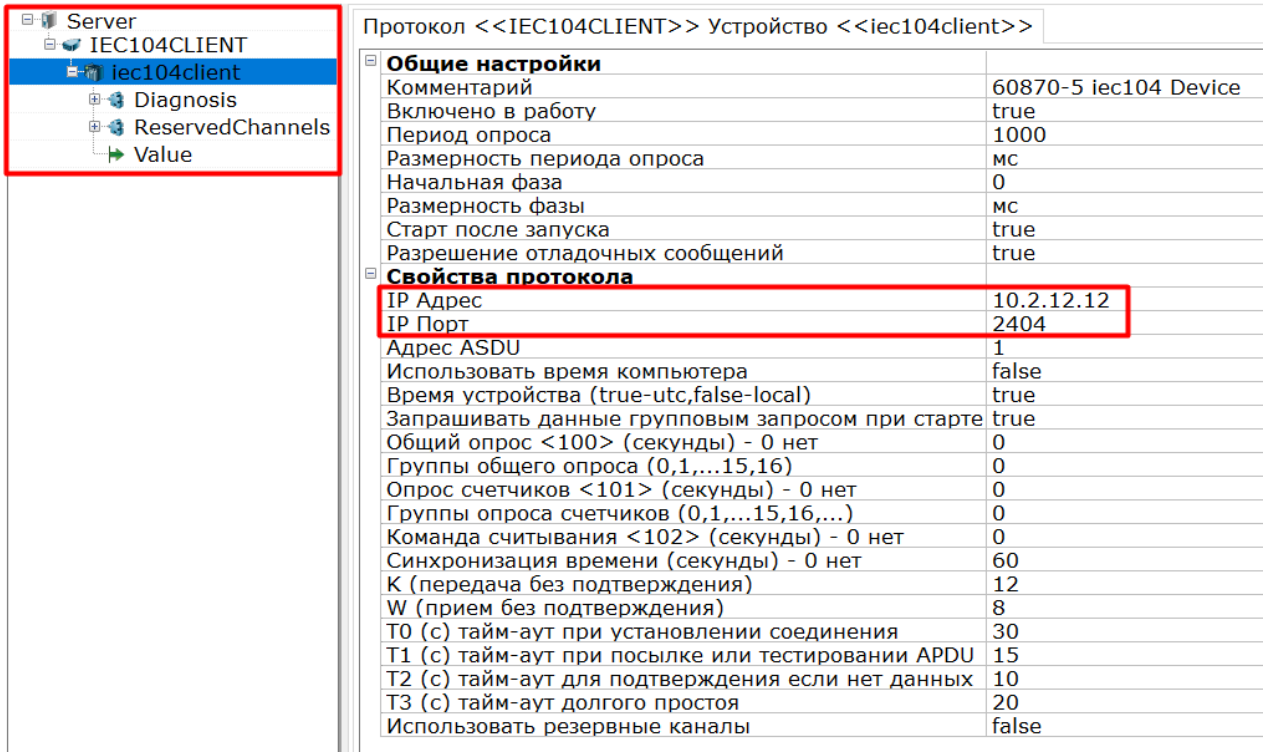


Рисунок 3.3 – Настройки Multi-Protocol MasterOPC

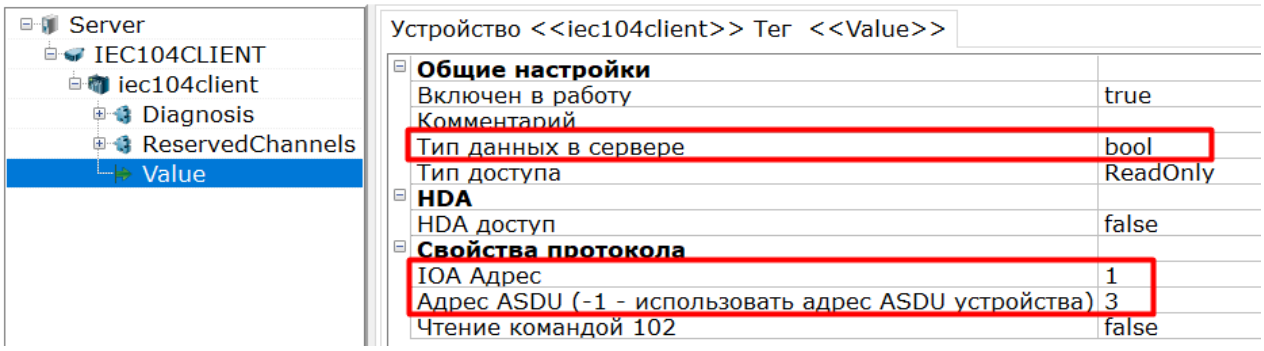


Рисунок 3.4 – Настройки канала Multi-Protocol MasterOPC

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

Рисунок 3.5 – Корректный обмен. Сервер

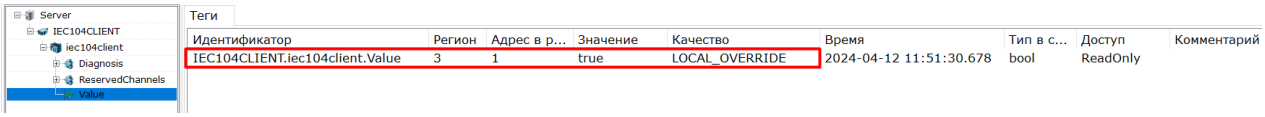


Рисунок 3.6 – Корректный обмен. Клиент

Рассмотрим отправку объекта информации с типом **M_ME_TF_1** – ТИ с описателем качества и меткой времени.

1. Добавим выходной буфер ***IECBufOut***.
2. Установим **tid = M_ME_TF_1**, адрес ASDU **adr = 3**, адрес объекта информации **ioa = 2**.

   Тип данных **tp** установим равным **AI**, так как **M_ME_TF_1** – величина с плавающей запятой.
3. Условия передачи данных **trt** установим 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** активируем **бит 0** – учет мертвой зоны на входе **tsh**. Установим **tsh = 5**.
6. Установим также нижний и верхний пределы равными, соответственно, **llm = 50** и **hlm = 200**.

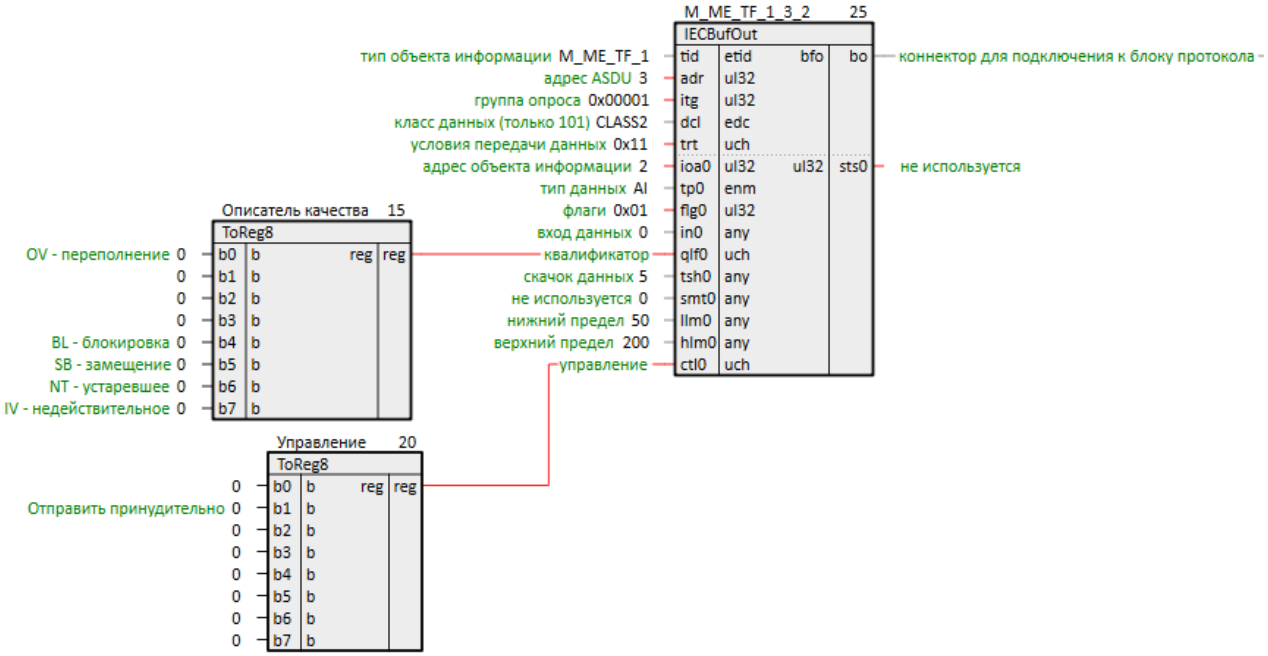


Рисунок 3.7 – Настройка IECBufOut. M_ME_TF_1

7. Добавим коннектор для подключения выходного буфера ***IECBufOut*** у блока ***IEC104Server***.
8. Подключим ***IECBufOut*** к коннектору блока ***IEC104Server***.
9. Настроим прием ТИ в клиенте:

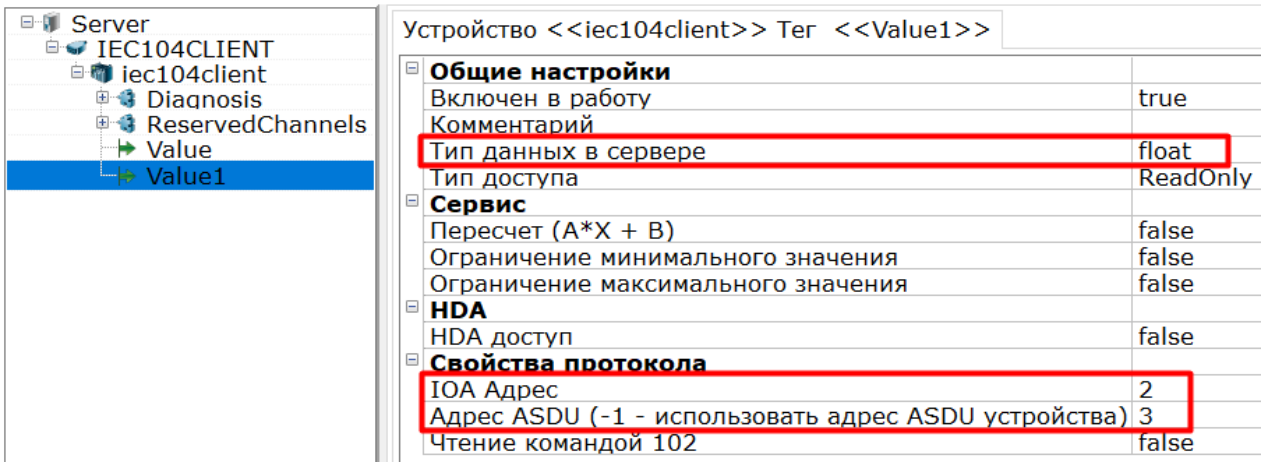


Рисунок 3.8 – Настройки OPC-сервера протокола 5-104

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

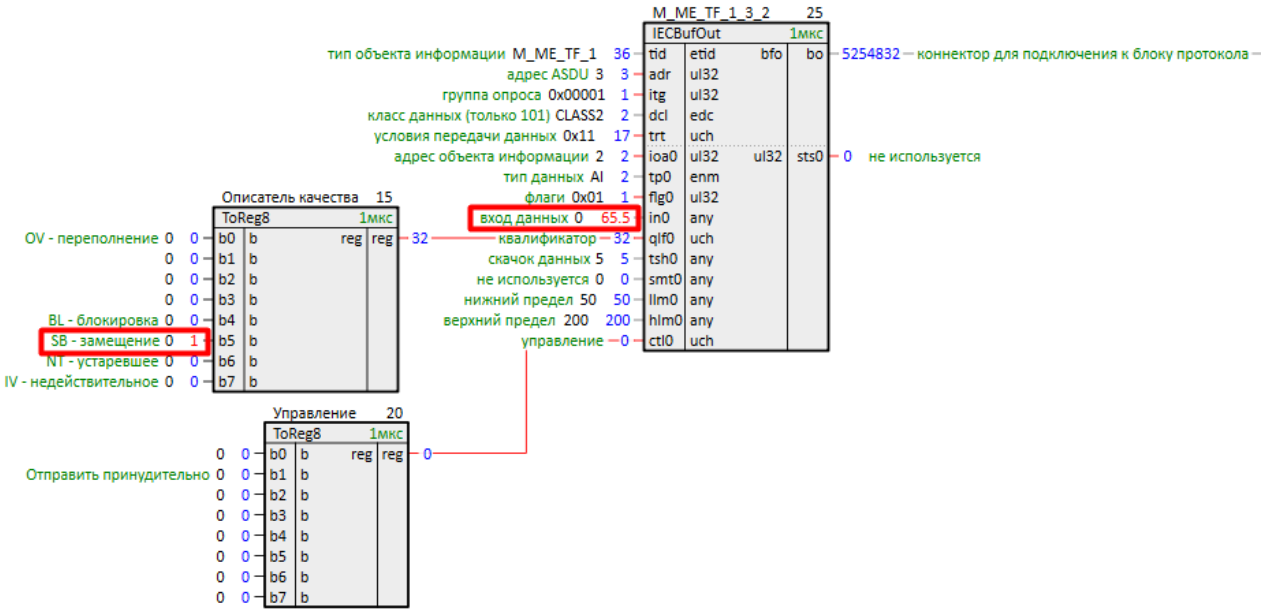


Рисунок 3.9 – Корректный обмен. Сервер

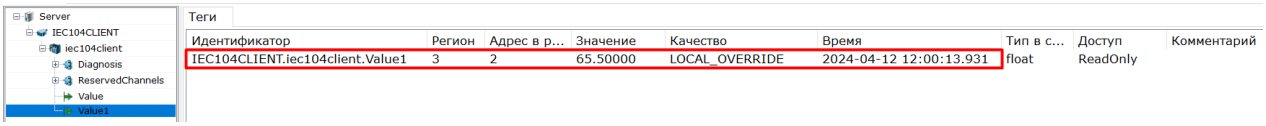


Рисунок 3.10 – Корректный обмен. Клиент

Если теперь мы на входе данных передадим значение **70** – значение клиенту не передастся, так как не была преодолена мертвая зона **tsh = 5**.

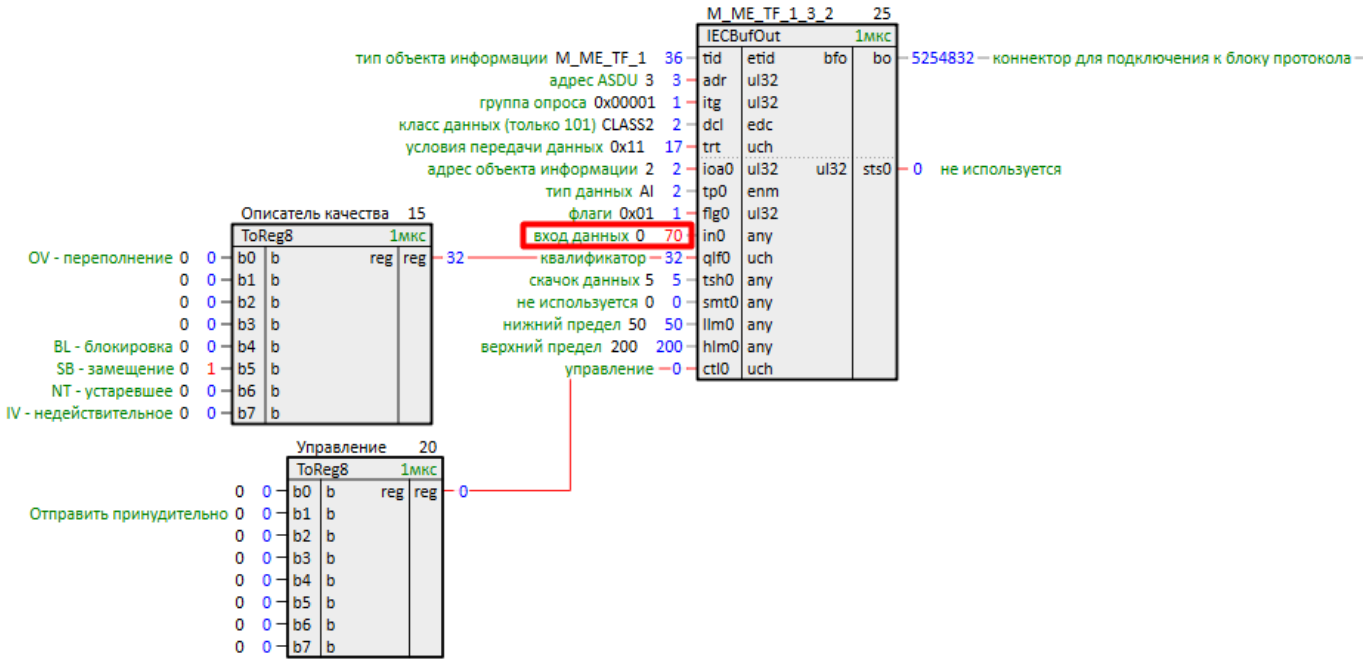


Рисунок 3.11 – Влияние параметра tsh. Сервер

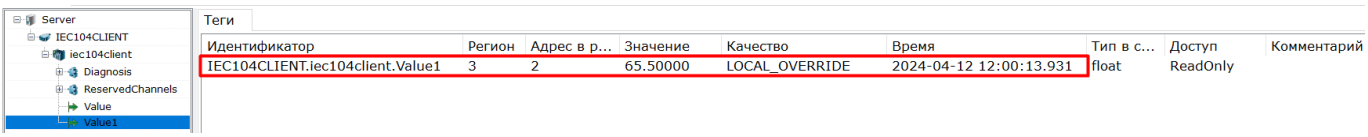


Рисунок 3.12 – Влияние параметра tsh. Клиент

Для принудительной отправки установим **бит 1** на входе **ctl**.

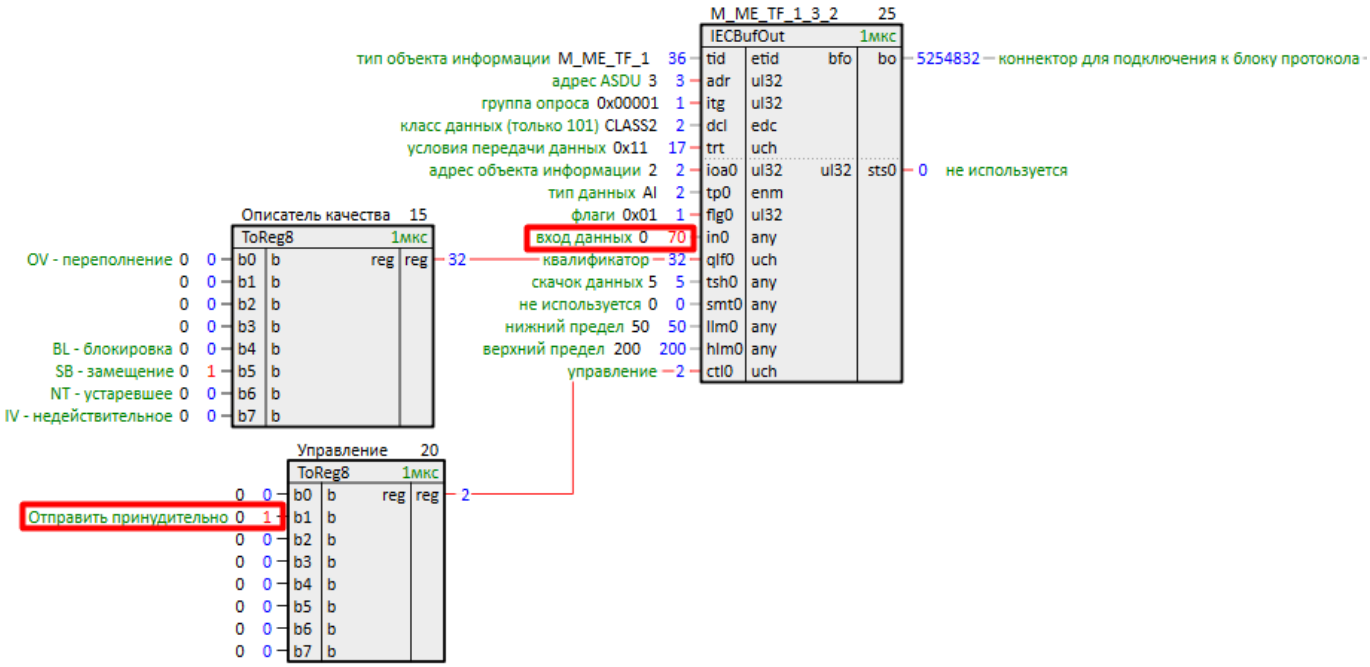


Рисунок 3.13 – Принудительная отправка. Сервер

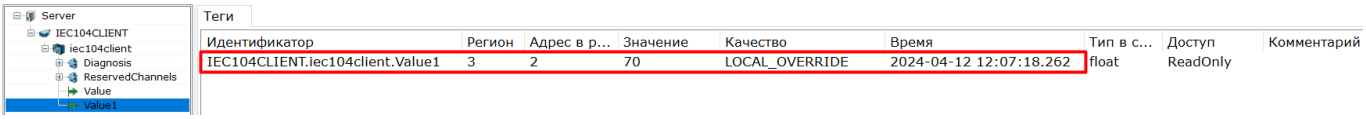


Рисунок 3.14 – Принудительная отправка. Клиент

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.

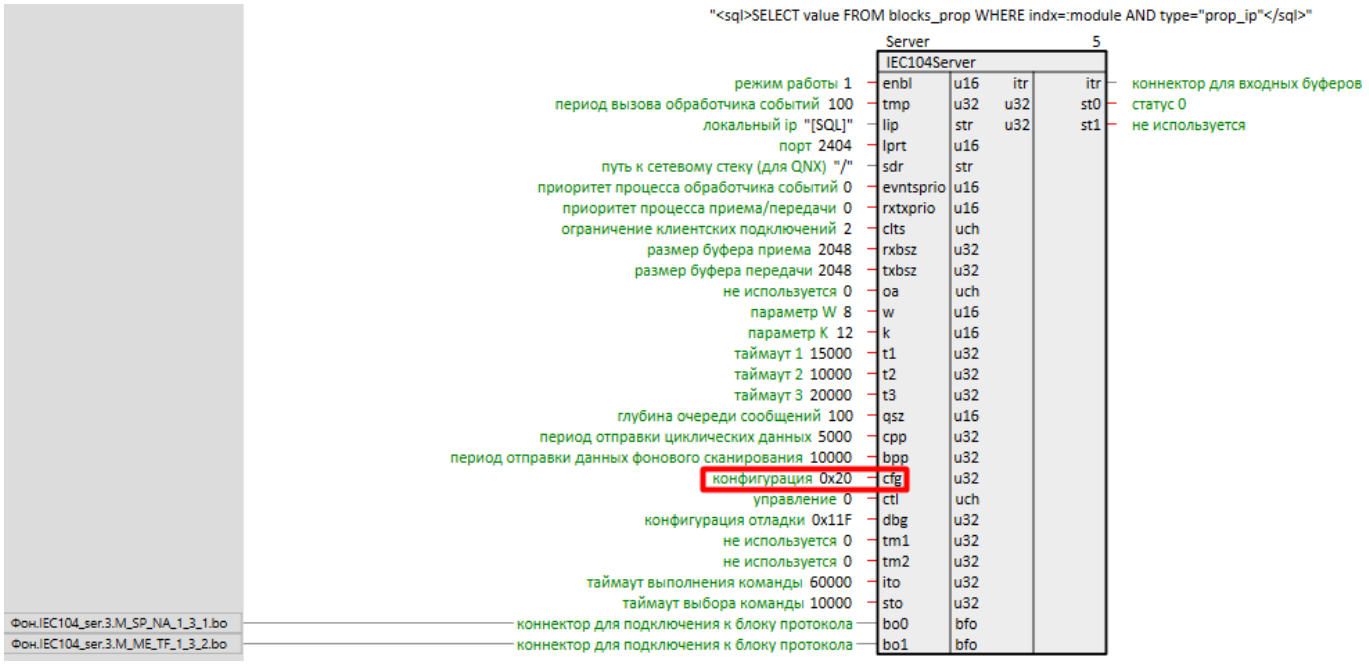


Рисунок 3.16 – Передача метки времени при циклической передаче

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 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

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

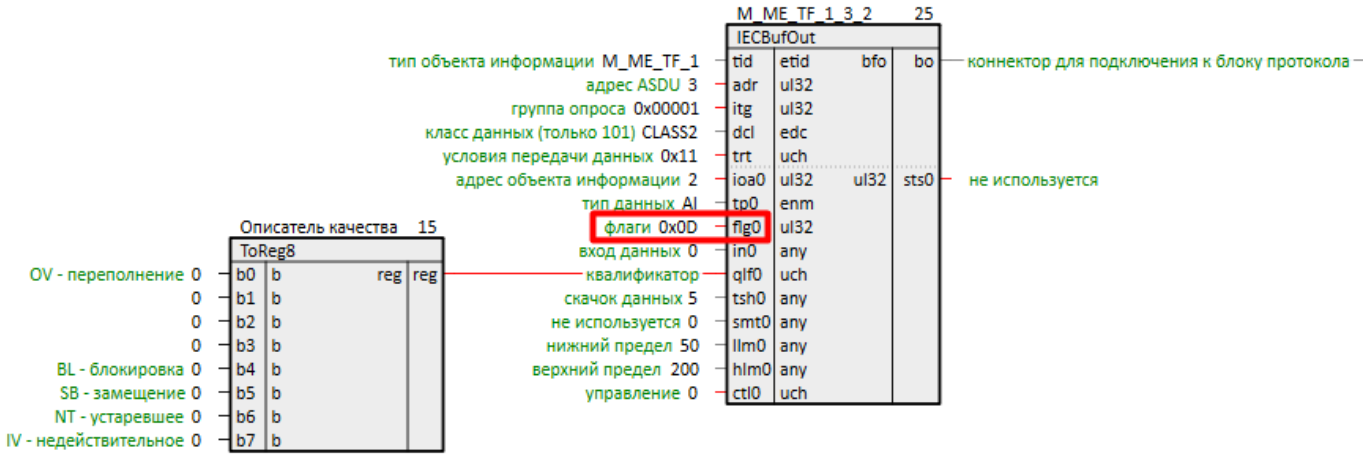


Рисунок 3.18 – Учет нижнего и верхнего порогов

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

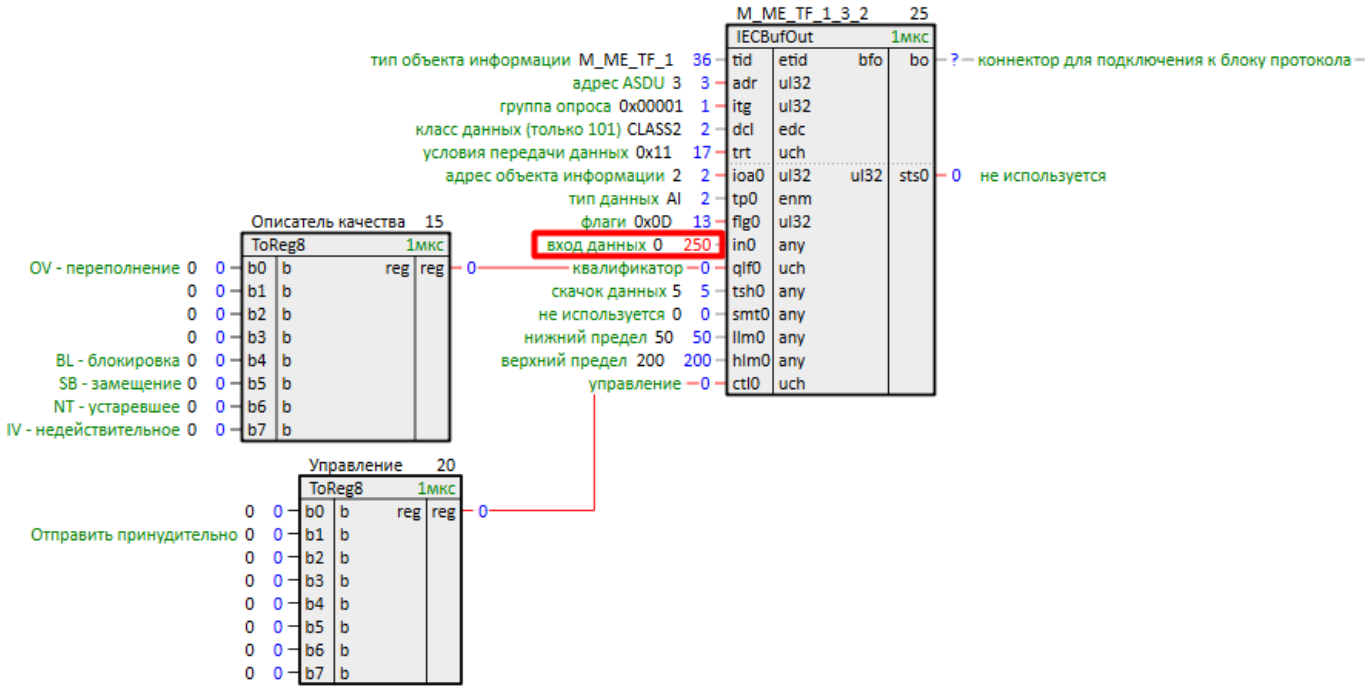


Рисунок 3.19 – – Передача при превышении hlm. Сервер

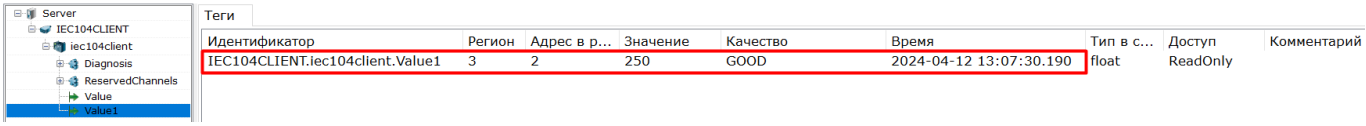


Рисунок 3.20 – Передача при превышении hlm. Клиент

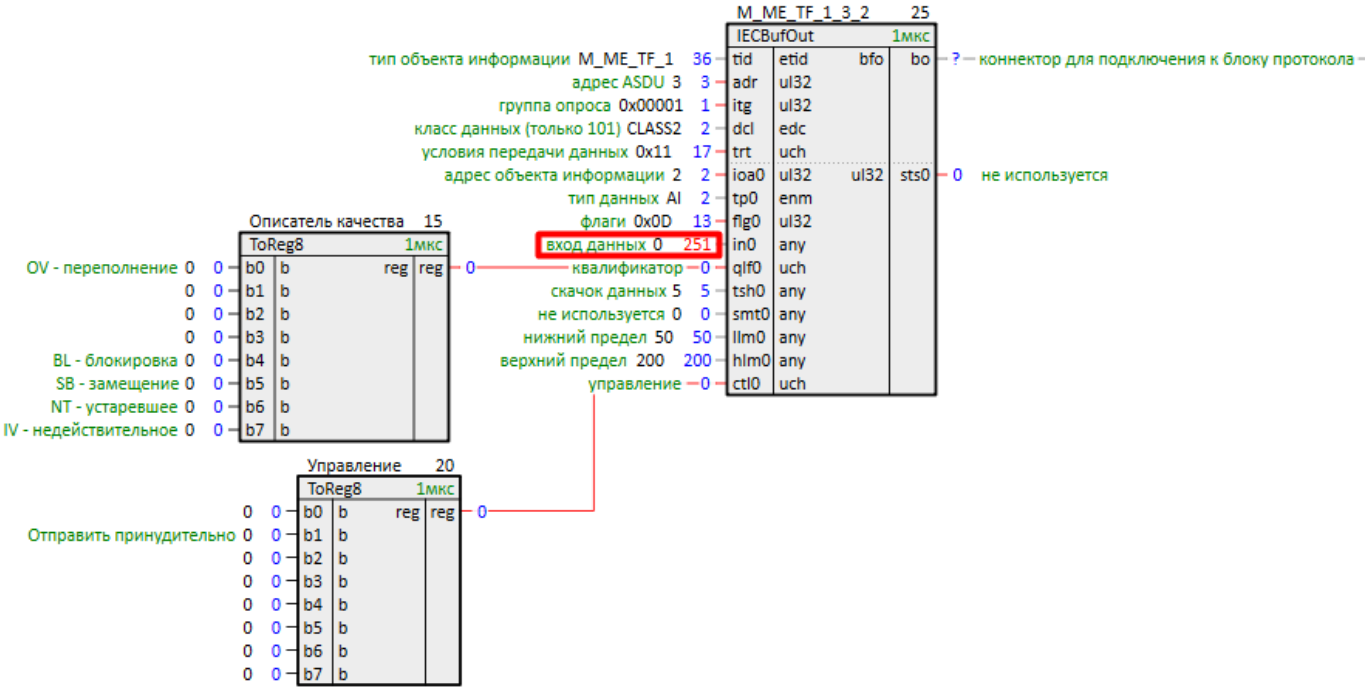


Рисунок 3.21 – Передача при превышении hlm. Сервер

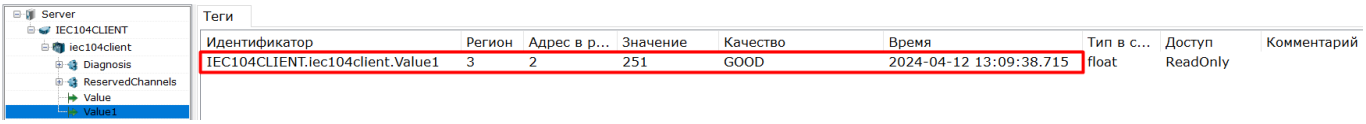


Рисунок 3.22 – Передача при превышении hlm. Клиент

3.1.2 Прием информации о процессе в направлении управления (C)

Рассмотрим обмен данными в направлении управления (метка **C** – см. Приложение А).

Рассмотрим прием объекта информации с типом **C_SC_NA_1** – однопозиционная команда ТУ.

1. Добавим на любую страницу места работы ***Фон*** входной буфер IECBufIn.
2. Установим **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.

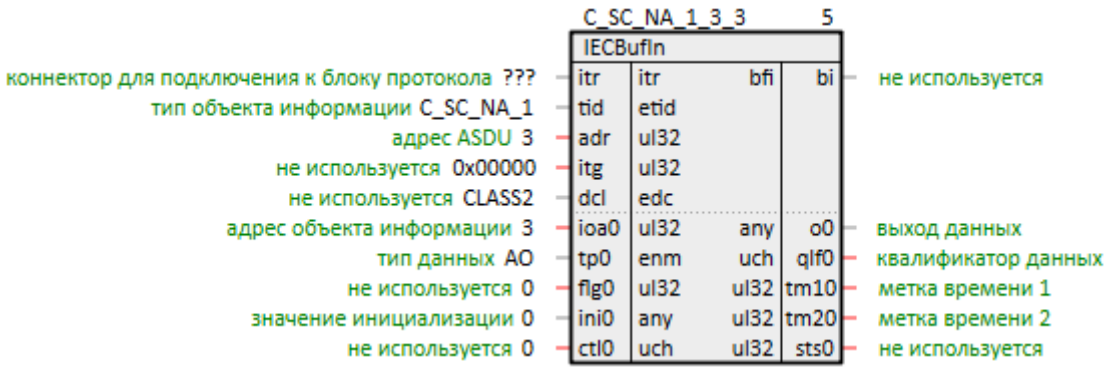


Рисунок 3.23 – Настройка IECBufIn. C_SC_NA_1

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

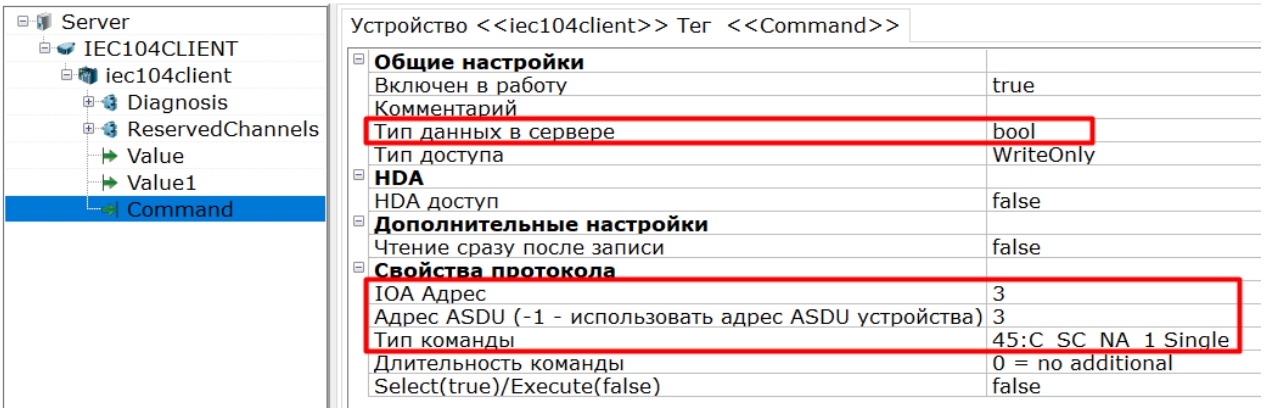


Рисунок 3.24 – Настройки команды Multi-Protocol MasterOPC

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

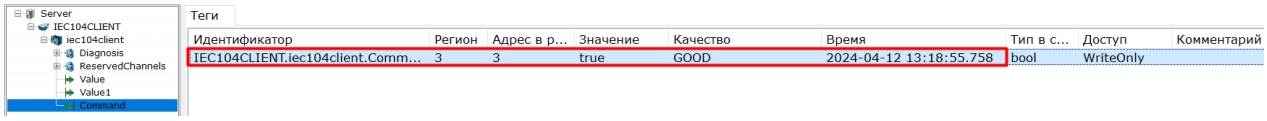


Рисунок 3.25 – Корректный обмен. Клиент

Рисунок 3.26 – Корректный обмен. Сервер

6. Рассмотрим прием объекта информации с типом **C_SE_TC_1** – команда уставки с меткой времени.
7. Добавим входной буфер ***IECBufIn***.
8. Установим **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.
9. Для расшифровки метки времени соединим выходы **tm1** и **tm2** с входами блока IECTransTime.

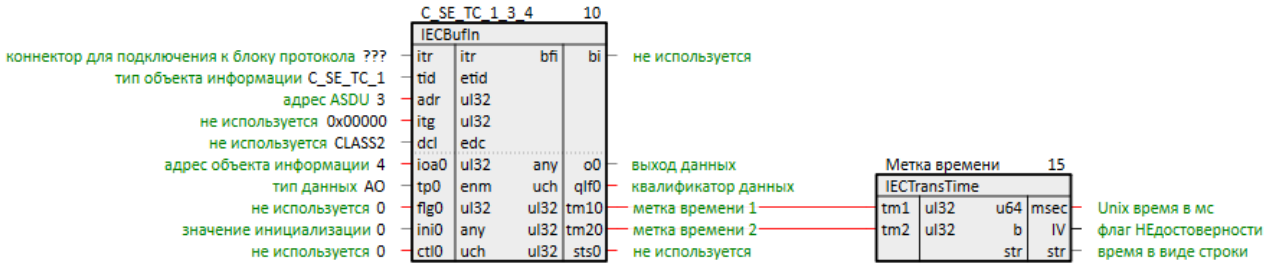


Рисунок 3.27 – Настройка IECBufIn. C_SE_TC_1

10. Подключим ***IECBufIn*** к коннектору блока ***IEC104Server***.
11. Настроим команду в Multi-Protocol MasterOPC:

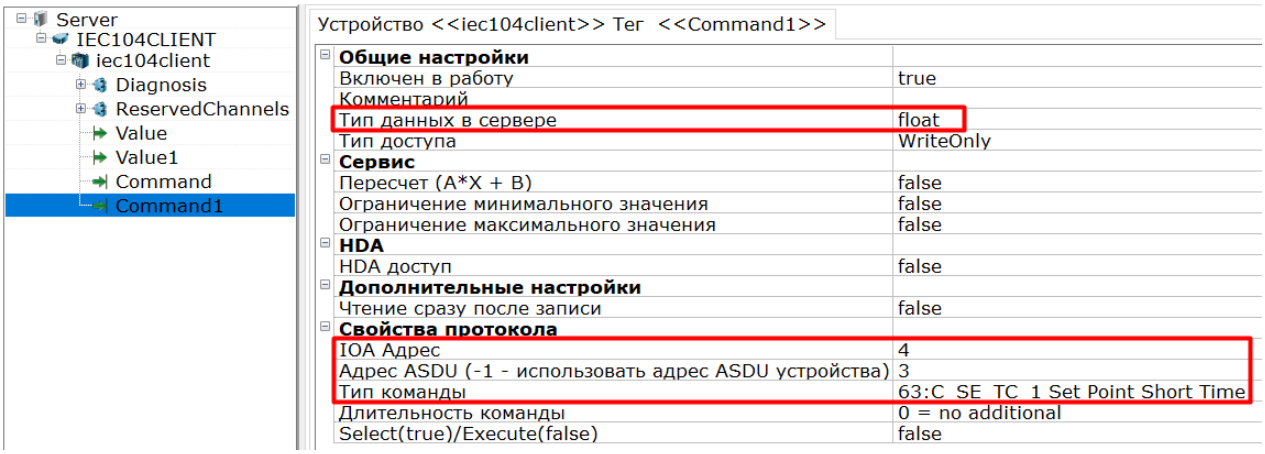


Рисунок 3.28 – Настройки команды Multi-Protocol MasterOPC

12. Запустим программы на ПЛК и OPC-сервере. Корректный обмен показан на рисунках ниже.

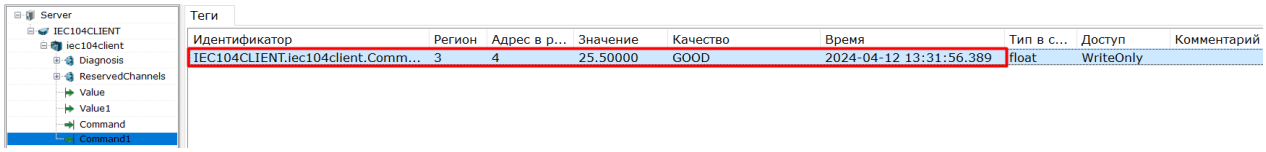


Рисунок 3.29 – Корректный обмен. Клиент

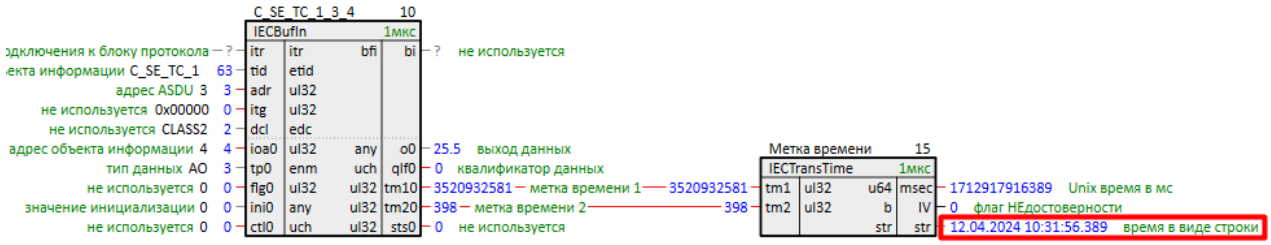


Рисунок 3.30 – Корректный обмен. Сервер

3.1.3 Передача упакованных данных (M_EP_*)

Рассмотрим передачу объекта информации с типом **M_EP_TE_1** – упакованное сообщение с меткой времени **CP56Время2а**. Особенность данного типа состоит в том, что в сообщении передается структура, состоящая из битовой маски и интервала времени.

1. Добавим на любую страницу места работы ***Фон*** выходной буфер IECBufOut.
2. Установим **tid = M_EP_TE_1**, адрес ASDU **adr = 3**, адрес объекта информации **ioa = 5**.
3. Тип данных **tp** установим равным **uLX**, т.к. входные данные для данного типа ASDU структурированы.

   Условия передачи данных **trt** установим равным **0x01**: **бит 0** – спорадическая передача (причина передачи **3**) (см. Приложение Б).
4. Для формирования структуры заведем 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*** заведем выход блока ***ToReg8*** из библиотеки ***paCore***.

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

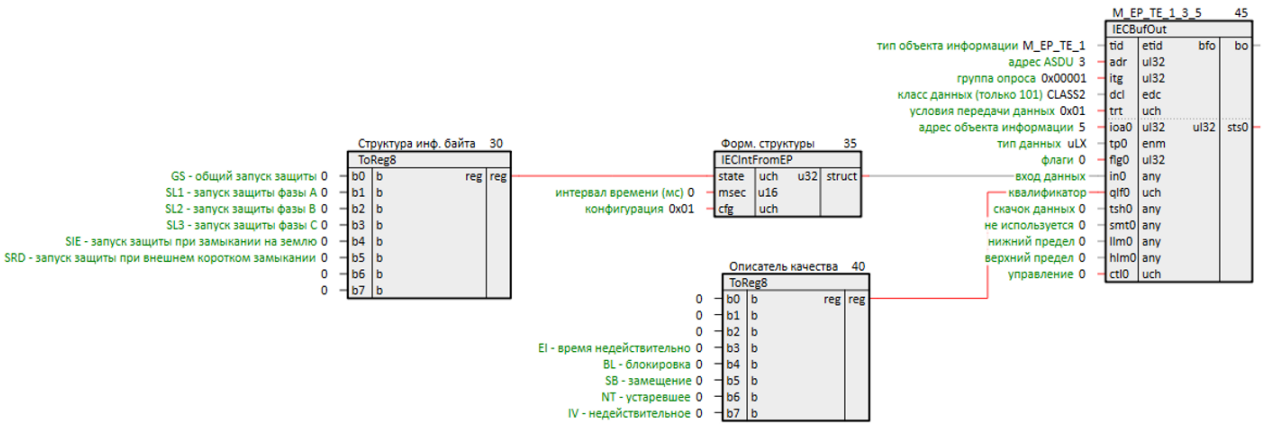


Рисунок 3.31 – Настройка IECBufOut. M_EP_TE_1

6. Добавим коннектор для подключения выходного буфера ***IECBufOut*** у блока ***IEC104Server***.
7. Подключим ***IECBufOut*** к коннектору блока ***IEC104Server***.

   В качестве клиента настроим 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 IECEPFromInt.

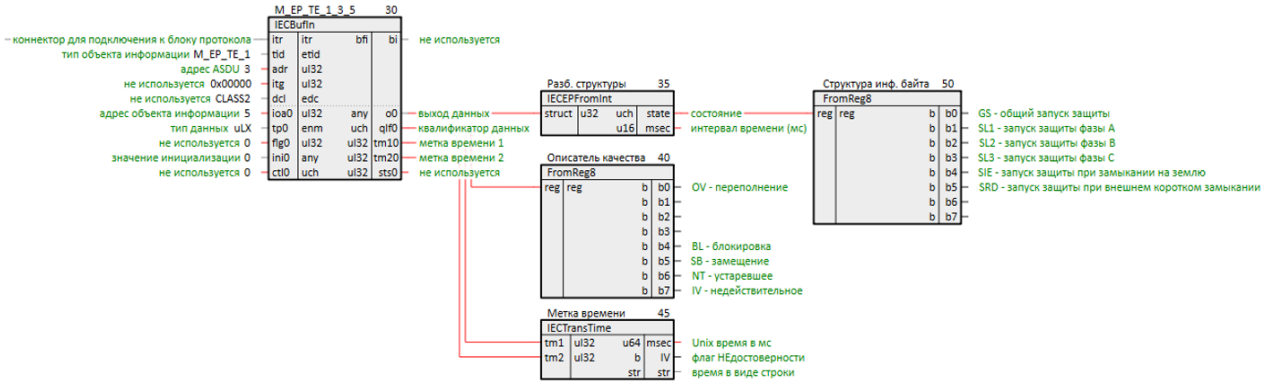


Рисунок 3.32 – Настройки клиента ПА

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

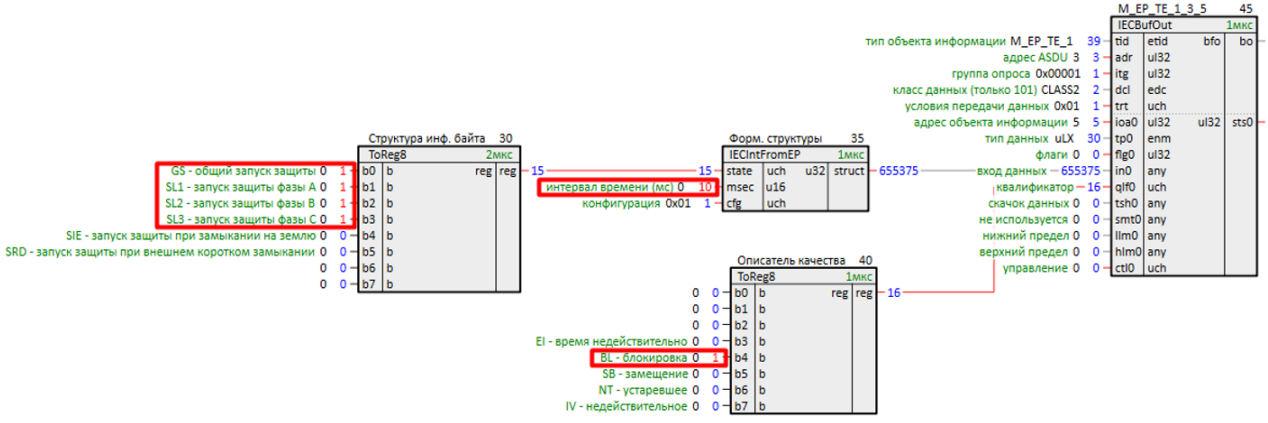


Рисунок 3.33 – Корректный обмен. Сервер

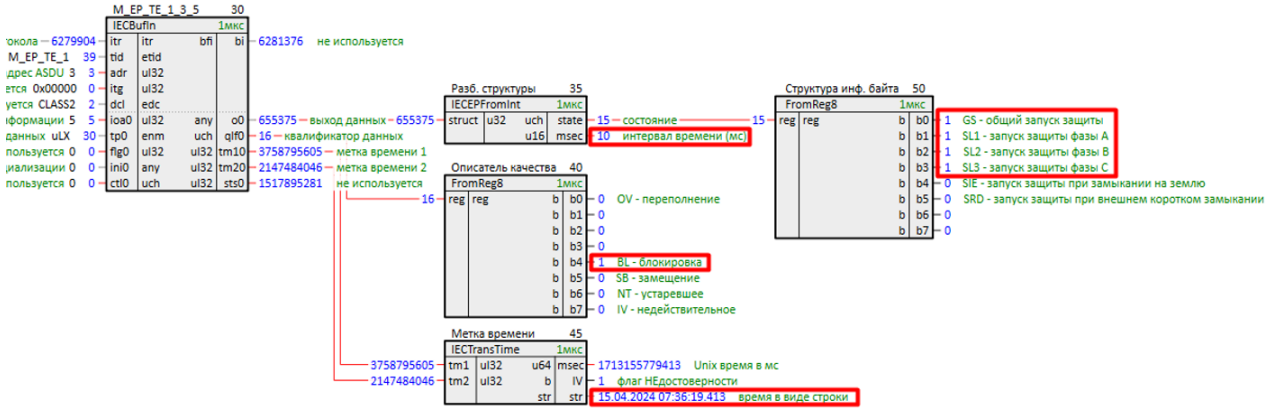


Рисунок 3.34 – Корректный обмен. Клиент

3.1.4 Передача интегральных сумм (M_IT_*)

Рассмотрим передачу 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).

1. Добавим на любую страницу места работы ***Фон*** выходной буфер IECBufOut.
2. Установим **tid = M_IT_TB_1**, адрес ASDU **adr = 3**, адрес объекта информации **ioa = 6**.
3. Тип данных **tp** установим равным **LX**.

   Условия передачи данных **trt** установим равным **0x01**: **бит 0** – спорадическая передача (причина передачи **3**) (см. Приложение Б).
4. Для формирования структуры заведем на вход **qlf** выход **out** блока IECITSQOut.

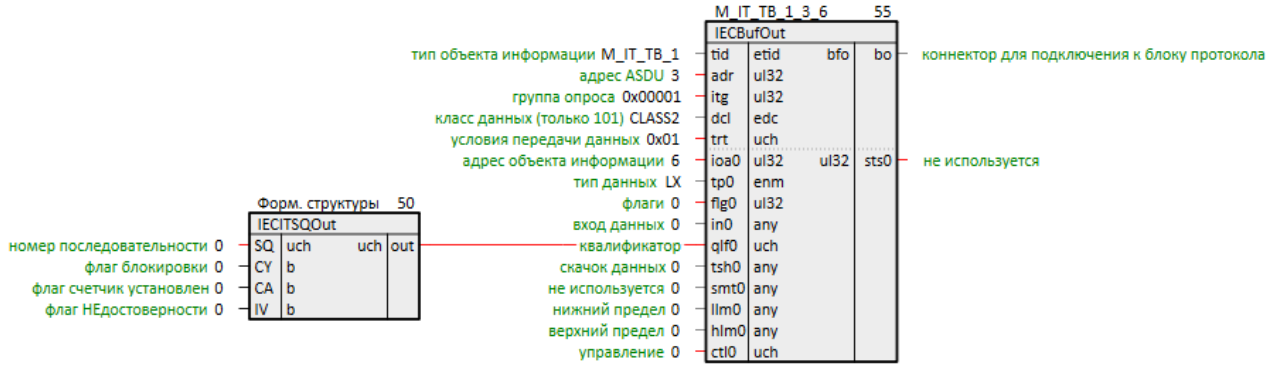


Рисунок 3.35 – Настройка IECBufOut. M_IT_TB_1

5. Добавим коннектор для подключения выходного буфера ***IECBufOut*** у блока ***IEC104Server***.
6. Подключим ***IECBufOut*** к коннектору блока ***IEC104Server***.
7. В качестве клиента настроим 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.

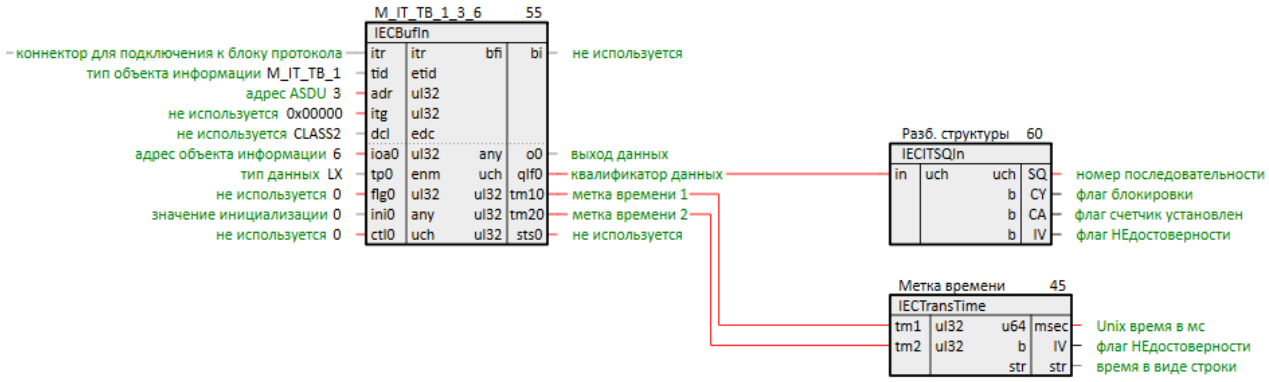


Рисунок 3.36 – Настройки клиента ПА

9. Запустим программы на ПЛК и виртуальном контроллере. Пронаблюдаем корректный обмен.

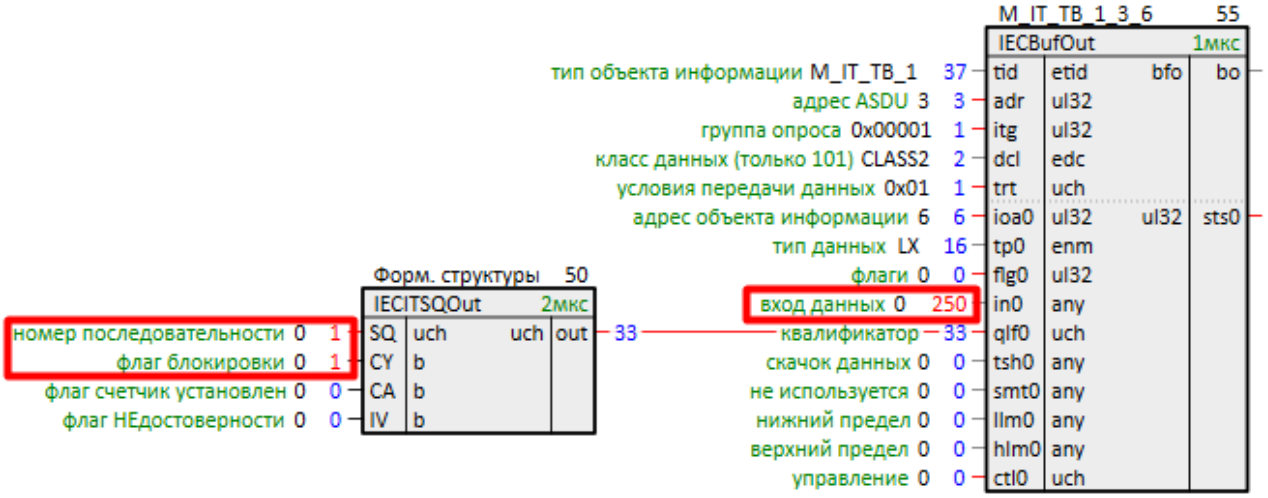


Рисунок 3.37 – Корректный обмен. Сервер

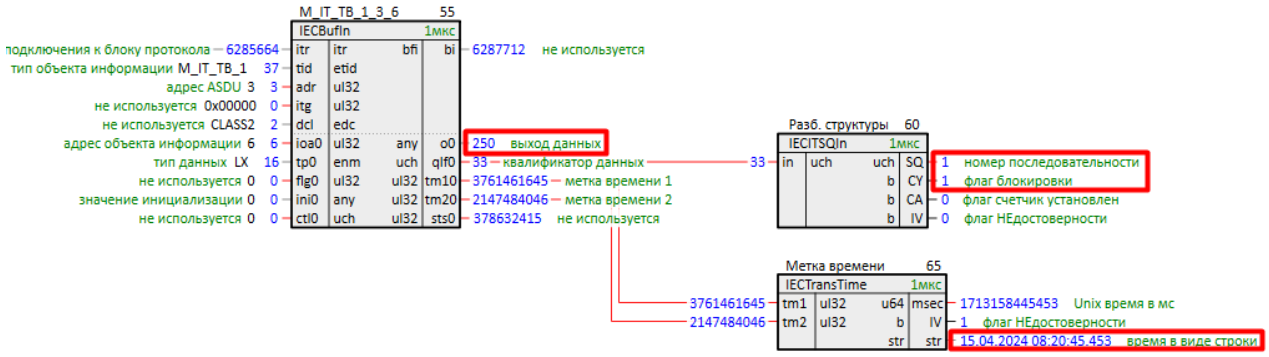


Рисунок 3.38 – Корректный обмен. Клиент

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**.

1. Добавим на любую страницу места работы ***Фон*** блок IEC104uni.
2. Соединим его с блоком ***TcpIpCIA*** из библиотеки ***paCore***.
3. На вход **Iip** подадим 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** подадим IP адрес контролируемого 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** зададим **IEC_MASTER**.

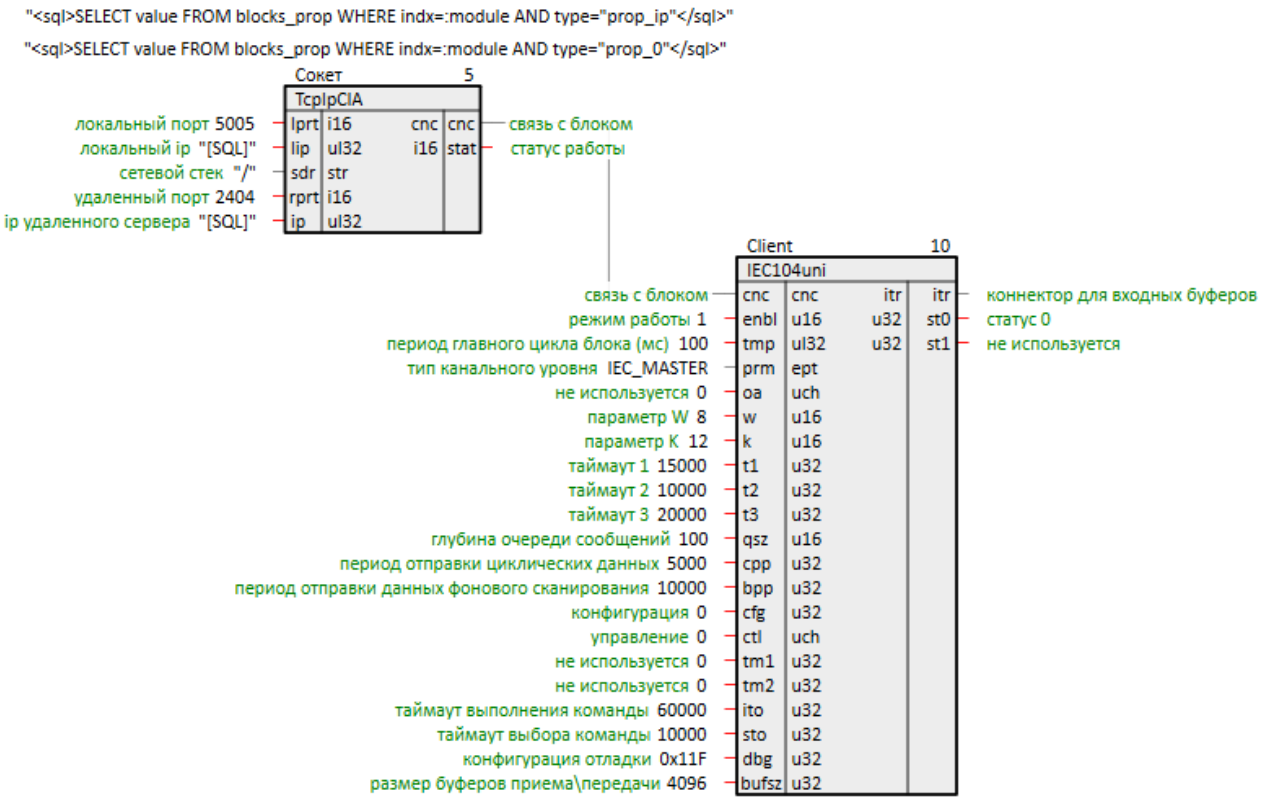


Рисунок 3.39 – Настройка IEC104uni

3.2.1 Прием информации о процессе в направлении контроля (М)

Рассмотрим обмен данными в направлении контроля (метка **М** – см. Приложение А).

Рассмотрим прием объекта информации с типом **M_SP_NA_1** – одноэлементный объект информации с описателем качества.

1. Добавим 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. Установим **tid = M_SP_NA_1**, адрес ASDU **adr = 3**, адрес объекта информации **ioa = 1**.

   Тип данных **tp** установим равным **DO**, так как у **M_SP_NA_1** в младшем разряде передается **1 бит** ТС.

3. Для расшифровки описателя качества с выхода **qlf** подсоединим его к входу блока ***FromReg8*** из библиотеки ***paCore***.

Рисунок 3.40 – Настройка IECBufIn. M_SP_NA_1

4. Подключим ***IECBufIn*** к коннектору блока ***IEC104uni***.

В качестве сервера будем 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 АРМ в MasterSCADA 4D:

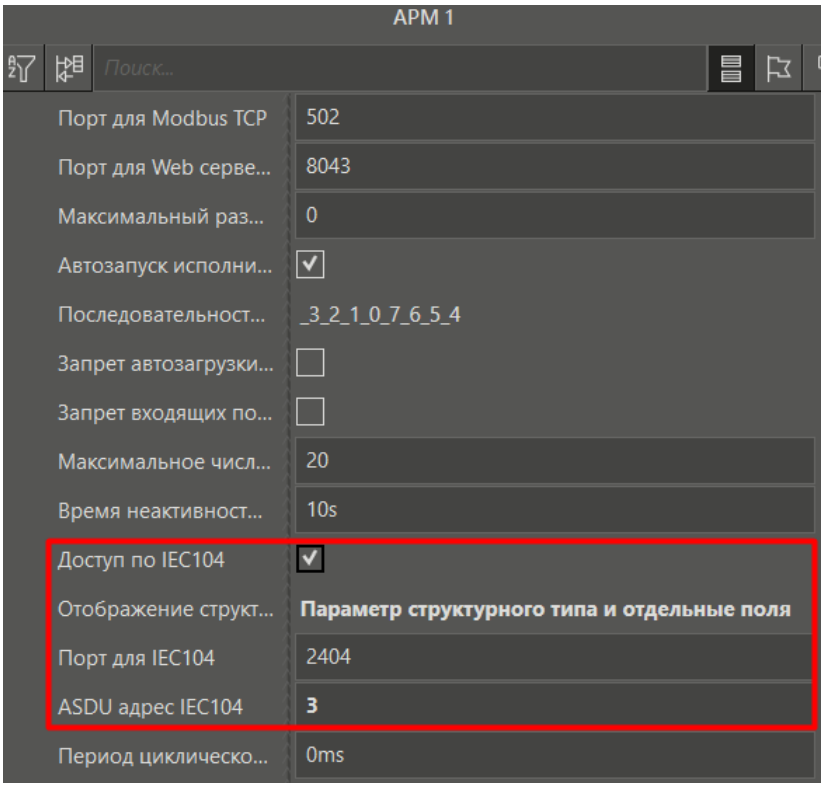


Рисунок 3.41 – Настройки сервера МЭК 104 АРМ

6. Добавим ***Канал*** во ***Внешние каналы*** и привяжем к нему параметр.

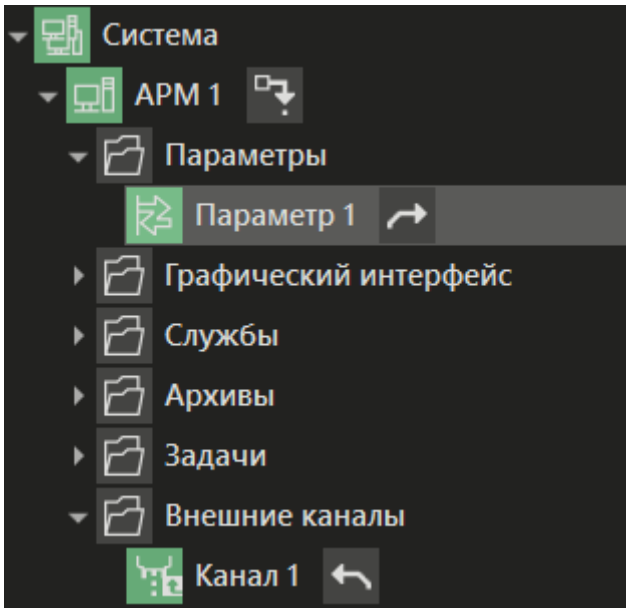


Рисунок 3.42 – Настройки канала АРМ

7. Настройки канала АРМ:

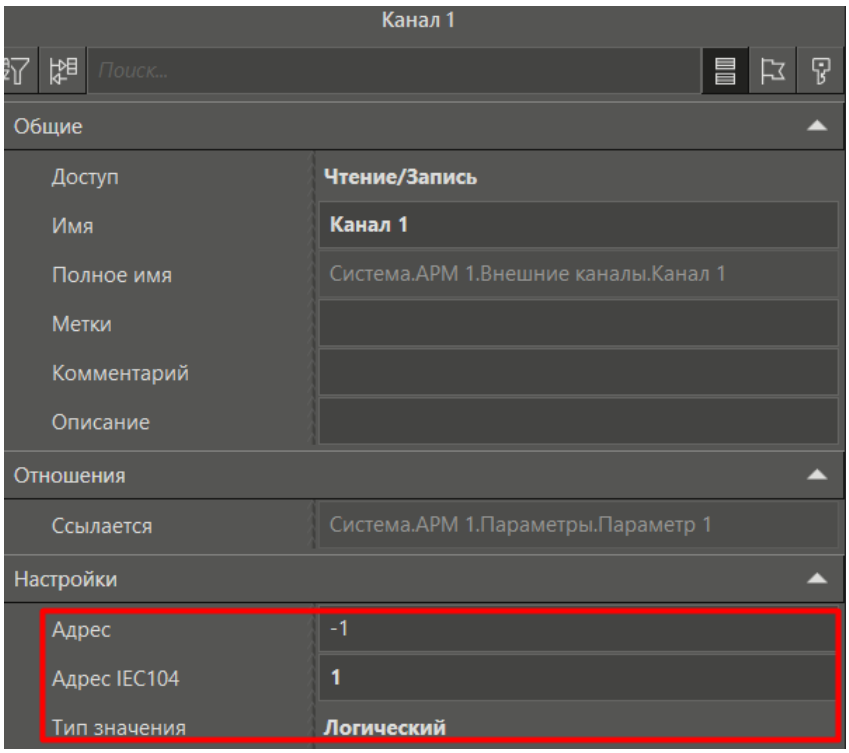


Рисунок 3.43 – Настройки канала АРМ

8. Запустим программы на ПЛК и АРМ. Пронаблюдаем корректный обмен.

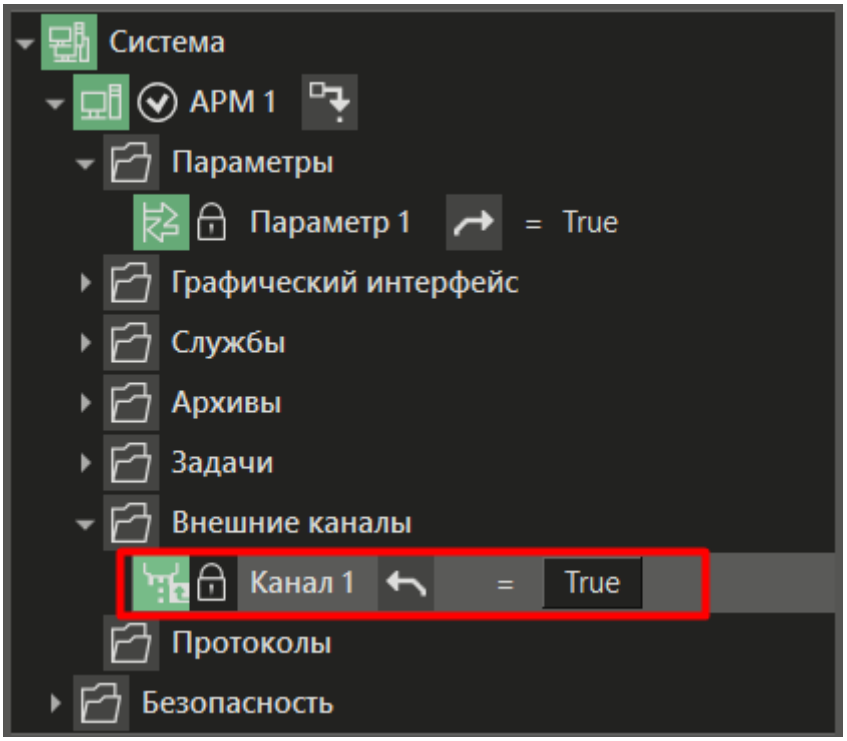


Рисунок 3.44 – Корректный обмен. Сервер

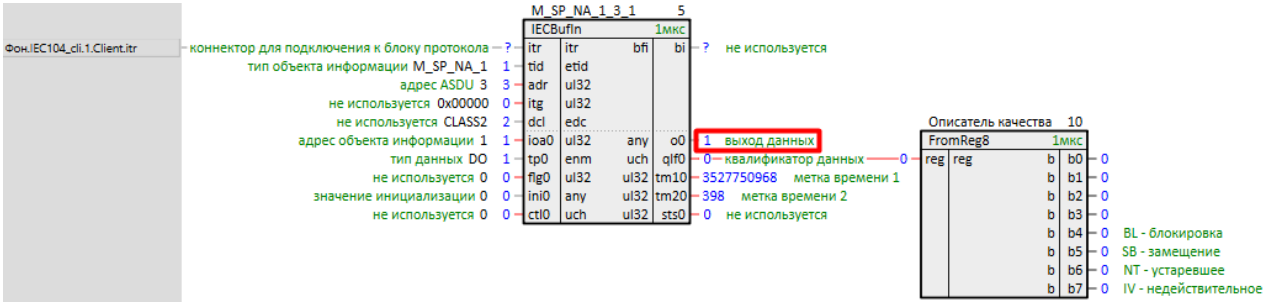


Рисунок 3.45 – Корректный обмен. Клиент

Рассмотрим прием объекта информации с типом **M_ME_TF_1** – ТИ с описателем качества и меткой времени.

1. Добавим выходной буфер ***IECBufIn***.
2. Установим **tid = M_ME_TF_1**, адрес ASDU **adr = 3**, адрес объекта информации **ioa = 2**.

   Тип данных **tp** установим равным **AO**, так как **M_ME_TF_1** – величина с плавающей запятой.
3. Для расшифровки описателя качества с выхода **qlf** подсоединим его к входу 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.

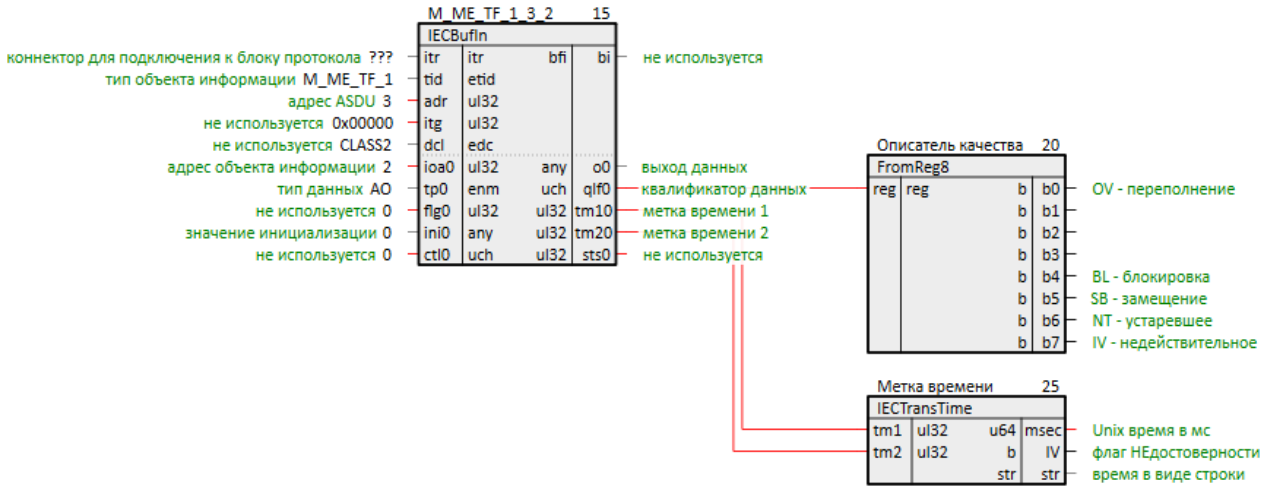


Рисунок 3.46 – Настройка IECBufIn. M_ME_TF_1

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

   Настройка канала АРМ в MasterSCADA 4D:

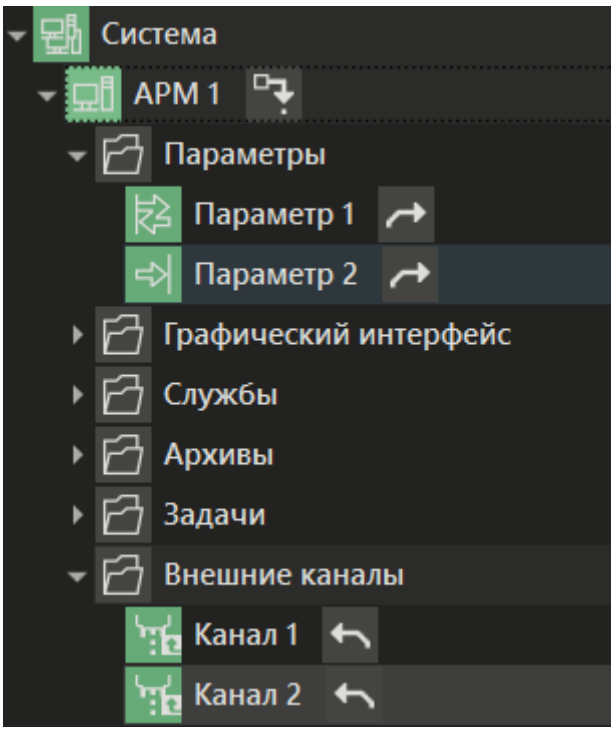


Рисунок 3.47 – Настройки канала АРМ

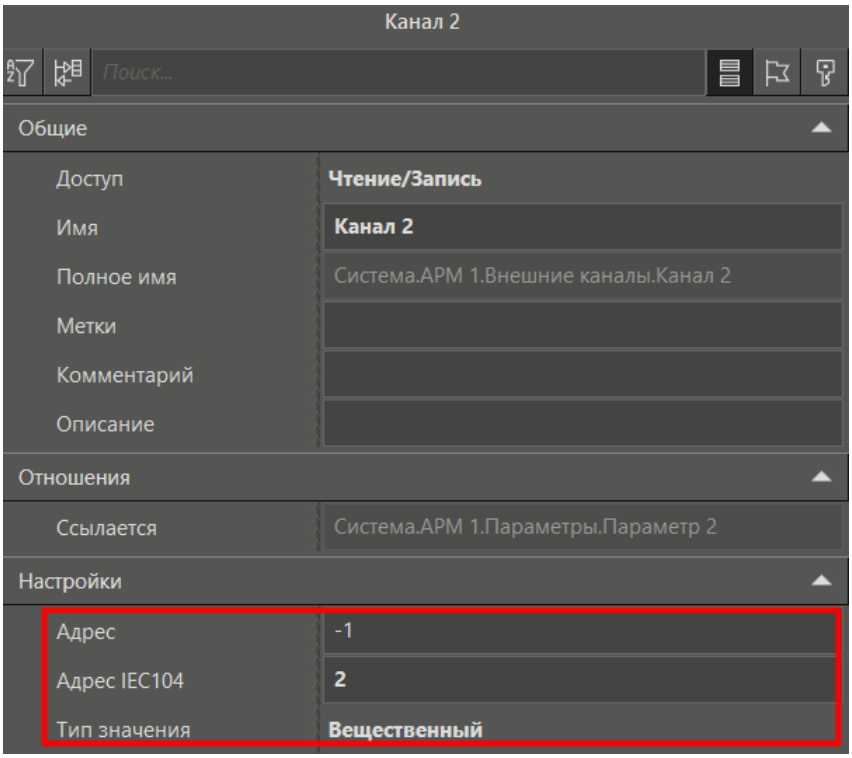


Рисунок 3.48 – Настройки канала АРМ

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

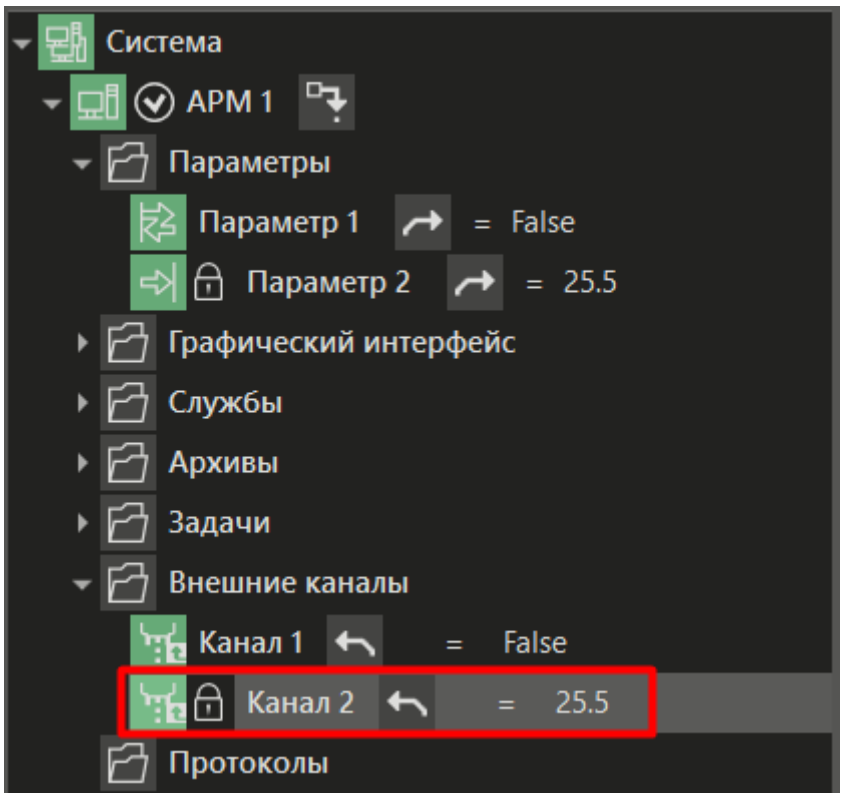


Рисунок 3.49 – Корректный обмен. Сервер

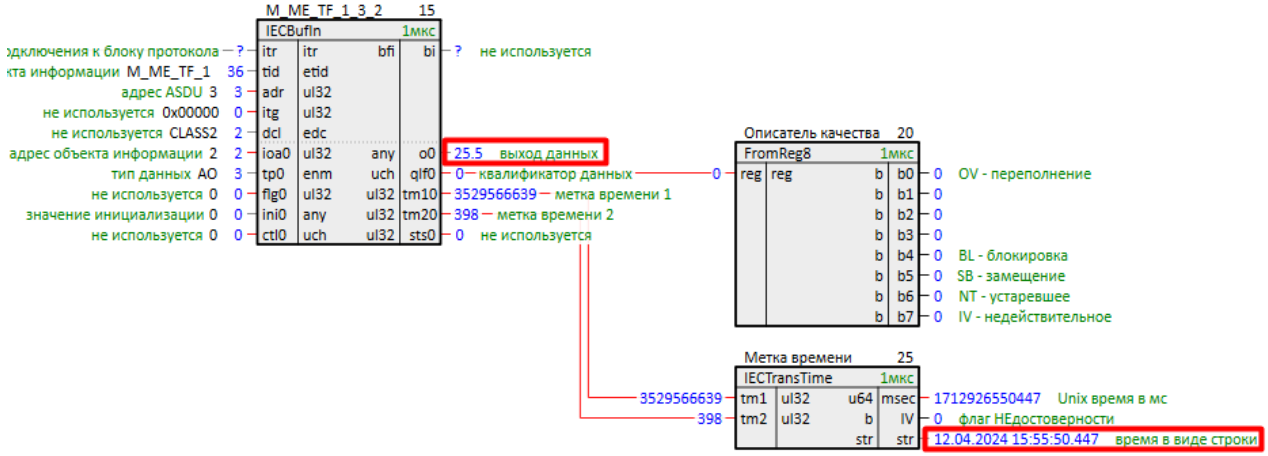


Рисунок 3.50 – Корректный обмен. Клиент

3.2.2 Передача информации о процессе в направлении управления (C)

Рассмотрим обмен данными в направлении управления (метка **C** – см. Приложение А).

Рассмотрим отправку объекта информации с типом **C_SC_NA_1** – однопозиционная команда ТУ.

1. Добавим на любую страницу места работы ***Фон*** выходной буфер IECBufOut.
2. Установим **tid = C_SC_NA_1**, адрес ASDU **adr = 3**, адрес объекта информации **ioa = 3**.

   Тип данных **tp** установим равным **DI**, так как **C_SC_NA_1** передает значение «включить/выключить» в младшем бите.
3. Условия передачи данных **trt** установим равным **0x08**: **бит 3** – команда (причины передачи **6**, **8**, **10**) (см. Приложение Б).

   Передача команды активируется **битом 0** входа **ctl** – заведем на него выход блока ***ToReg8*** из библиотеки ***paCore*** для удобства.

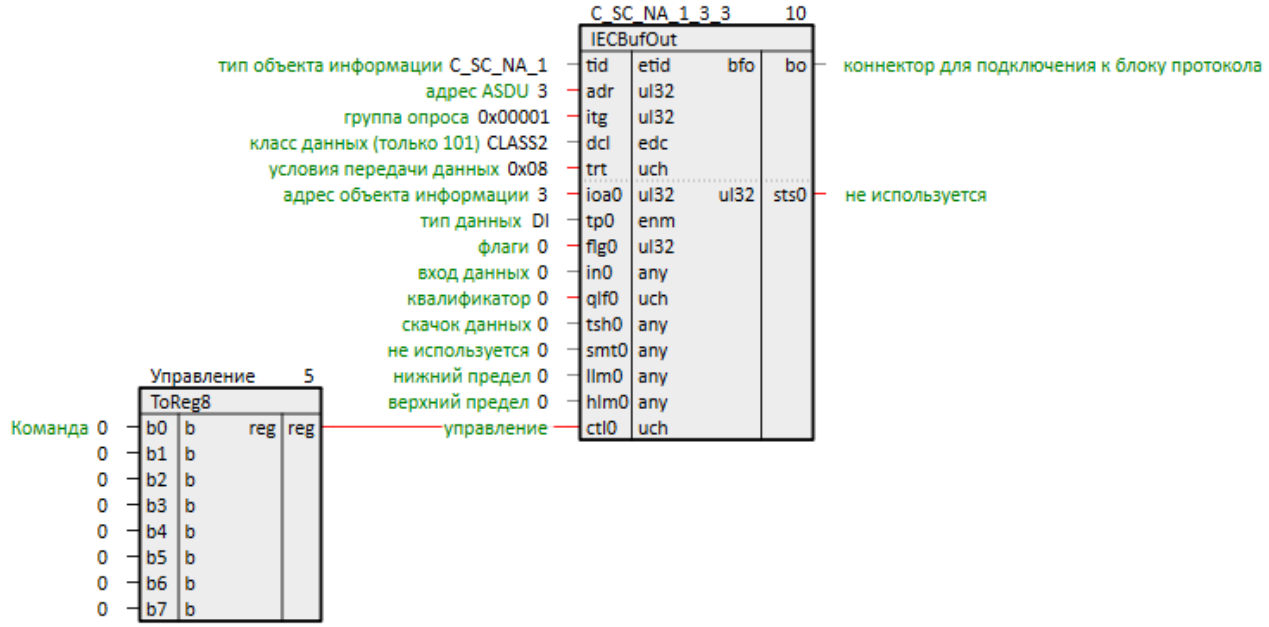


Рисунок 3.51 – Настройка IECBufOut. C_SC_NA_1

4. Добавим коннектор для подключения выходного буфера ***IECBufOut*** у блока ***IEC104uni***.
5. Подключим ***IECBufOut*** к коннектору блока ***IEC104uni***.

   В качестве сервера будем 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:

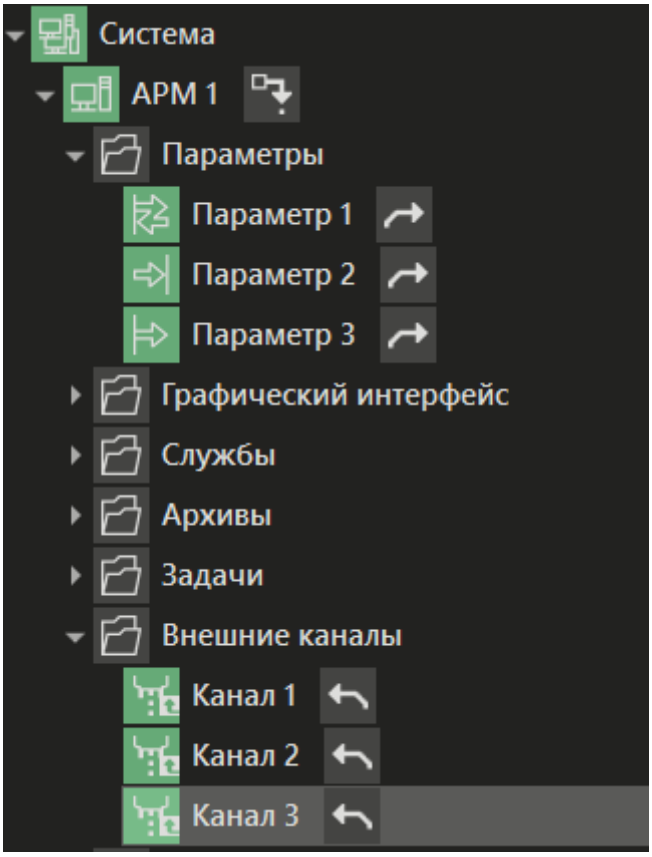


Рисунок 3.52 – Настройки канала АРМ

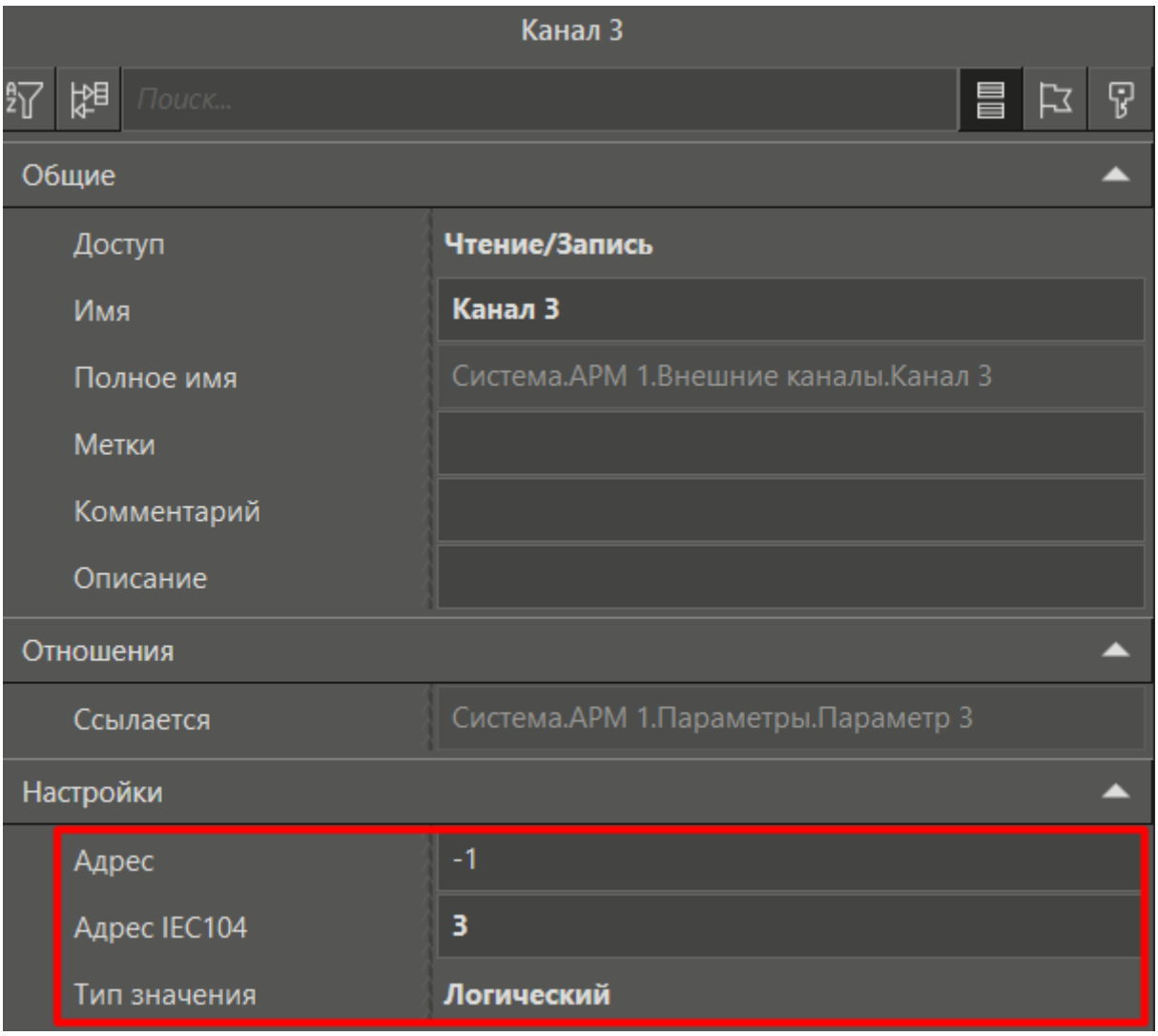


Рисунок 3.53 – Настройки канала АРМ

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

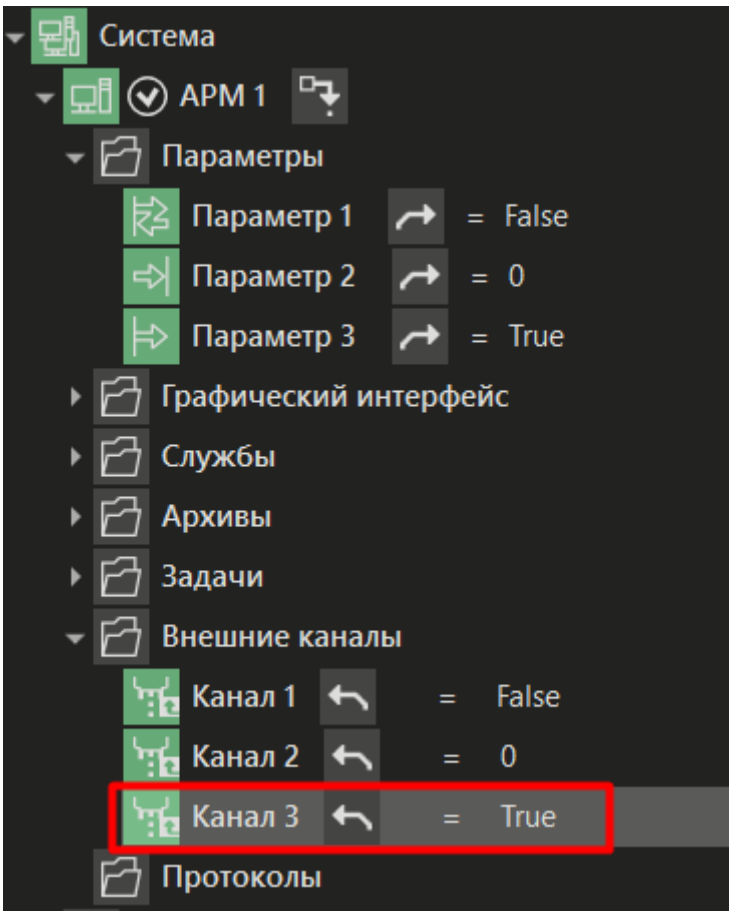


Рисунок 3.54 – Корректный обмен. Сервер

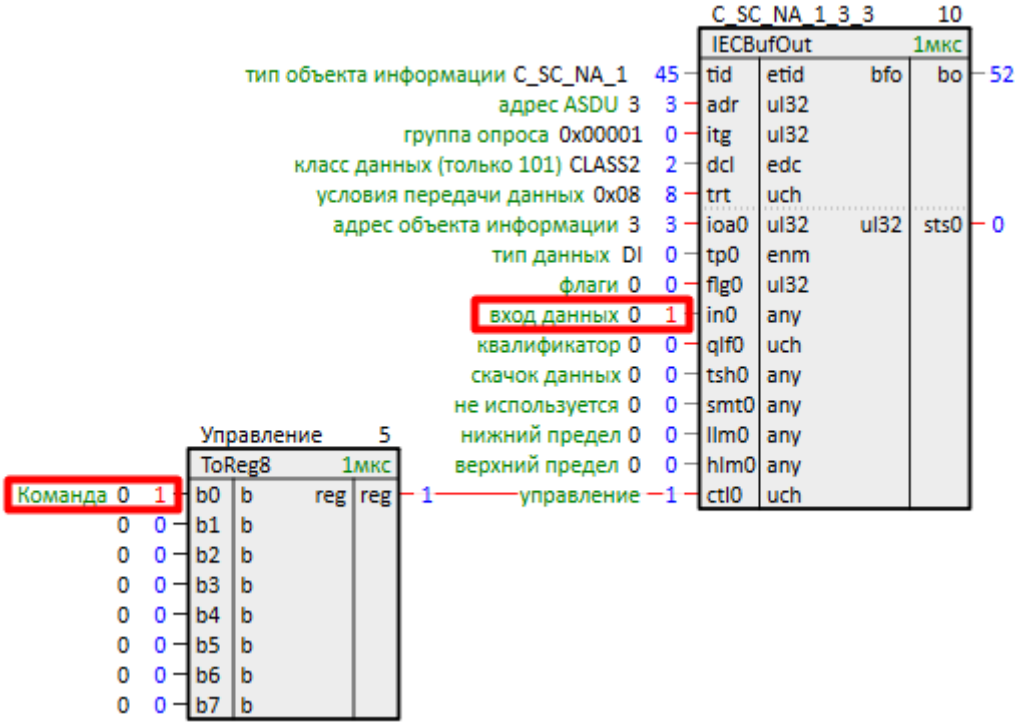


Рисунок 3.55 – Корректный обмен. Клиент

Рассмотрим отправку объекта информации с типом **C_SE_TC_1** – команда уставки с меткой времени.

1. Добавим выходной буфер ***IECBufOut***.
2. Установим **tid = C_SE_TC_1**, адрес ASDU **adr = 3**, адрес объекта информации **ioa = 4**.

   Тип данных **tp** установим равным **AI**, так как **C_SE_TC_1** имеет формат с плавающей запятой.

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

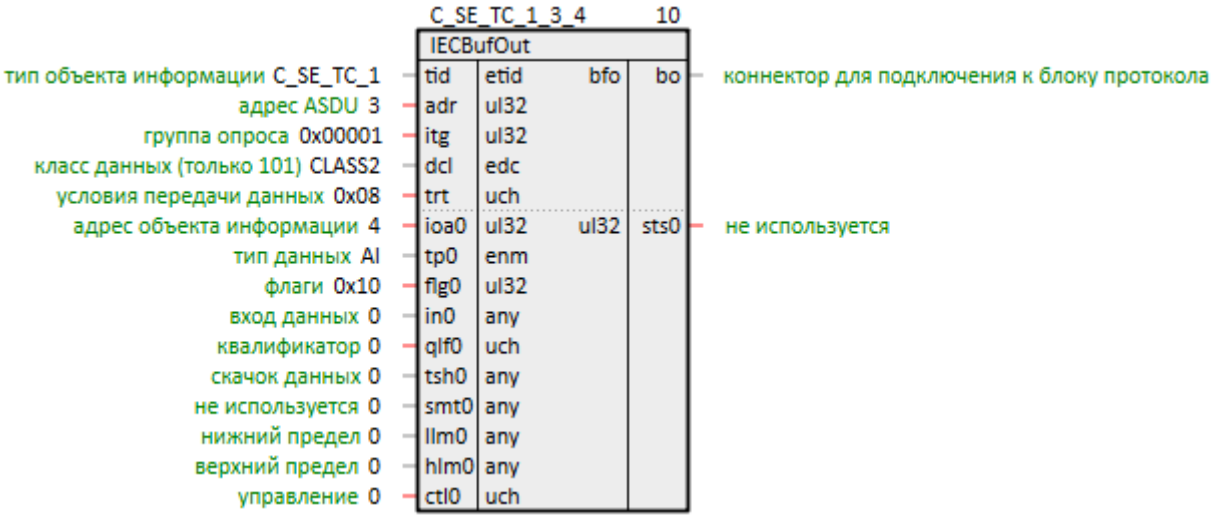


Рисунок 3.56 – Настройка IECBufIn. C_SE_TC_1

4. Добавим коннектор для подключения выходного буфера ***IECBufOut*** у блока ***IEC104uni***.
5. Подключим ***IECBufOut*** к коннектору блока ***IEC104uni***.

   Настройка канала АРМ в MasterSCADA 4D:

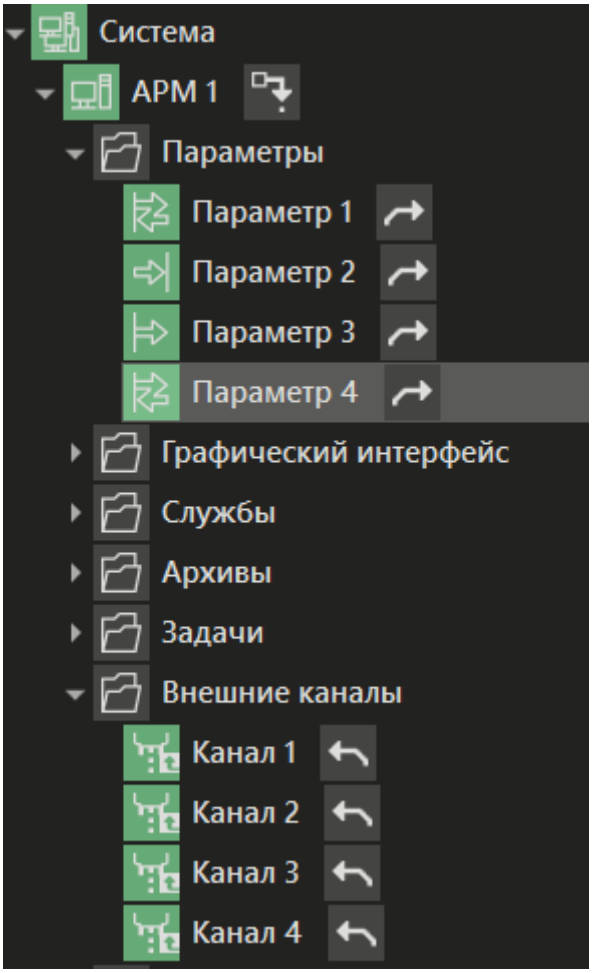


Рисунок 3.57 – Настройки канала АРМ

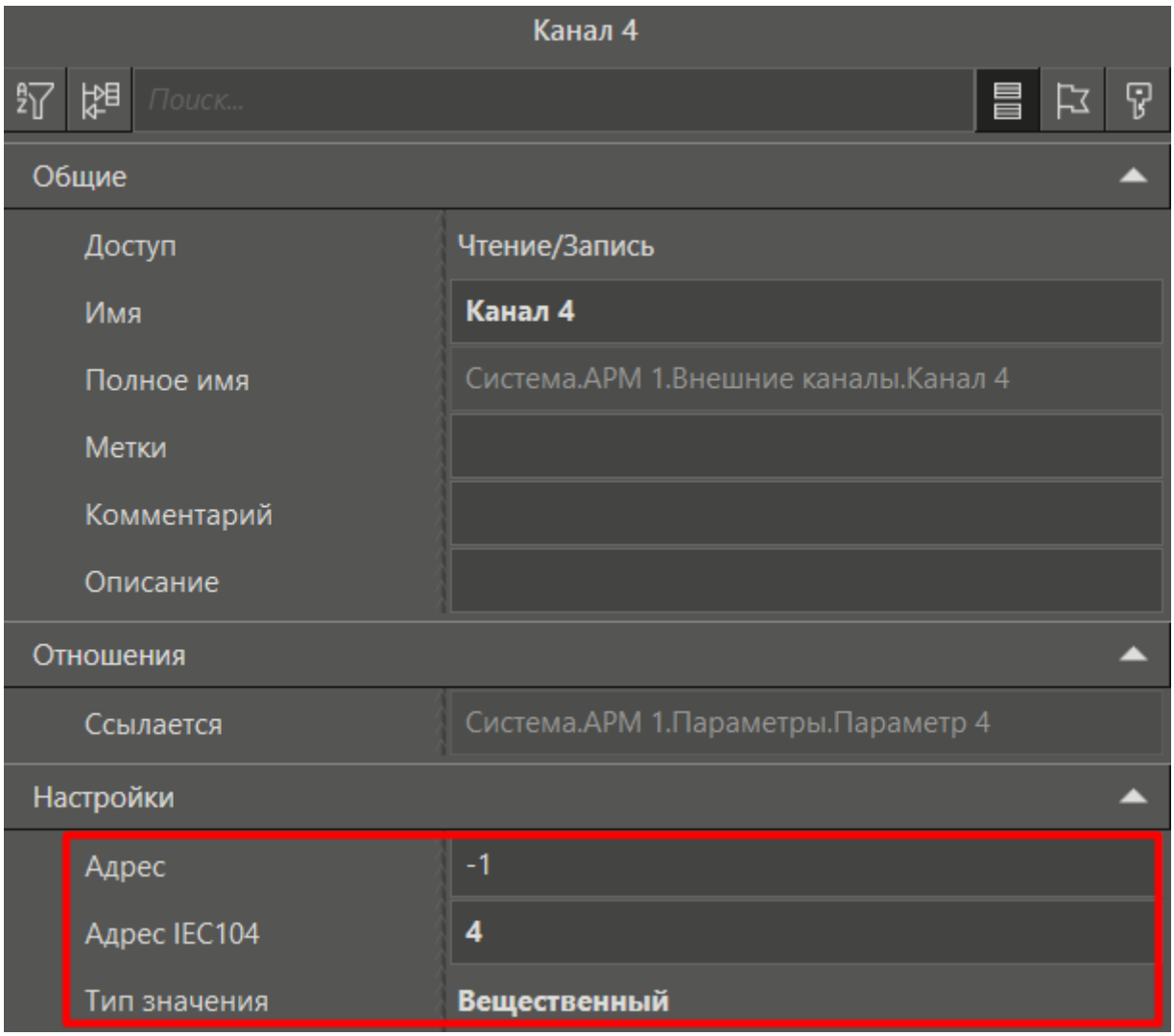


Рисунок 3.58 – Настройки канала АРМ

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

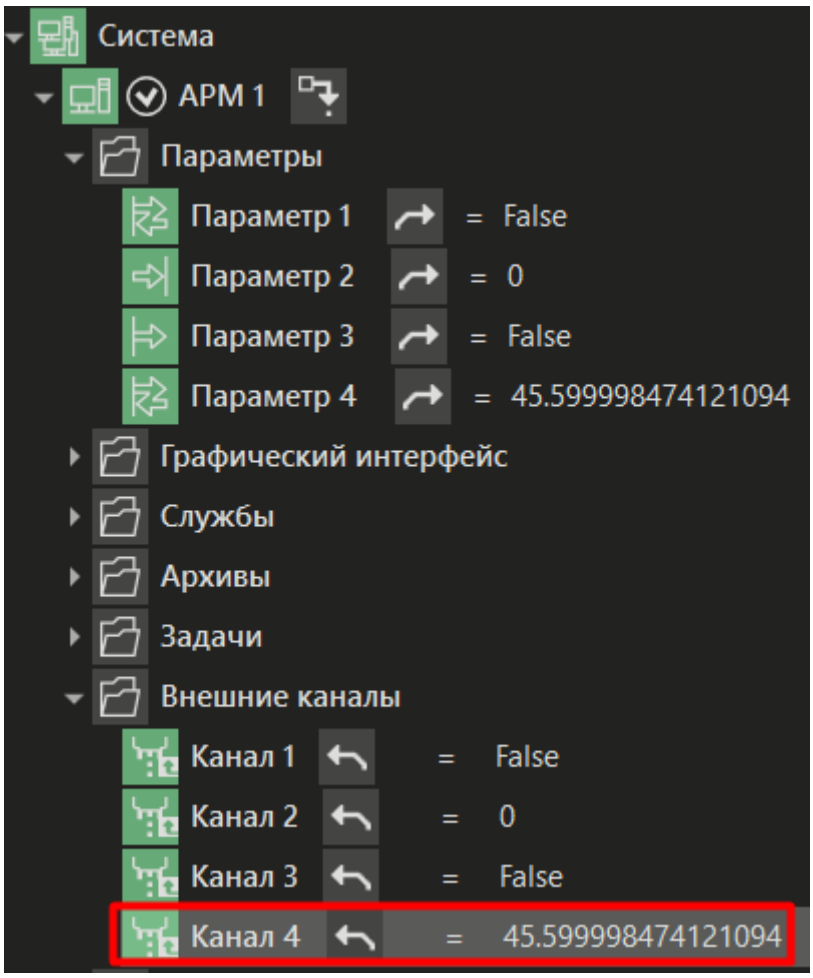


Рисунок 3.59 – Корректный обмен. Сервер

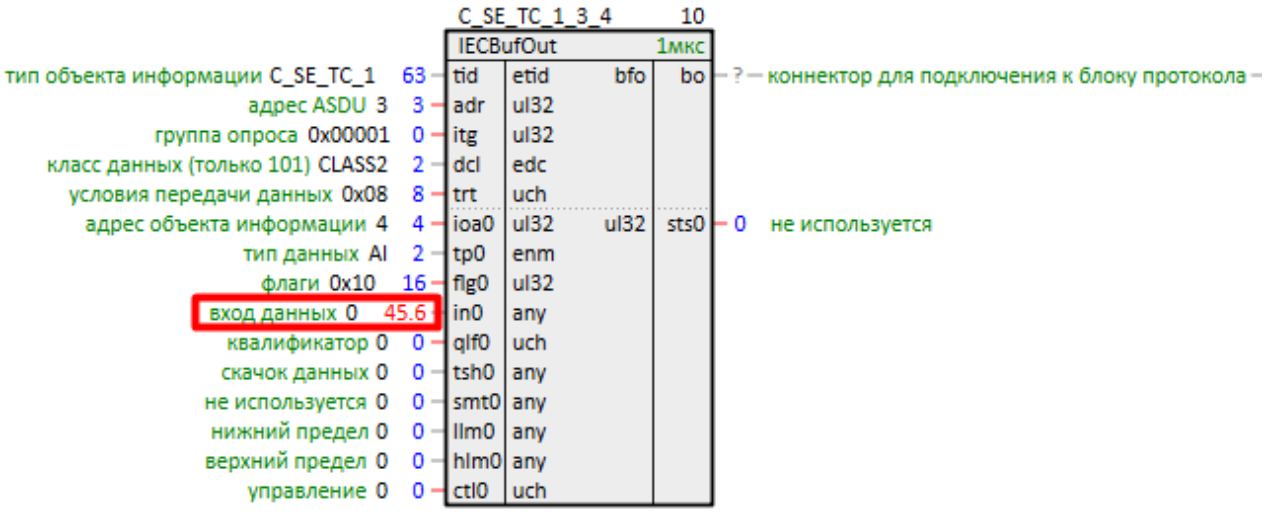


Рисунок 3.60 – Корректный обмен. Клиент

3.2.3 Прием упакованных данных (M_EP_*)

Рассмотрим прием объекта информации с типом **M_EP_TE_1** – упакованное сообщение с меткой времени **CP56Время2а**. Особенность данного типа состоит в том, что в сообщении передается структура, состоящая из битовой маски и интервала времени.

1. Добавим на любую страницу места работы ***Фон*** выходной буфер IECBufIn.
2. Установим **tid = M_EP_TE_1**, адрес ASDU **adr = 3**, адрес объекта информации **ioa = 5**.

   Тип данных **tp** установим равным **uLX**, т.к. входные данные для данного типа ASDU структурированы.

3. Для расшифровки структуры на выходе **o** используем блок IECEPFromInt. Для удобства подключим к нему блок ***ToReg8*** из библиотеки ***paCore***.
4. Для расшифровки описателя качества с выхода **qlf** подсоединим его также к входу блока ***FromReg8***. Здесь добавляется атрибут **EI** – действительность интервала длительности.
5. Для расшифровки метки времени соединим выходы **tm1** и **tm2** с входами блока IECTransTime.

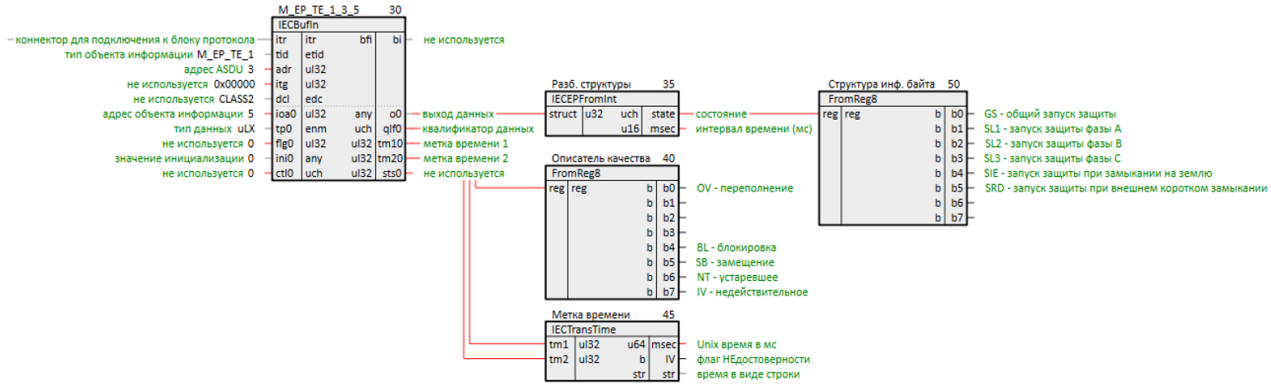


Рисунок 3.61 – Настройка IECBufIn. M_EP_TE_1

6. Подключим ***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.
7. Для формирования структуры используется блок IECIntFromEP.

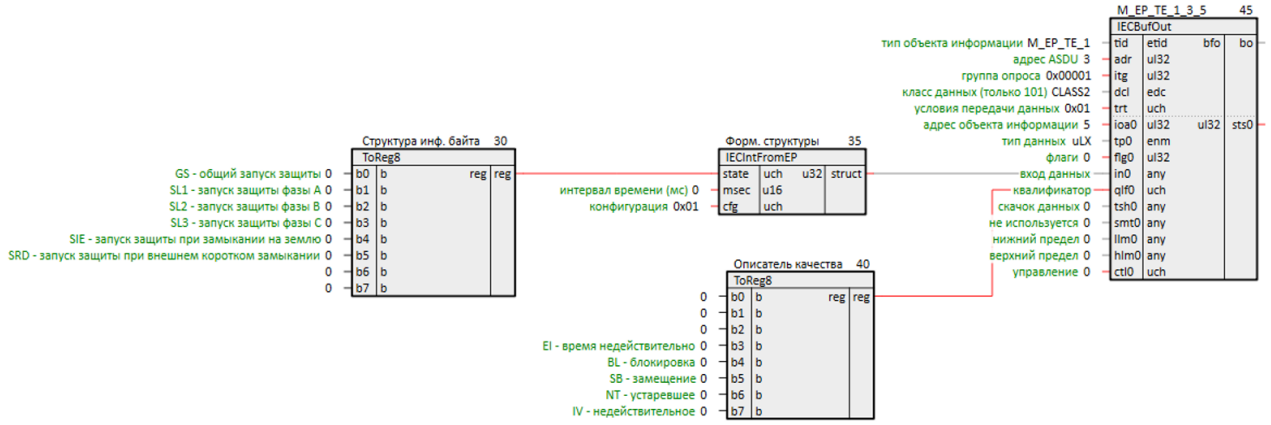


Рисунок 3.62 – Настройки сервера ПА

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

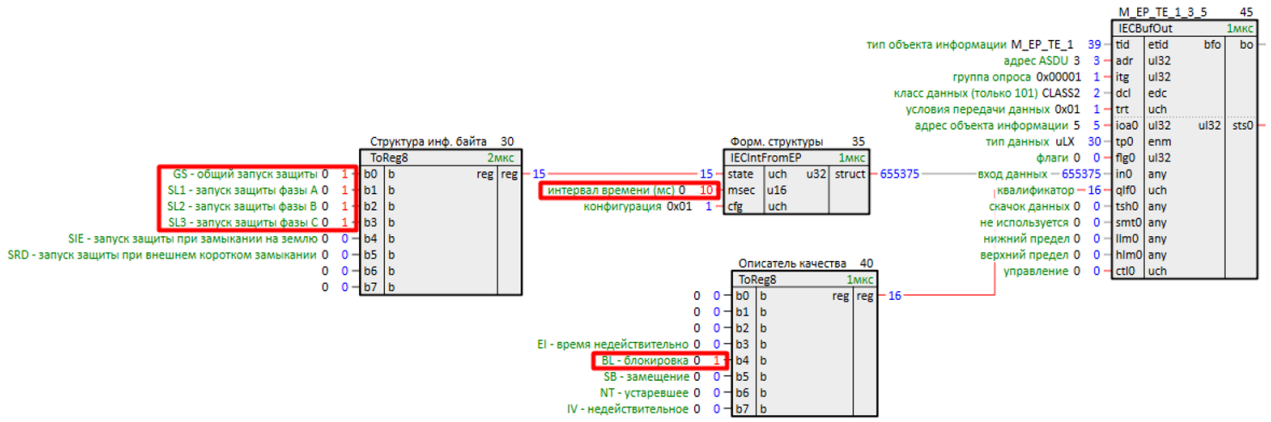


Рисунок 3.63 – Корректный обмен. Сервер

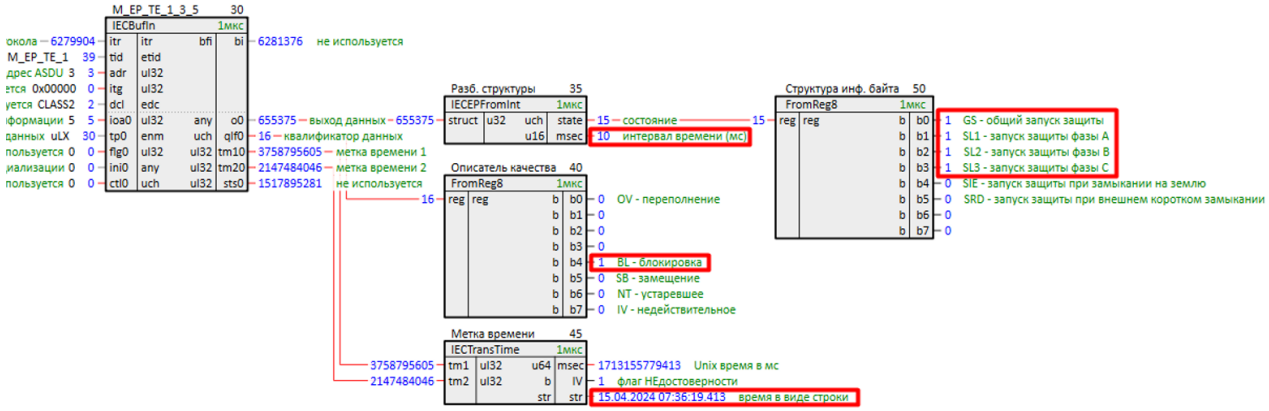


Рисунок 3.64 – Корректный обмен. Клиент

3.2.4 Прием интегральных сумм (M_IT_*)

Рассмотрим 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).

1. Добавим на любую страницу места работы ***Фон*** выходной буфер IECBufIn.
2. Установим **tid = M_IT_TB_1**, адрес ASDU **adr = 3**, адрес объекта информации **ioa = 6**.

   Тип данных **tp** установим равным **LX**.
3. Для расшифровки структуры на выходе **qlf** используем блок IECITSQIn.

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

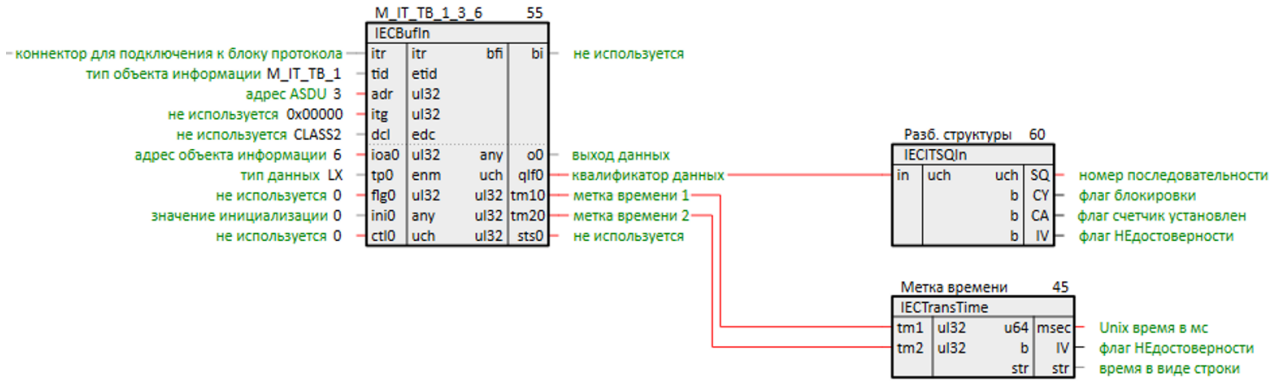


Рисунок 3.65 – Настройка IECBufIn. M_IT_TB_1

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

   В качестве сервера настроим 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 используется блок IECITSQOut.

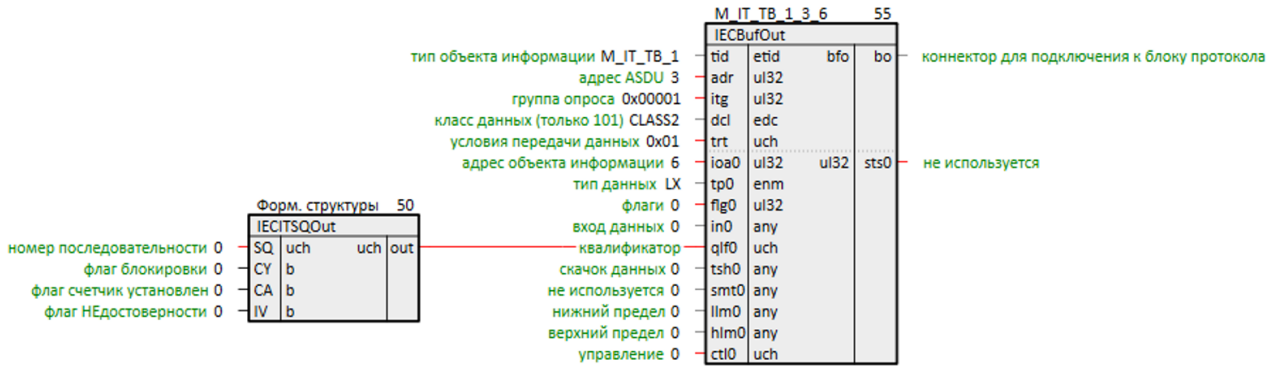


Рисунок 3.66 – Настройки сервера ПА

6. Запустим программы на ПЛК и виртуальном контроллере. Пронаблюдаем корректный обмен.

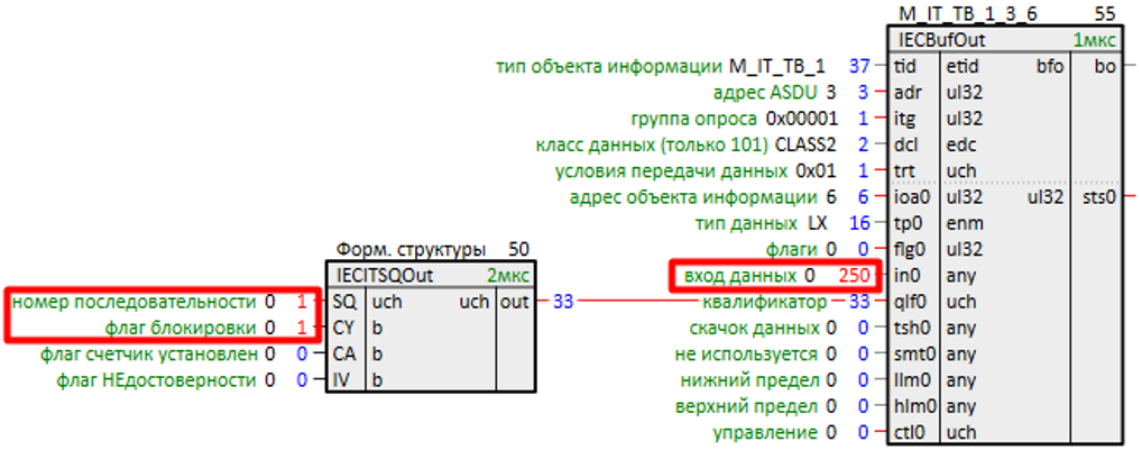


Рисунок 3.67 – Корректный обмен. Сервер

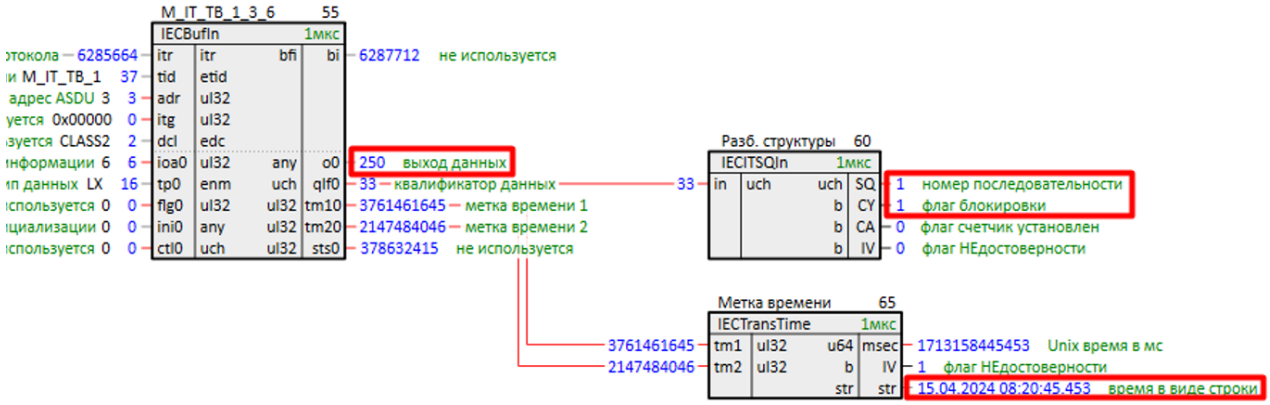


Рисунок 3.68 – Корректный обмен. Клиент

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

4.1 Настройка обмена в режиме сервера протокола 5-101

Настроим 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**.

1. Добавим на любую страницу места работы ***Фон*** блок IEC101uni.
2. На вход блока **cnc** заведем 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** зададим режим **IEC_SLAVE**.
4. Добавим коннектор для выходного буфера IECBufOut.

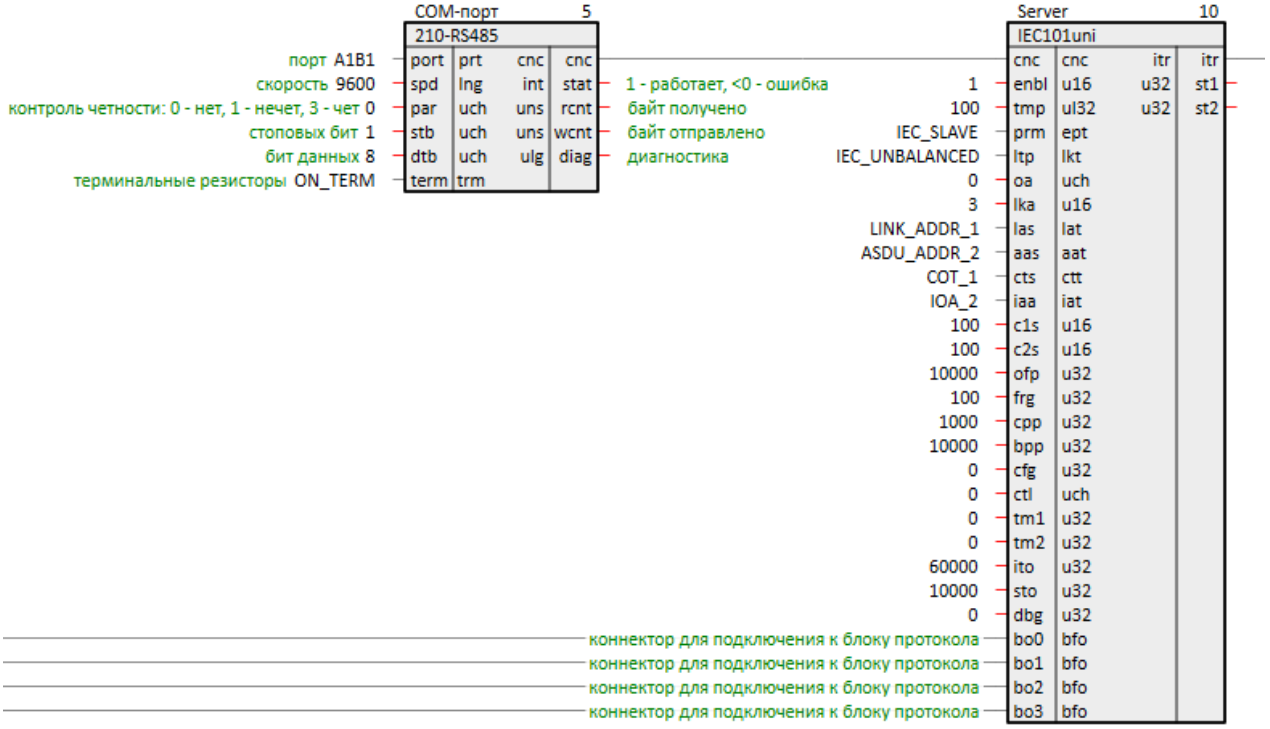


Рисунок 4.1 – Настройка IEC101uni в режиме IEC_SLAVE

ПРИМЕЧАНИЕ
Прием и передача ASDU для протокола **5-101** аналогична рассмотренным в разделах 3.1 и 3.2. Поддерживаемые типы ASDU приведены в Приложении А и в справке среды программирования.

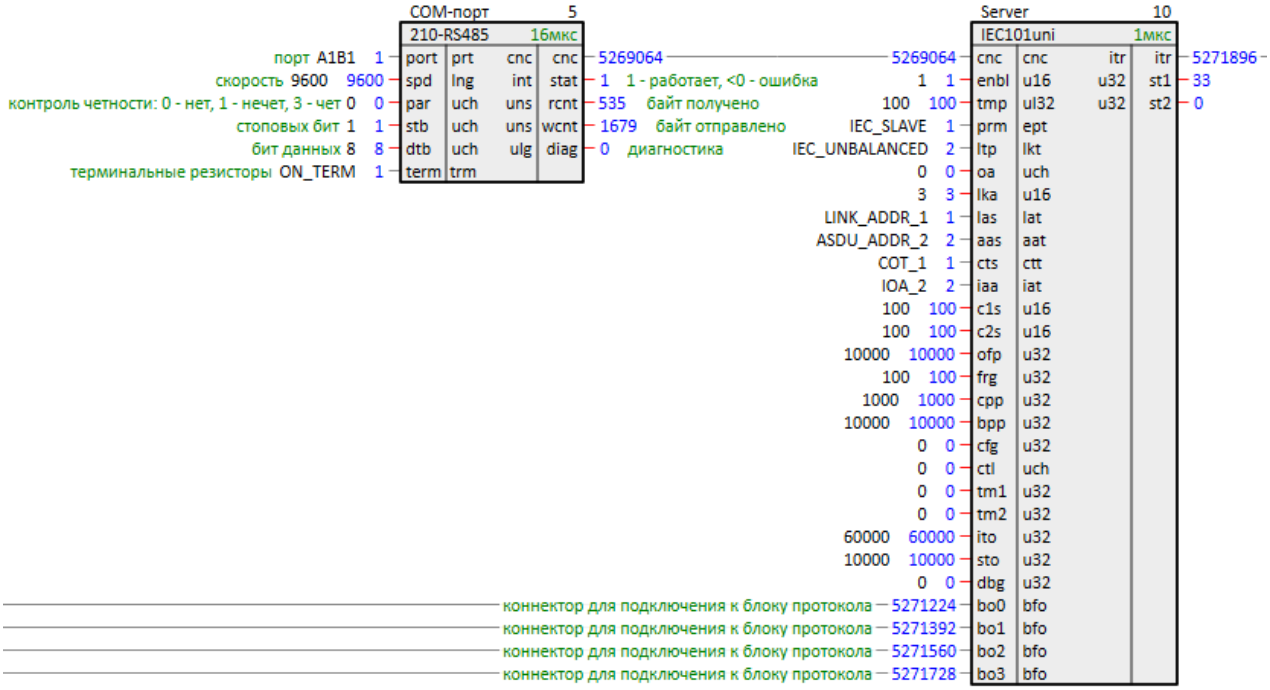


Рисунок 4.2 – Корректный обмен

4.2 Настройка обмена в режиме клиента протокола 5-101

Настроим обмен по протоколу **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**.

1. Добавим на любую страницу места работы ***Фон*** блок IEC101uni.
2. На вход блока **cnc** заведем 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** зададим режим **IEC_MASTER**.
4. Добавим коннектор для выходного буфера IECBufOut.

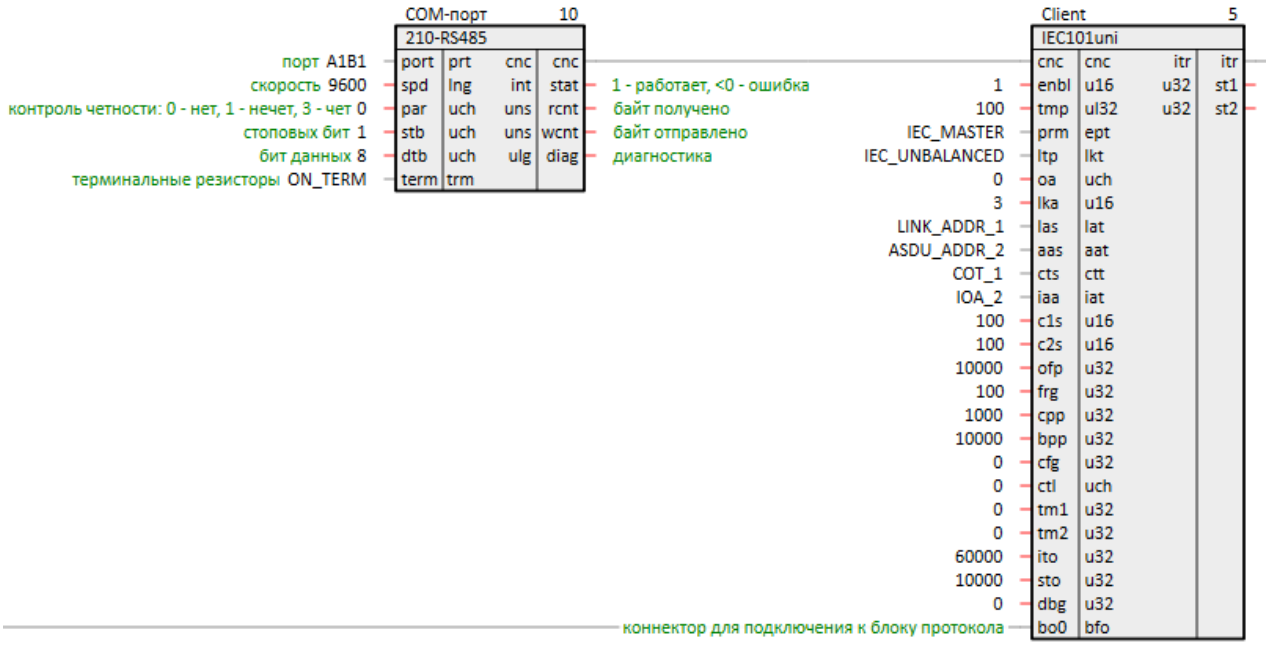


Рисунок 4.3 – Настройка IEC101uni в режиме IEC_MASTER

ПРИМЕЧАНИЕ
Прием и передача ASDU для протокола **5-101** аналогична рассмотренным в разделах 3.1 и 3.2. Поддерживаемые типы ASDU приведены в Приложении А и в справке среды программирования.

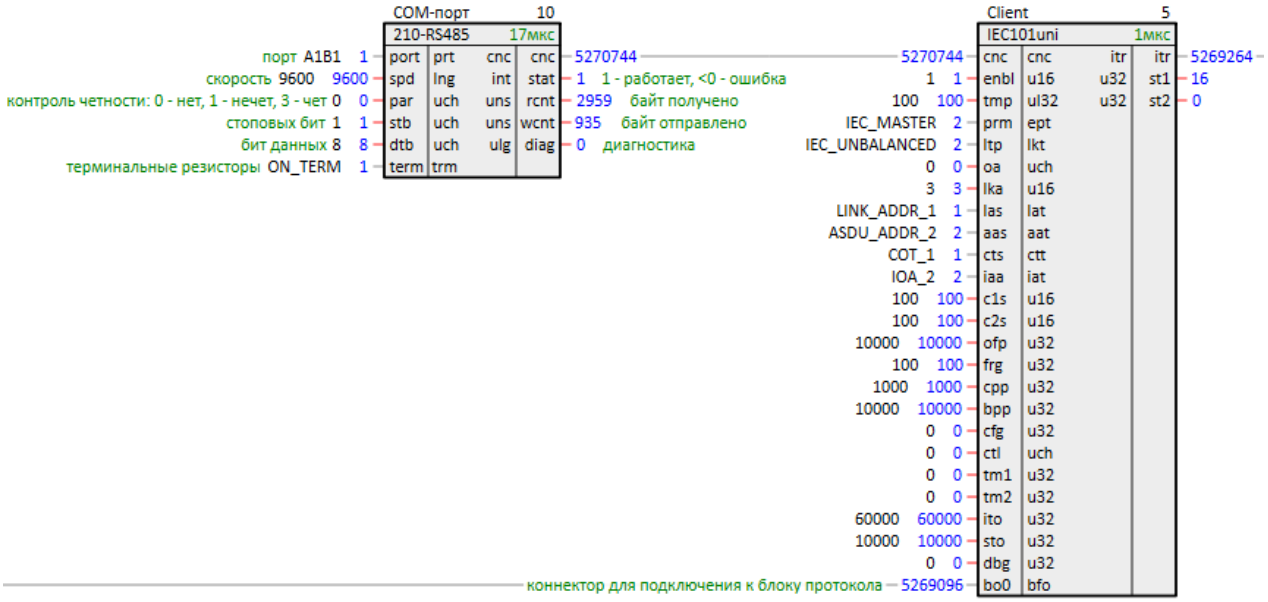


Рисунок 4.4 – Корректный обмен

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.

При установке **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**.

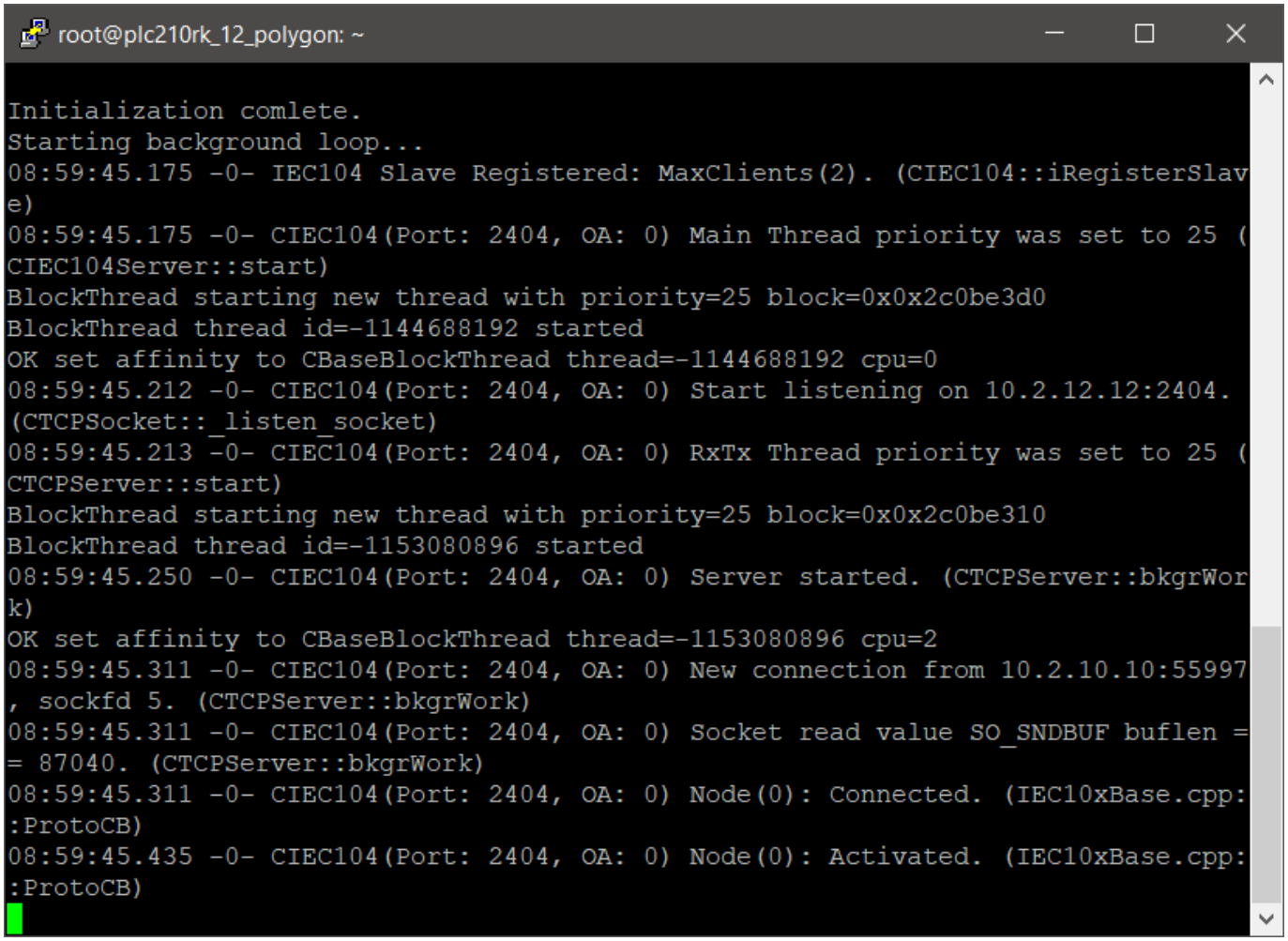


Рисунок 5.1 – Вывод сообщений при dbg = 0

Вывод сообщений осуществляется двумя различными способами. Способ устанавливается 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.

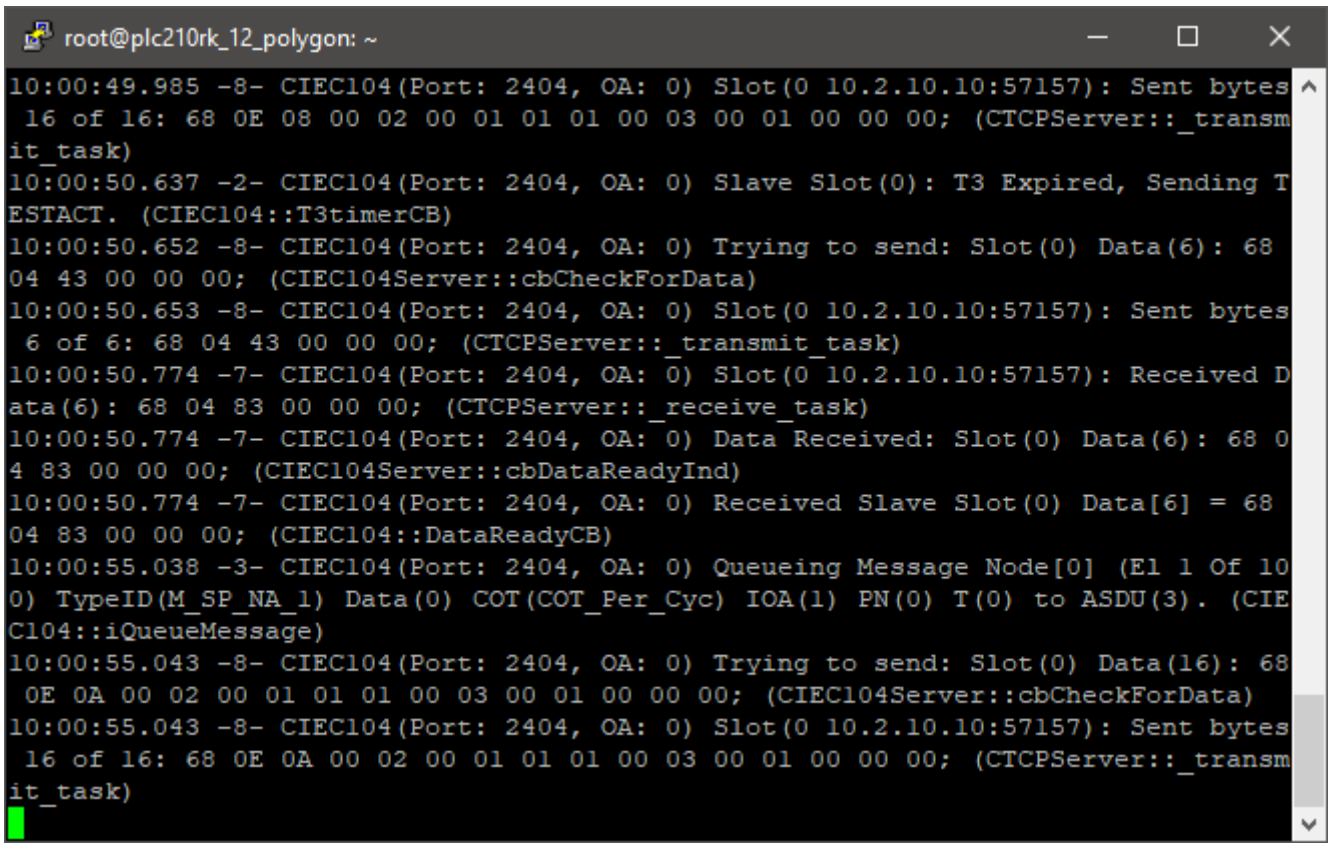


Рисунок 5.2 – Вывод сообщений в консоль при dbg = 0x1C7

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:

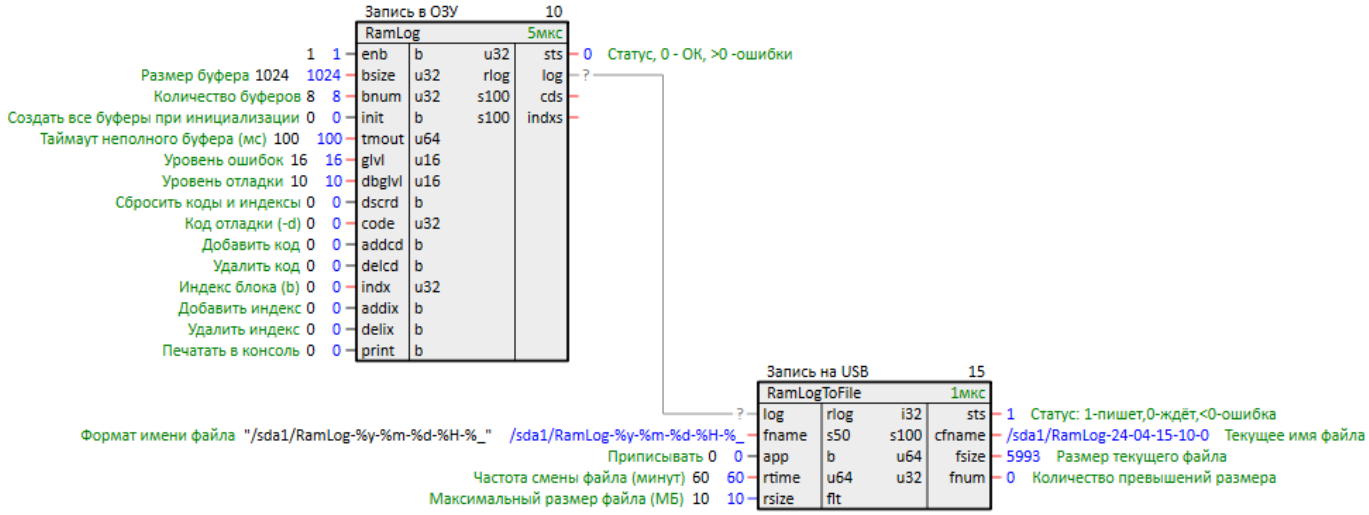


Рисунок 5.3 – Настройка вывода лога через RamLog

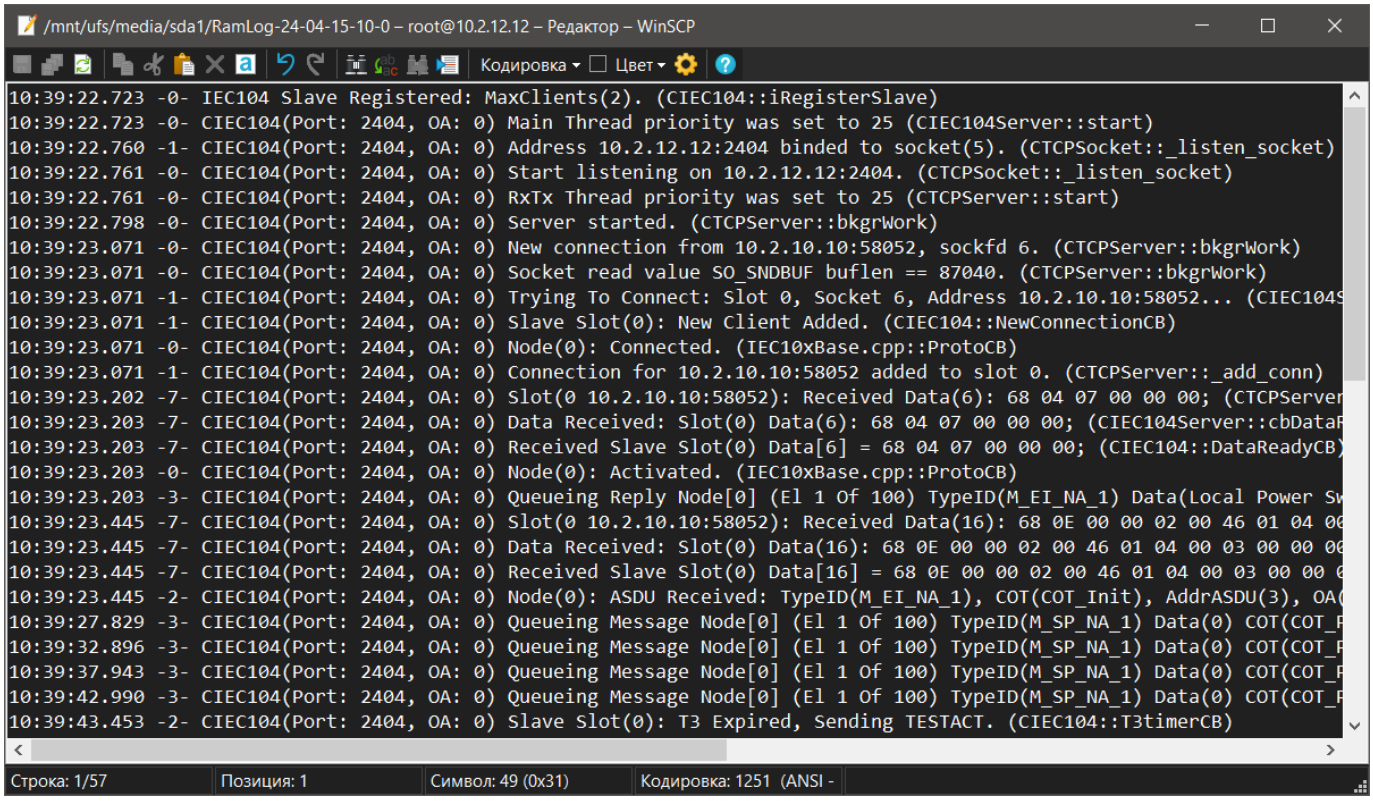


Рисунок 5.4 – Вывод сообщений в файл при dbg = 0x2C7

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***.

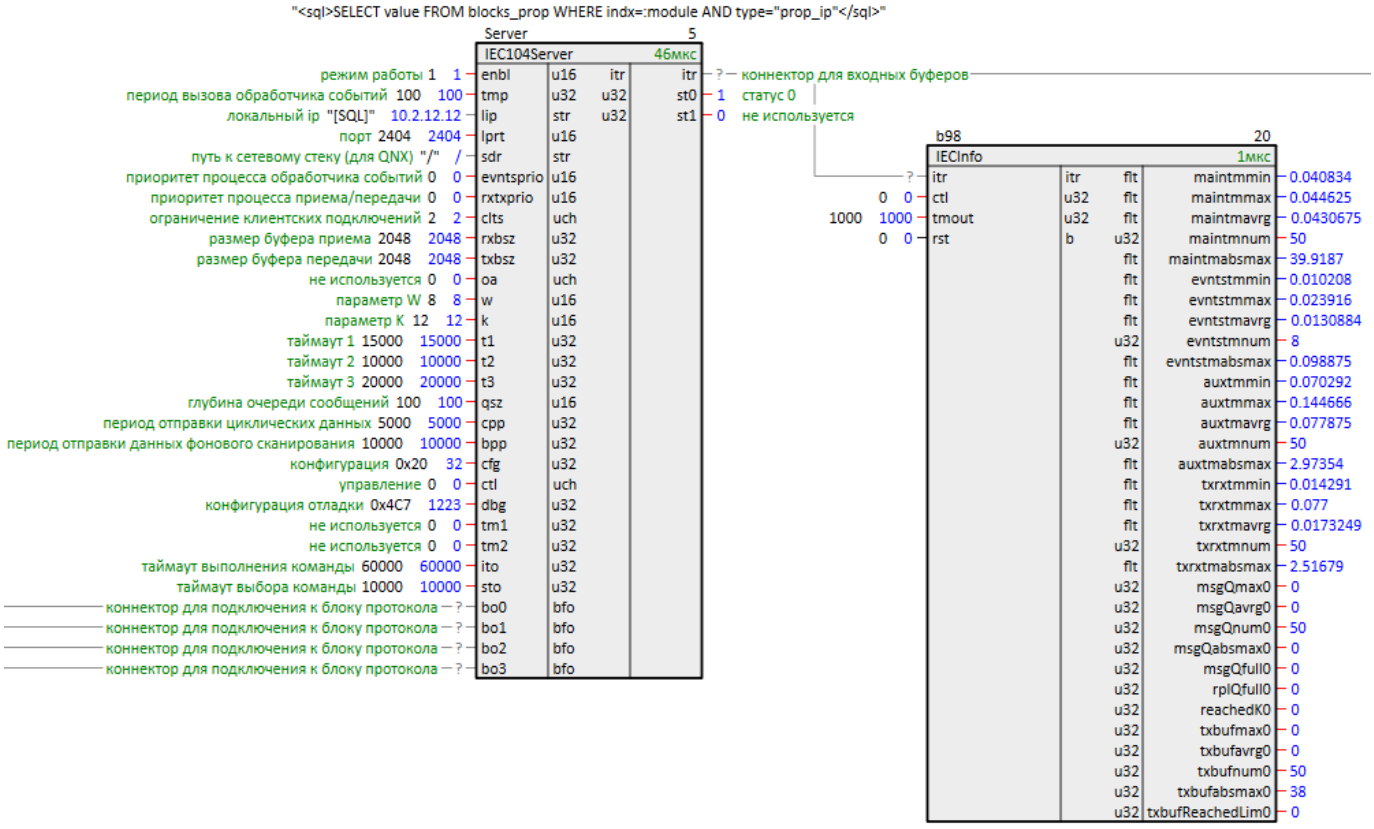


Рисунок 5.5 – Работа блока IECInfo

Приложение А. Описание типов информации ASDU

Таблица А.1 – Описание типов информации ASDU

Код	Вид информации	Метка	Реализация в palEC104
Информация о процессе в направлении контроля (М)			
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 отпаек	M_ST_NA_1	Нет
6	Информация о положении отпаек 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 **CP56Время2a**[1)]	M_SP_TB_1	Да
31	Двухэлементная информация с меткой времени **CP56Время2a**	M_DP_TB_1	Да
32	Информация о положении отпаек с меткой времени **CP56Время2a**	M_ST_TB_1	Нет
33	Строка из 32 бит с меткой времени **CP56Время2a**	M_BO_TB_1	Да
34	Значение измеряемой величины, нормализованное, с меткой времени **CP56Время2a**	M_ME_TD_1	Нет
35	Значение измеряемой величины, масштабированное, с меткой времени **CP56Время2a**	M_ME_TE_1	Да

Продолжение таблицы А.1

Код	Вид информации	Метка	Реализация в palEC104
36	Значение измеряемой величины, короткий формат с плавающей запятой с меткой времени **CP56Время2a**	M_ME_TF_1	Да
37	Интегральная сумма с меткой времени **CP56Время2a**	M_IT_TB_1	Да
38	Информация о работе релейной защиты с меткой времени **CP56Время2a**	M_EP_TD_1	Да
39	Упакованная информация о срабатывании пусковых органов защиты с меткой времени **CP56Время2a**	M_EP_TE_1	Да
40	Упакованная информация о срабатывании выходных цепей защиты с меткой времени **CP56Время2a**	M_EP_TF_1	Да
41…44	Резерв для дальнейших совместимых определений	-	-
Информация о процессе в направлении управления (С)			
45	**CON**[2)] Команда однопозиционная	C_SC_NA_1	Да
46	**CON** Команда двухпозиционная	C_DC_NA_1	Да
47	**CON** Команда пошагового регулирования	C_RC_NA_1	Нет
48	**CON** Команда уставки, нормализованное значение	C_SE_NA_1	Нет
49	**CON** Команда уставки, масштабируемое значение	C_SE_NB_1	Да
50	**CON** Команда уставки, короткое число с плавающей запятой	C_SE_NC_1	Да
51	**CON** Строка из 32 битов	C_BO_NA_1	Да
52…57	Резерв для дальнейших совместимых определений	-	-
58	Однопозиционная команда с меткой времени **CP56Время2a**	C_SC_TA_1	Да
59	Двухпозиционная команда с меткой времени **CP56Время2a**	C_DC_TA_1	Да
60	Команда пошагового регулирования с меткой времени **CP56Время2a**	C_RC_TA_1	Нет
61	Команда уставки, нормализованное значение с меткой времени **CP56Время2a**	C_SE_TA_1	Нет
62	Команда уставки, масштабированное значение с меткой времени **CP56Время2a**	C_SE_TB_1	Да
63	Команда уставки, короткий формат с плавающей запятой с меткой времени **CP56Время2a**	C_SE_TC_1	Да
64	Строка из 32 бит с меткой времени **CP56Время2a**	C_BO_TA_1	Да
65…69	Резерв для дальнейших совместимых определений	-	-
Системная информация в направлении контроля (М)			
70	Конец инициализации	M_EI_NA_1	Да
71…99	Резерв для дальнейших совместимых определений	-	-
Системная информация в направлении управления (С)			
100	**CON** Команда опроса	C_IC_NA_1	Да
101	**CON** Команда опроса счетчика	C_CI_NA_1	Да
102	**CON** Команда считывания	C_RD_NA_1	Да
103	**CON** Команда синхронизации часов	C_CS_NA_1	Да
104	**CON** Команда тестирования	C_TS_NA_1	Нет
105	**CON** Команда возврата процесса в исходное состояние	C_RP_NA_1	Нет
106	**CON** Команда передачи задержки	C_CD_NA_1	Нет
107	Команда тестирования с меткой времени **CP56Время2a**	C_TS_TA_1	Нет
108… 109	Резерв для дальнейших совместимых определений	-	-

Продолжение таблицы А.1

Код	Вид информации	Метка	Реализация в palEC104
Параметры в направлении управления (P)			
110	**CON** Параметр измеряемой величины, нормализованное значение	P_ME_NA_1	Нет
111	**CON** Параметр измеряемой величины, масштабируемое значение	P_ME_NB_1	Нет
112	**CON** Параметр измеряемой величины, короткое число с плавающей запятой	P_ME_NC_1	Нет
113	**CON** Параметр активации	P_AC_NC_1	Нет
114… 119	Резерв для дальнейших совместимых определений	-	-
Передача файлов (F)			
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
[1)] ASDU с метками времени **CP56Время2а** 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
[2)] 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 описание типов см. 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 (**Background Scan**)
3	Спорадическая передача при возникновении события (**Spontaneous**)
4	Сообщение об инициализации (**Initialized**)
5	Запрос или запрашиваемые данные (**Request Or Requested**)
6	Активация (**Activation**)
7	Подтверждение активации (**Activation Confirm**)
8	Деактивация (**Deactivation**)
9	Подтверждение деактивации (**Deactivation Confirm**)
10	Завершение активации (**Activation Termination**)
11	Обратная информация, вызванная удаленной командой (**Return Information Caused By A Remote Command**)
12	Обратная информация, вызванная местной командой (**Return Information Caused By A Local Command**)
13	Передача файлов (**File Transfer**)
14…19	Резерв
20	Общий опрос (**Interrogated By Station Interrogation**)
21…36	Опрос группы 1…16 (**Interrogated By Group 1…16 Interrogation**)
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-135062-1.1