Research Laboratory
of Design Automation
НИЛ автоматизации проектирования
Сделано в России
Тел.: +7 (495) 26-66-700 info@reallab.ru • https://www.reallab.ru

Среда разработки Полигон

Работа с ПЛК и модулями RealLab в среде разработки Полигон: создание, настройка и запуск проектов

Резервирование ПЛК RealLab в среде разработки Полигон

Руководство пользователя

© НИЛ АП, 2025

Версия от 1 сентября 2025 г.

Одной проблемой стало меньше!

Уважаемый покупатель!

Научно-исследовательская лаборатория автоматизации проектирования (НИЛ АП) благодарит Вас за использование нашей продукции и просит сообщать нам свои пожелания по улучшению программного обеспечения. Ваши пожелания можно направлять по почтовому или электронному адресу, а также сообщать по телефону:

НИЛ АП, пер. Биржевой спуск, 8, Таганрог, 347900,

Тел. (495) 26-66-700,

e-mail: info@reallab.ru, https://www.reallab.ru.

Вы можете также получить консультации по применению нашей продукции, воспользовавшись указанными выше координатами.

Авторские права на программное обеспечение, модуль и настоящее руководство принадлежат НИЛ АП.

Оглавление

1. Вводная часть 5

1.1. Введение 5

1.2. Общие сведения о среде разработки «Полигон» 5

1.3. Поддерживаемые контроллеры RealLab 6

2. Основы резервирования в среде Полигон 6

2.1. Термины 6

2.2. Функциональные возможности 7

2.3. Алгоритм резервирования 7

2.4. Подключение и схемы резервирования 8

2.5. Характеристики резервирования 10

2.6. Библиотеки и компоненты 10

2.6.1. Библиотека paSync 10

2.6.2. Блок OpcUAServer 10

2.6.3. Блок SyncMan 11

2.6.4. Блок MasterSel 14

2.6.5. Блок RedundancyStateCtrl 16

3. Состояния ПЛК при резервировании 17

3.1. Выбор ролей ПЛК при неопределенной ситуации 17

3.2. Описание состояний 18

3.3. Условия переключения состояний: 19

4. Пример проекта 21

4.1. Готовый пример проекта 21

4.2. Начало работы 21

4.3. Написание логики 22

4.3.1. Программа Reservation 22

4.4. Настройка Ethernet-интерфейсов на ПЛК 25
4.4.1. Настройка IP-адресов для резервирования на ПЛК 27
4.4.2. Установка IP-адресов в свойства проекта 28
4.5. Работа с протоколами 30
4.5.1. Подключение по Modbus RTU 31
4.5.2. Подключение по Modbus TCP 33
4.5.1. Запуск проекта 35
4.5.2. Синхронизация отладчика OPC UA 35
4.5.3. Синхронизация входов/выходов 36
4.5.4. Визуализация через MasterSCADA 4D 37
5. Проверка и диагностика 38
5.1. Переключатель Run-Stop 40
5.2. Мониторинг ролей и состояния резервирования через индикацию на корпусе ПЛК 40
5.3. Мониторинг ролей и состояния резервирования через отладчик Полигона (OPC UA) 41
Лист регистрации изменений 43

1. Вводная часть

1.1. Введение

Настоящее руководство посвящено настройке синхронизации проектов и организации резервирования контроллеров RealLab, программируемых в среде разработки Полигон. Документ предназначен для специалистов, уже имеющих базовые навыки работы с платформой Полигон. Поэтому вопросы, связанные с созданием, редактированием и загрузкой проектов, в данном руководстве не рассматриваются. Подробная информация по этим темам содержится в документе: «**Руководство по программированию ПЛК RealLab в среде разработки Полигон**», доступное по ссылке на сайте производителя.

Реализация синхронизации и резервирования в среде Полигон осуществляется с помощью функциональных блоков из библиотеки paSync. Для использования библиотеки paSync необходимо наличие соответствующей лицензии paSync.

1.2. Общие сведения о среде разработки «Полигон»

Полигон представляет собой комплекс программных решений, предназначенных для разработки, отладки и загрузки прикладного программного обеспечения в программируемые логические контроллеры (ПЛК). Основными компонентами системы являются интегрированная среда разработки и исполняющая среда, функционирующая непосредственно в ПЛК.

Программирование в Полигоне осуществляется на языке функциональных блоков (FBD). Система автоматизирует создание переменных, обеспечивая их уникальность и корректное преобразование типов данных, что устраняет необходимость их ручного описания разработчиком. Помимо алгоритмов, в форме функциональных блоков реализованы драйверы оборудования и протоколы обмена данными. Полигон также поддерживает ряд расширений, включая объектно-ориентированные подходы, что повышает гибкость и удобство разработки сложных проектов.

Также доступны языки C++ и ST.

Данный подход позволяет сосредоточиться на проектировании логики приложения, минимизируя рутинные задачи, связанные с низкоуровневой реализацией.

1.3. Поддерживаемые контроллеры RealLab

Ниже приведен список контроллеров RealLab, поддерживающих среду разработки Полигон:

- NLScon-RSB-S-PLG.
- NLScon-RSB-XS-PLG-RD.

2. Основы резервирования в среде Полигон

2.1. Термины

Ведомый контроллер - устройство, находящееся в режиме горячего резерва и обеспечивающее синхронизацию данных с ведущим контроллером.

Ведущий контроллер - устройство, с которым синхронизируется ведомый контроллер. В зависимости от схемы резервирования он может выполнять функции мастера для модулей ввода/вывода, управлять выходными сигналами на своей платформе и выполнять другие критически важные задачи.

Переключение ролей / определение ведущего контроллера - алгоритм, устанавливающий текущую роль контроллера (ведущий или ведомый), который может быть реализован как автоматически, так и по команде оператора.

Резервирование (согласно ГОСТ 27.002-89) - метод повышения надежности системы за счёт использования дополнительных средств или возможностей, избыточных относительно минимально необходимых для выполнения заданных функций.

Синхронизация данных - процесс устранения расхождений между идентичными наборами данных контроллеров путём обмена информацией по выделенным каналам связи.

2.2. Функциональные возможности

Среда разработки Полигон поддерживает следующие возможности для организации резервирования контроллеров:

- Копирование пользовательских программ (модулей контроллеров в проекте Полигон) на этапе разработки — как полностью, так и выборочно;
- Синхронизация данных между дублированными программами в режиме реального времени с использованием блоков *_sync и раздела SyncMan из библиотеки paSync;
- Автоматическое переключение ролей контроллеров (ведущий/ведомый) через блок MasterSel из библиотеки paSync;
- Ручное управление переключением ролей контроллеров с применением того же блока MasterSel;
- Полная свобода в реализации пользовательских алгоритмов переключения ролей без ограничений со стороны среды.

Схемы резервирования ПЛК Reallab в среде Полигон отличаются высокой адаптивностью, позволяя гибко модернизировать их под задачи конкретных технологических процессов.

2.3. Алгоритм резервирования

Два ПЛК с загруженной в них исполнительной системой Полигон специальной конфигурации работают одновременно. Один имеет статус «Ведущий» (MASTER), другой – «Ведомый» (SLAVE).

Из коробки предоставляется механизм для присвоения и получения текущего статуса: Ведущий», Ведомый, Отсутствие связи (автономный Ведущий). Все остальные механизмы пользователь реализует самостоятельно, включая связь с модулями и её резервирование.

Программы на обоих контроллерах выполняются независимо. Опрос устройств могут вести два контроллера одновременно, так и только один - Ведущий, в зависимости от топологии сети и настройки. При возникновении отказа основного контроллера, управление переходит к резервному.

Оба ПЛК имеют OPC UA – сервера для подключения к ним извне. При редактировании параметров удаленно через OPC UA, они будут синхронно

изменяться на обоих ПЛК, для этого необходимо указать не равное нулю максимальное количество синхронизируемых параметров отладчика (вход *fnum* блока *SyncMan*).

2.4. Подключение и схемы резервирования

Варианты построения схем резервирования ПЛК RealLab с использованием исполнительной среды Полигон практически не ограничены и могут быть адаптированы или модернизированы в зависимости от требований конкретного технологического объекта. Ниже приведены типовые примеры схем организации резервирования ПЛК RealLab в среде Полигон.

Для реализации резервирования модулей под протоколом Modbus RTU необходимо подключить набор модулей к портам RS-485 с одинаковым номером (0 или 1) обоих ПЛК (рис 2.1).

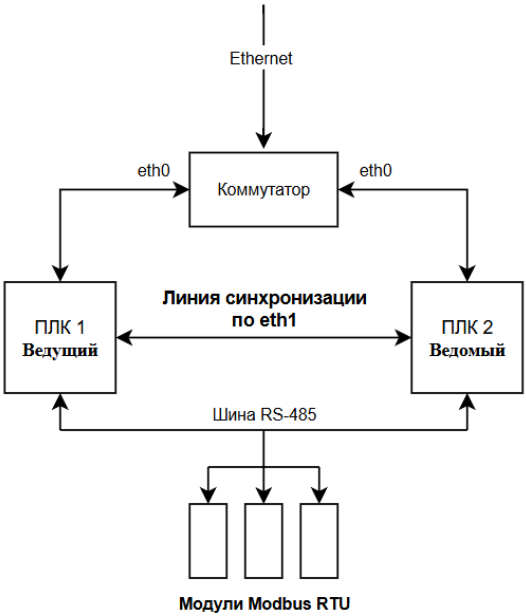


Рис 2.1. – Схема подключения резервируемых ПЛК с модулями Modbus RTU

Для реализации резервирования модулей под протоколом Modbus TCP достаточно подключить резервируемые ПЛК, основные и резервные модули в одну подсеть (рис 2.2).

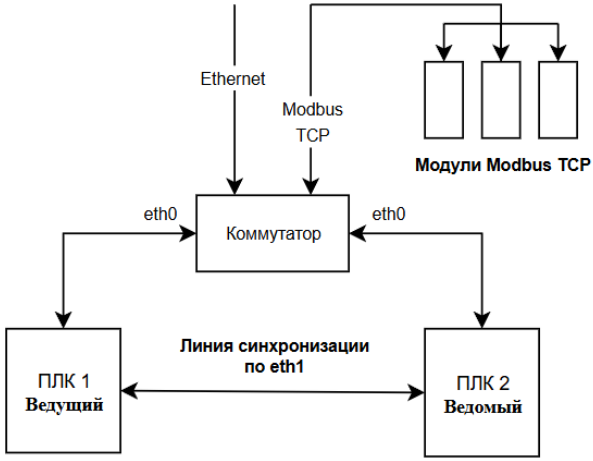


Рис 2.2. – Схема подключения резервируемых ПЛК с модулями Modbus TCP

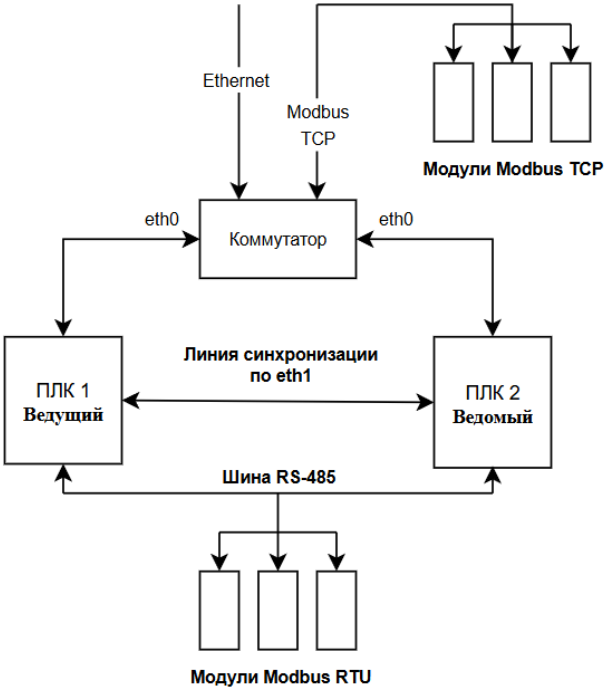


Рис 2.3. – Схема подключения резервируемых ПЛК с модулями Modbus RTU и Modbus TCP

2.5. Характеристики резервирования

Характеристики резервирования зависят от самого проекта и его конфигурации.

Минимальное время переключения: 400 мс.

Поддерживаемые протоколы: OPC UA, Modbus RTU/TCP.

2.6. Библиотеки и компоненты

Для реализации синхронизации необходимы 3 библиотеки:

- *paSync* – библиотека синхронизации;
- *paReallab* – драйверы ПЛК RealLab;
- *paOpcUA* – поддержка OPC UA.

В проекте-примере также присутствуют:

- *paModbus* – библиотека для работы с Modbus RTU/TCP;
- *rlbModulesModbus* – библиотека для работы с модулями ввода/вывода Reallab по протоколам Modbus RTU/TCP.

2.6.1. Библиотека paSync

paSync – библиотека, обеспечивающая синхронизацию данных между дублированными проектами контроллеров и организацию резервирования. Один из контроллеров выбирается ведущим – с ним синхронизируется ведомый контроллер.

Синхронизация сигналов во время работы контроллеров обеспечивается для блоков с поддержкой синхронизации из библиотеки paSync, которые реализуют базовые алгоритмы аналогично блокам из библиотек paCore и paControls. Работа данных блоков описана в справке среды на соответствующие библиотеки и в данном документе не рассматривается.

2.6.2. Блок OpcUAServer

Блок OpcUAServer (рис 2.4) обеспечивает реализацию сервера по протоколу OPC UA. OpcUAServer используется для синхронизации данных и работы механизма резервирования ПЛК. В табл. 1 представлены входы/выходы блока OpcUAServer.

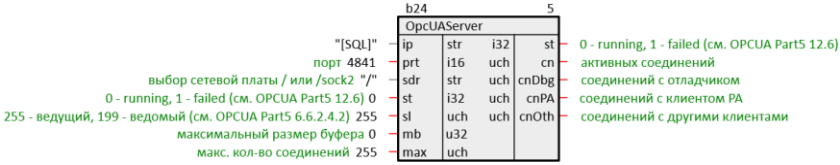


Рис 2.4. – Блок OpcUAServer

Табл. 1 – Входы/выходы блока OpcUAServer

Входы	**Назначение**
ip	Локальный IP адрес
prt	Локальный порт
sdr	Сетевой стек (только для QNX):"/", "/sock2" и т.д.
st	Статус сервера в соответствии со спецификацией OpcUA, особый случай -1 - выключает сервер полностью
sl	Дополнительный статус сервера, ServiceLevel в соответсвии со спецификацией OpcUA
max	Максимально разрешенное количество соединений
Выходы	**Назначение**
st	Текущий статус сервера (см. вход st)
cn	Количество активных соединений (сумма cnDbg, cnPA, cnOth)
cnDbg	Количество соединений с отладчиком среды разработки
cnPA	Количество соединений с клиентами OpcUAClient
cnOth	Количество соединений с другими клиентами

2.6.3. Блок SyncMan

Блок SyncMan (рис 2.5) обеспечивает синхронизацию данных между ведущим и ведомым контроллером. Обмен реализован через протокол OPC UA, в ведомом контроллере создается подписка на изменение данных от ведущего.

SyncMan основан на блоке OpcUAClient из библиотеки paOpcUA. Данный блок можно разместить только в Фоне.

Данные, необходимые для синхронизации блоков библиотеки paSync, добавляются в подписку автоматически. Дополнительные входы/выходы, которыми необходимо обмениваться с сервером, должны быть добавлены в раздел Данные внутри этого блока. В табл. 2 представлены входы/выходы блока SyncMan.

Выходы функциональных блоков, добавленные в раздел Данные передаются на сервер всегда, в соответствии со свойствами (подробнее см. в описании блока OpcUAClient из библиотеки paOpcUA).

Входы функциональных блоков, добавленные в раздел Данные, читаются из сервера в соответствии со следующими правилами:

- Если у входа есть свойство ID источника/приемника, то он читается из сервера всегда при наличии связи. Такие данные используются для двустороннего обмена между ведущим и ведомым независимо от текущей роли контроллера.
- Если у входа нет свойства ID источника/приемника, то он читается из сервера только при включенной синхронизации (***sync = 1***). Такие данные используются для синхронизации вручную.

Для синхронизации параметров можно использовать блоки BufSupEx, подключенные к входам m_rbufs блока SyncMan (дополнительные входы добавляются командой «Создать»). Значения параметров блока BufSupEx синхронизируются только при ***sync = 1***.

ВАЖНО! Для правильной работы устанавливайте таймаут (wait) больше 500мс.

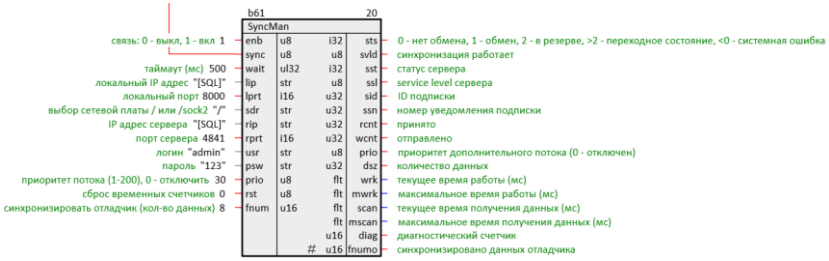


Рис 2.5. – Блок SyncMan

Табл. 2 – Входы/выходы блока SyncMan

Входы	Назначение
enb	Разрешение на работу блока
sync	Синхронизация: 0 – выключена; 1 – включена. На данный вход может быть подан инвертированный сигнал с выхода L_Master блока MasterSel или пользовательского блока, реализующего алгоритм определения ведущего контроллера

wait	Таймаут ожидания ответа от сервера, мс (константный)
lip	Локальный IP адрес (константный)
lprt	Локальный порт (константный)
sdr	Сетевой стек, для ПЛК Reallab "/" (константный)
rip	IP адрес сервера (константный)
rprt	Порт сервера (константный)
usr	Логин для доступа к серверу (константный)
psw	Пароль для доступа к серверу (константный)
prio	Приоритет дополнительного потока (константный), в котором выполняется синхронизация, обычно устанавливается выше других фоновых потоков, чтобы обеспечить максимальную скорость синхронизации: 0 – отключает создание дополнительного потока (обмен идет в текущем фоновом потоке); 1…47 – приоритет потока
rst	Сброс максимальных значений временных счетчиков: выходов mwrk и mscan
fnum	Максимальное количество подмененных во время отладки входов/выходов, которое можно синхронизировать: 0 – отключает синхронизацию подмененных значений.
m_rbufs	Вход для подключения блоков типа BufSupEx (циклический)
Выходы	**Назначение**
sts	Статус работы: 0 – нет обмена; 1 – обмен; 2 – в резерве; >2 – переходное состояние
svld	Синхронизация работает
sst	Статус сервера в соответствии со спецификацией OPC UA
ssl	Дополнительный статус сервера ServiceLevel в соответствии со спецификацией OPC UA
sid	ID подписки
ssn	Номер уведомления подписки
rcnt	Количество принятых пакетов
wcnt	Количество отправленных пакетов
prio	Приоритет дополнительного потока (0 – отключен)
dsz	Количество данных
wrk	Текущее время работы, мс
mwrk	Максимальное время работы, мс
scan	Текущее время получения данных, мс
mscan	Максимальное время получения данных, мс

diag	Диагностический счетчик
fnumo	Количество синхронизируемых подмененных во время отладки входов/выходов

2.6.4. Блок MasterSel

Блок MasterSel (рис 2.6) определяет роли Ведущий/Ведомый для двух контроллеров в резервированной конфигурации.

После запуска программы контроллера выход блока ***L_Master = 0***, т.е. ПЛК является ведомым.

Срабатывание любого из условий переключения в роль ведущего возможно только после завершения инициализации (выход ***L_init_over = 1***). Инициализация считается завершенной после окончания отсчета времени tinit от появления единицы на входе L_init (обычно сразу установлена 1).

Контроллер может стать ведущим в случаях:

- Если он ведомый и готов (***L_Master = 0*** и ***L_ready = 1***), а соседний – не готов (***R_ready = 0***);
- По получению соседним ведущим контроллером команды на смену мастера (master1 или master2). Тогда этот контроллер станет ведущим, если он готов, или оба контроллера не готовы;
- После восстановления связи между ПЛК, когда оба ведомые, и этот контроллер – ПЛК1 (вход ***me1 = 1***);
- При потере связи с соседним ПЛК (***conn_fault = 0***).

Контроллер может стать ведомым в случаях:

- Если он ведущий и не готов (***L_Master = 1*** и ***L_ready = 0***), а соседний – готов (***R_ready = 1***);
- По получению им команды на смену мастера (master1 или master2). Тогда другой контроллер станет ведущим, если он готов, или оба контроллера не готовы;
- После восстановления связи между контроллерами, когда оба ведущие, и этот ПЛК – ПЛК2 (вход ***me1 = 0***).

В иных случаях смены ролей не происходит.

Готовность контроллера (выход L_ready блока) формируется на основании двух условий по логике «И»:

- Есть внешнее условие готовности, заводимое на вход ready;

- Инициализация связи завершена.

С соседним контроллером через две выделенные линии синхронизации производится обмен данными и контролируется связь посредством передачи по каждой линии пилообразного сигнала:

- Если не происходит обновления значений пилообразных сигналов ни по одной линии за заданное время (вход tpila), то фиксируется потеря связи между ПЛК (выход ***conn_fault = 1***);
- Наличие связи фиксируется (выход conn_fault = 0) с задержкой времени (вход trecon) после того, когда снова начинают изменяться значения передаваемых пилообразных сигналов;
- Текущие значения пилообразного сигнала выдаются на выходы: L_pila и R_pila_1, R_pila_2, соответственно, собственный сигнал и сигнал от второго контроллера по двум линиям связи.

Для обмена двумя диагностическими сигналами в проекте контроллера должно быть добавлено два менеджера SyncMan, каждый из которых связывается по своему порту.

Выход L_Master используется в прикладной программе для задания признака ведущего у линейки ввода/вывода. А его инверсия – как признак необходимости синхронизации (входы sync блоков SyncMan). Другие выходы блока могут использоваться для диагностики и в пользовательской логике.

ВАЖНО! Для правильной работы устанавливайте таймеры *tpila < trecon < tinit.*

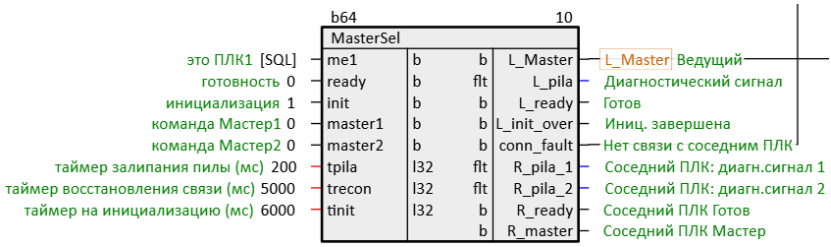


Рис 2.6. – Блок MasterSel

Табл. 3 – Входы/выходы блока MasterSel

Входы	Назначение
me1	Признак ПЛК1 (me1=1 у ПЛК1). Под ПЛК1 подразумевается тот контроллер, который должен

	становится ведущим при неопределенных условиях (восстановление связи между двумя работающими контроллерами)
ready	Внешнее условие готовности контроллера (разрешение стать основным)
init	Запуск таймера на инициализацию
master1	Команда ПЛК1 стать ведущим (внутри выделяется фронт с 0 на 1)
master2	Команда ПЛК2 стать ведущим (внутри выделяется фронт с 0 на 1)
tpila	Таймер залипания пилы, мс
trecon	Таймер восстановления связи, мс
tinit	Таймер на инициализацию, мс
Выходы	**Назначение**
L_Master	Признак ведущего: 0 – данный контроллер ведомый; 1 – данный контроллер ведущий
L_pila	Диагностика: генерируемый пилообразный сигнал этого контроллера
L_ready	Диагностика: готовность этого контроллера
L_init_over	Инициализация завершена
conn_fault	Нет связи с соседним контроллером
R_pila_1	Диагностика: пила соседнего контроллера по каналу связи 1
R_pila_2	Диагностика: пила соседнего контроллера по каналу связи 2
R_ready	Диагностика: готовность соседнего контроллера
R_master	Диагностика: соседний контроллер ведущий

2.6.5. Блок RedundancyStateCtrl

Блок RedundancyStateCtrl (рис 2.7) предназначен для управления индикаторами состояния в зависимости от состояния резервирования ПЛК (по входам), а также для просмотра их текущих значений. Рекомендуется использовать совместно с блоком paSync_MasterSel.

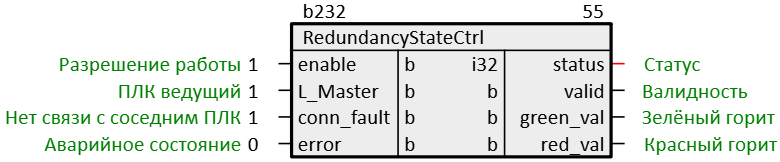


Рис 2.7. – Блок RedundancyStateCtrl

Табл. 4 – Входы/выходы блока RedundancyStateCtrl

Входы	Назначение
enable	Разрешение работы
L_Master	Если 1, то ПЛК ведущий, если 0 - ведомый
conn_fault	Если 1, то связи с соседним ПЛК нет, если 0 - есть
error	Если 1, то возникла ошибка или другое аварийное состояние. Если 0 - нет
Выходы	**Назначение**
status	Статус, равный 1, если блок работает корректно, и <0 в случае ошибки. Список ошибок находится в разделе "Статусы блока" в справке блока
valid	Валидность. Равен 1 в случае отсутствия ошибок
green_val	Текущее значение зелёного светодиода. 0 - светодиод выключен. 1 - светодиод включён
red_val	Текущее значение красного светодиода. 0 - светодиод выключен. 1 - светодиод включён

3. Состояния ПЛК при резервировании

3.1. Выбор ролей ПЛК при неопределенной ситуации

Неопределенная ситуация возникает при обрыве линии синхронизации – оба ПЛК являются ведущими.

При восстановлении линии синхронизации выбор кто станет ведущим, а кто ведомым не определен.

Чтобы выбор был определенным, нужно выполнить следующие действия:

1. В свойствах модулей добавить “Пользовательское свойство 02” и ввести в него 1 на предполагаемом ведущем ПЛК (рис 3.1).

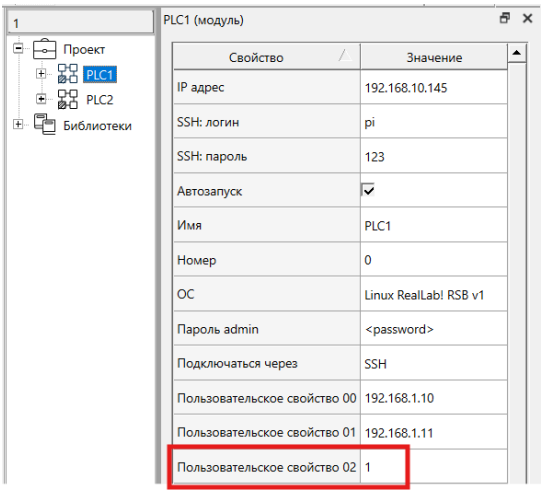


Рис 3.1. – Пользовательское свойство 2

2. На вход “me1” функционального блока MasterSel вставить SQL-запрос (рис 3.2):

   <sql>SELECT value FROM blocks_prop WHERE indx=:module AND type="prop_2"</sql>

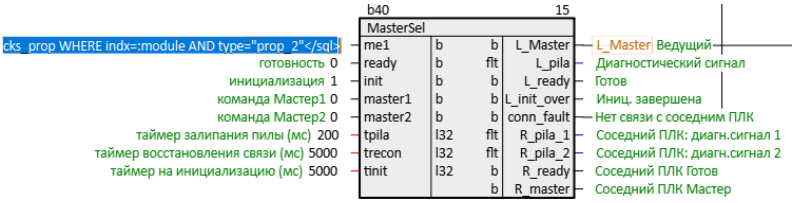


Рис 3.2. – Вход “me1” функционального блока MasterSel

3.2. Описание состояний

Варианты состояний контроллеров при резервировании и причины их изменения приведены в табл. 1. ПЛК могут находиться в одном из следующих состояний:

Ведущий – штатное состояние основного ПЛК в резервной паре. В этом состоянии ПЛК выполняет пользовательское приложение, осуществляет управление технологическим процессом и отправляет сообщения второму (Ведомому) ПЛК.

Ведомый – штатное состояние резервного ПЛК в резервной паре. В этом состоянии ПЛК также может вести опрос, если это позволяет используемый протокол. Как правило, опрос не производится. ПЛК переходит в состояние Master, если текущий Master переходит в состояние “Ведомому” или “Отключен”.

Отключен – состояние ПЛК, когда он отключен от питания, либо вышел из строя, либо выполнение проекта остановлено (например, переключателем Run-Stop в состоянии Stop).

Также можно выделить ещё одно состояние:

Ведущий автономный – То же самое, что и Ведущий, но с условием, что второй ПЛК не отвечает на запросы – отключен, без питания или отключен от линии синхронизации.

3.3. Условия переключения состояний:

Условия перехода из одного состояния ПЛК в другое при резервировании описаны в Табл. 5.

Читать таблицу следующим образом:

- текущее состояние ПЛК 1 – столбец,
- ячейка в таблице – условие перехода,
- итоговое состояние – строка,
- пунктирная линия между условий перехода означает **ИЛИ.**

Табл. 5 – матрица смежности состояний ПЛК при резервировании

ПЛК 1	Master	Slave	Master (автономный)	Выключен или переключатель Run-Stop в Stop
Master	-	На ПЛК 1 в блоке “MasterSel” на один из входов: “master1”, “master2” было подано значение 1 в	ПЛК 2 выключился Обрыв линии синхронизации	Не было питания дольше 10 сек

		зависимости от того, на каком из ПЛК вход “me1” установлен в 1		Переключатель Run-Stop переведён в Stop
Slave	На ПЛК 2 в блоке “MasterSel” на один из входов: “master1”, “master2” было подано значение 1 в зависимости от того, на каком из ПЛК вход “me1” установлен в 1	-	ПЛК 2 выключился	Не было питания дольше 10 сек
			Обрыв линии синхронизации	Переключатель Run-Stop переведён в Stop
Master (автономный)	ПЛК 2 включился	Переходов нет	-	Не было питания дольше 10 сек
	Линия синхронизации восстановлена, с условием, что на ПЛК 1 в блоке “MasterSel” на входе “me1” значение 1, а на ПЛК 2 значение 0			Переключатель Run-Stop переведён в Stop
Выключен или переключатель Run-Stop в Stop	Переходов нет	ПЛК 1 включился при включенном ПЛК 2	ПЛК 1 включился при выключенном ПЛК 2	-
			ПЛК 1 включился при обрыве линии синхронизации	

4. Пример проекта

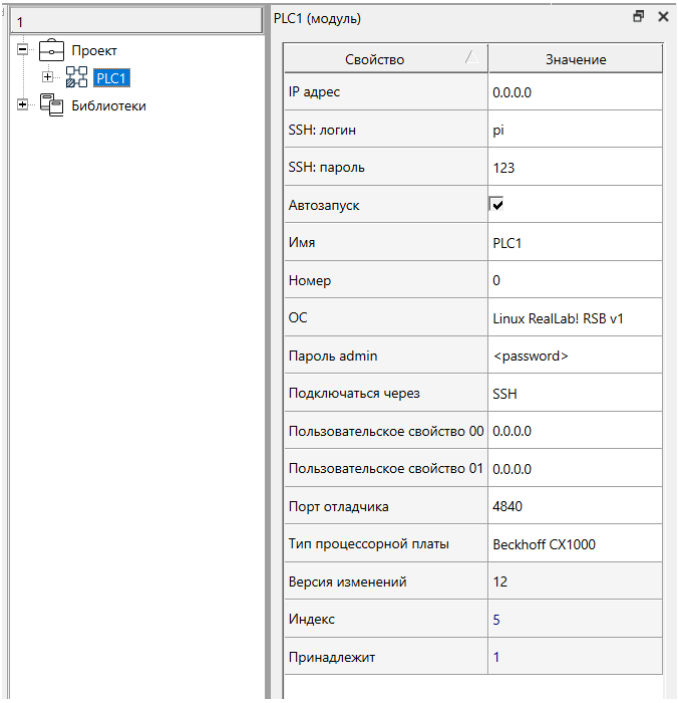
4.1. Готовый пример проекта

Для примера работы резервирования будет создан проект. Готовую его версию можно найти на сайте на странице ПЛК NLScon-RSB-XS-PLG-RD.

4.2. Начало работы

Скачайте и установите среду разработки. Запустите Полигон, создайте проект с отладчиком, скачайте и импортируйте библиотеку с шаблонами модулей(rlbModulesModbus), библиотеку ПЛК RealLab(paReallab) и библиотеку для организации резервирования(paSync).

Добавьте в созданный модуль свойства для работы с ПЛК RealLab и резервирования (рис 4.1). Переименуем модуль в PLC1 для удобства.



Свойство	Значение
IP адрес	0.0.0.0
SSH: логин	pi
SSH: пароль	123
Автозапуск	☑
Имя	PLC1
Номер	0
ОС	Linux RealLab! RSB v1
Пароль admin	<password>
Подключаться через	SSH
Пользовательское свойство 00	0.0.0.0
Пользовательское свойство 01	0.0.0.0
Порт отладчика	4840
Тип процессорной платы	Beckhoff CX1000
Версия изменений	12
Индекс	5
Принадлежит	1

Рис 4.1. – Свойства модуля

Скопируйте модуль, в который добавили свойства, в этот же проект.

Перетащите места работы Таймер и Фон из PLC1 в PLC2, выберите пункт “Добавить” (рис 4.2). Удалите скопированные старые места работы из PLC2 (Таймер_01, Фон_01).

Таким образом все изменения в местах работы одного модуля будут дублироваться во втором модуле, и наоборот.

Будут дублироваться не только сами блоки, но и их расположение, значения инициализации на входах блоков, комментарии, связи.

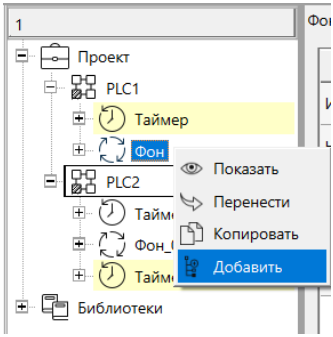


Рис 4.2. – Добавление места работы в другой модуль

4.3. Написание логики

4.3.1. Программа Reservation

Создадим в месте работы “Фон” программу, выбрав язык функциональных блоков, назовём её “Reservation”.

Блок OpcUAServer

1. Откроем страницу, добавим туда блок “OpcUAServer” из библиотеки “paOpcUA”.
2. Вставим во вход “ip” SQL запрос к свойствам модуля:

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

Вход “ptr” – *4841;*

Сервер OPC UA необходим для работы резервирования.

Блок Sync Man

1. Создадим блок “Sync Man” из менеджеров библиотеки “paSync”.
2. Вставим во входы SQL-запросы:

Вход “lip” - *"<sql>SELECT value FROM blocks_prop WHERE indx=:module AND type="prop_0"</sql>";*

Вход “rip” - *"<sql>SELECT value FROM blocks_prop WHERE indx=:module AND type="prop_1"</sql>".*

3. В остальные входы вставим:

Вход “wait” – *100;*

Вход “rprt” – *4841;*

Вход “usr” – *“admin”;*

Вход “psw” – *“123”* (или Ваш пароль в свойствах модуля).

Блок является главным менеджером для работы сихронизации.

Блок MasterSel

1. Создадим блок “MasterSel” из менеджеров библиотеки “paSync”.
2. Вставим во входы:

Вход “trecon” - *100*

Вход “tinit” – *100*

3. Откроем свойства выхода “L_Master”, добавим 2 новых свойства: “Полный алиас” и “Глобальная ссылка”.
4. В свойстве “Полный алиас” пропишем произвольное название – “L_Master”.

Таким образом мы сделаем выход блока глобальным и его будет проще соединять. Этот выход будет соединяться со множеством других блоков, так как он возвращает роль текущего ПЛК – мастер или слейв.

Блок MasterSel отвечает за выбор и переключения роли ПЛК при резервировании.

Блок RedundancyStateCtrl

1. Создадим блок “RedundancyStateCtrl” из библиотеки “paReallab”.

Блок отвечает за светодиодную индикацию текущего состояния резервирования.

Соединение блоков

1. Создадим блок NOT из логических блоков библиотеки “paCore”.
2. Соединим L_Master (MasterSel) со входом блока NOT.
3. Выход блока NOT соединим с sync (SyncMan).

Таким образом SyncMan будет делать попытки синхронизации с ПЛК-Мастером только на ПЛК-Слейве.

4. Соединим выход L_Master (MasterSel) со входом L_Master (RedundancyStateCtrl).
5. Соединим выход conn_fault (MasterSel) со входом conn_fault (RedundancyStateCtrl).

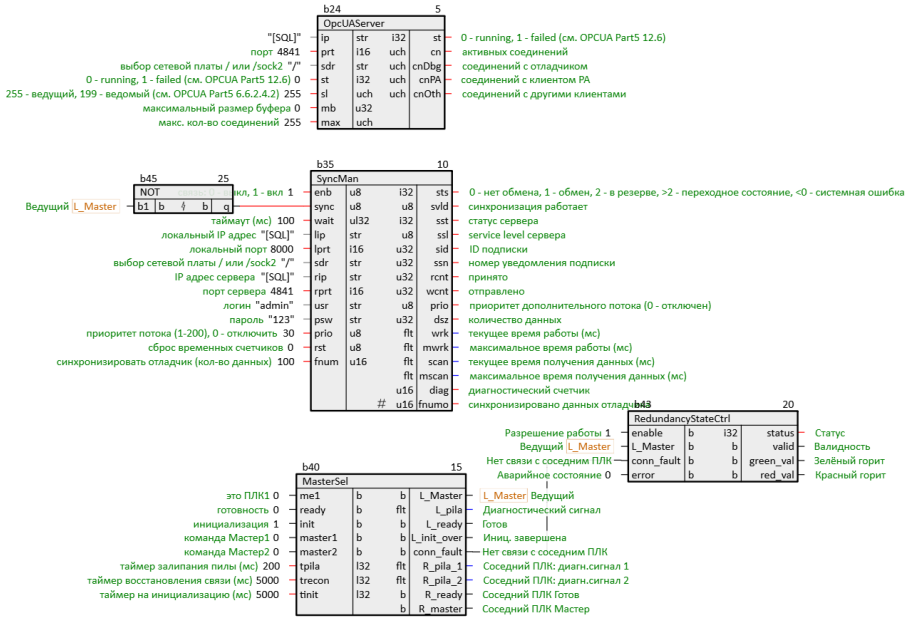


Рис 4.3. – Готовая программа “Reservation”

Следующим этапом идёт настройка статических ip адресов на ethernet портах, предназначенных для резервирования.

4.4. Настройка Ethernet-интерфейсов на ПЛК

ВАЖНО! IP-адреса интерфейсов eth1 обоих ПЛК должны принадлежать к одной подсети, а IP-адреса интерфейсов eth0 обоих ПЛК – к другой (отличающейся от подсети, к которой принадлежат адреса eth1).

Через консольный конфигуратор rl-config на NLScon-RSB

Для запуска конфигуратора на ПЛК серии NLScon-RSB необходимо открыть терминал (можно удаленно по SSH) и выполнить следующую команду:

#sudo rl-config

При запуске rl-config откроется главное окно. Общий вид главного окна представлен на рис 4.4.

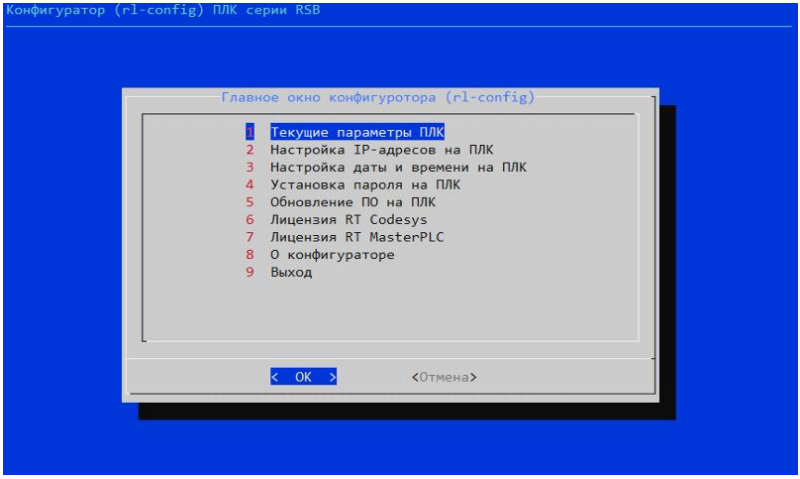


Рис 4.4. - Общий вид главного окна конфигуратора

Для настройки IP-адресов ПЛК в открытом диалоговом окне нужно выбрать из меню конфигуратора «Настройка IP-адресов на ПЛК» и нажать клавишу «Enter», появится диалоговое окно с интерфейсами Ethernet на ПЛК (рис 4.5).

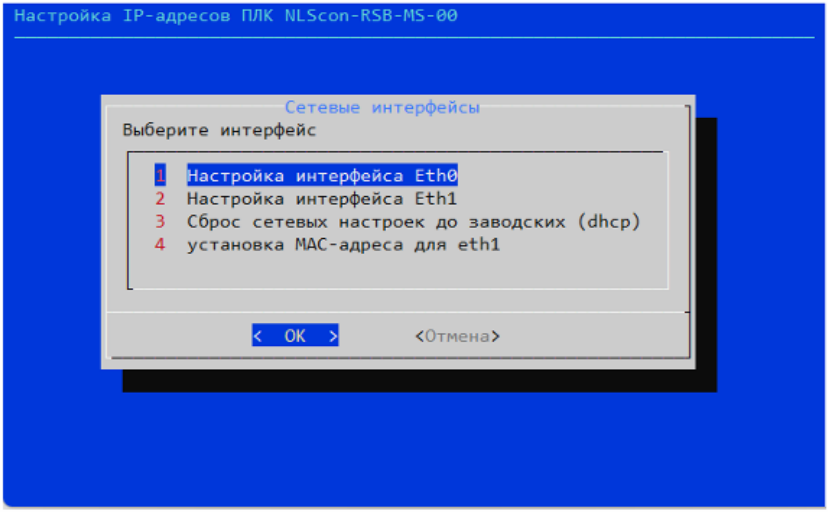


Рис 4.5. – Интерфейсы Ethernet на ПЛК

В открытом диалоговом окне интерфейсов Ethernet выберите интерфейс и нажмите на клавишу «Enter», появится диалоговое окно с режимами подключения интерфейсов Ethernet (рис 4.6).

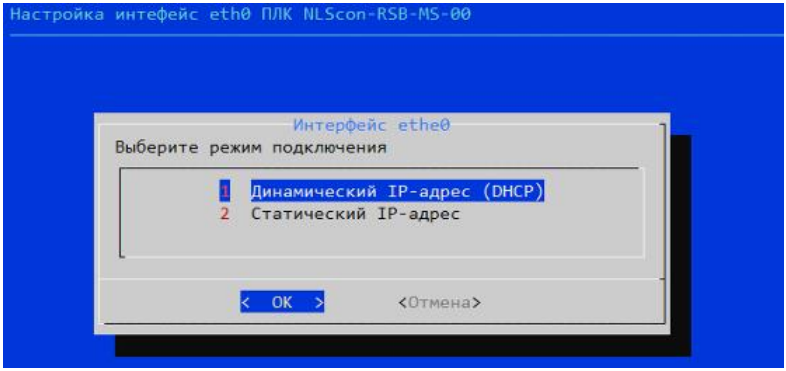


Рис 4.6. – Режим подключения интерфейсов Ethernet

Если выбрать режим статического IP-адреса для интерфейса, то можно ввести параметры статического IP-адреса как показано на рис 4.7.

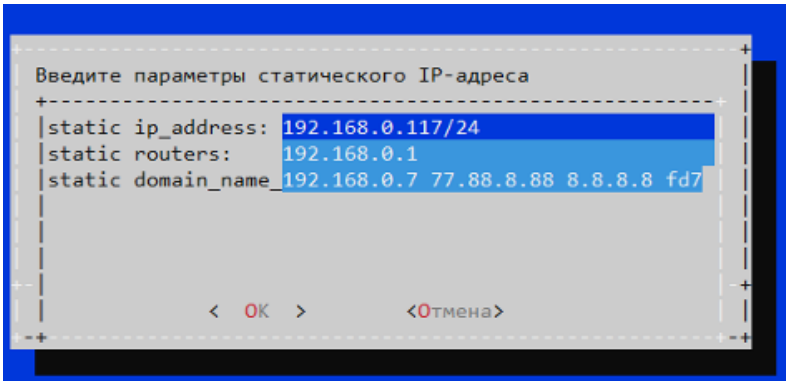


Рис 4.7. – Параметры статического IP-адреса

Далее с помощью кнопки Tab перейдите на кнопку < OK > и нажмите Enter на клавиатуре. Конфигуратор предложит сохранить изменения – выберите «Да» и снова нажмите Enter.

Более подробная информация по использованию конфигуратора предоставлена по ссылке.

4.4.1. Настройка IP-адресов для резервирования на ПЛК

Соедините ПЛК между собой ethernet-кабелем через порты eth1. Для резервирования можно использовать также eth0 или оба интерфейса сразу. При соединении одним кабелем рекомендуется использовать порт eth1.

Произвольно выберите какой ПЛК будет первым и какой вторым.

Просмотрите текущие параметры ПЛК, убедитесь, что MAC-адреса портов eth1 и eth0 отличаются друг от друга.

Сделайте ip адреса портов, предназначенных для резервирования, статическими и задайте их так, что они находились в одной подсети.

Пример настройки двух ПЛК представлен на рис 4.8 - рис 4.9.

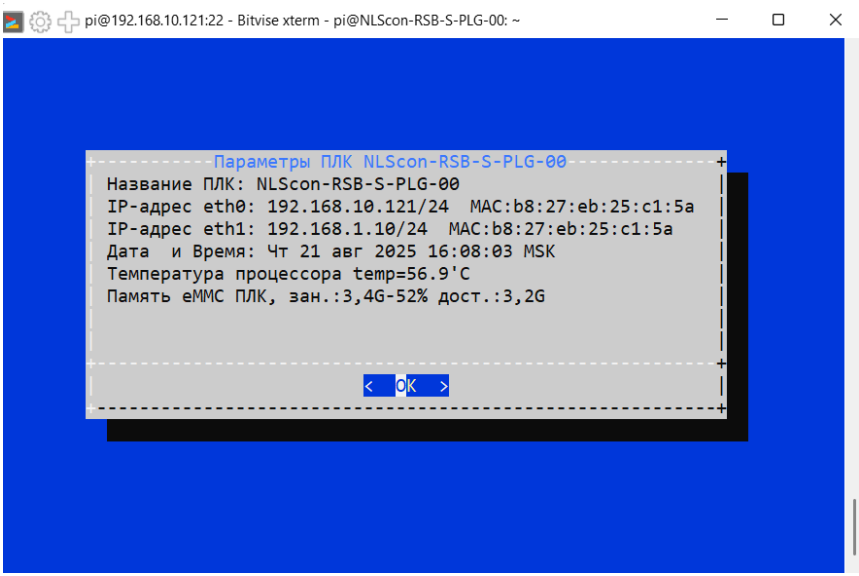


Рис 4.8. – Параметры ПЛК 1

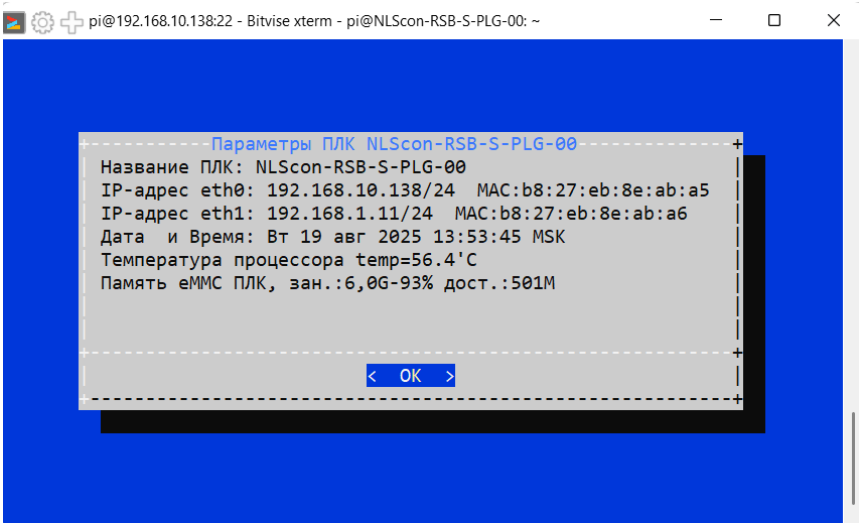


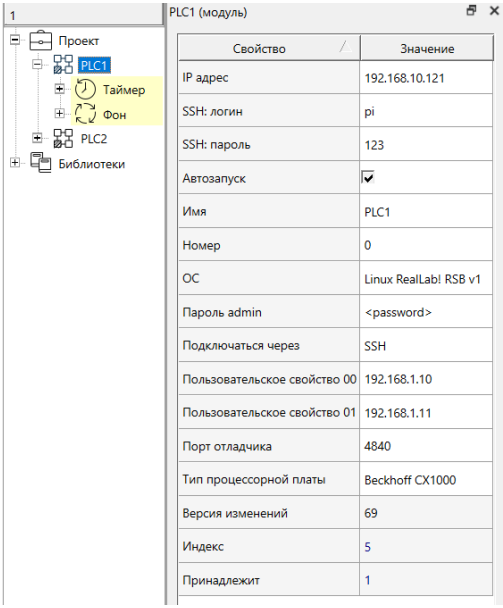
Рис 4.9. – Параметры ПЛК 2

4.4.2. Установка IP-адресов в свойства проекта

В свойствах модуля “PLC1”:

1. Впишем в “IP адрес” адрес порта, не используемого для резервирования.
2. Впишем в “Пользовательское свойство 00” адрес порта, используемого для резервирования.
3. Впишем в “Пользовательское свойство 01” адрес порта, используемого для резервирования на PLC2.
4. В свойствах модуля “PLC2” проделаем то же самое, для “Пользовательского свойства 01” используем адрес порта PLC1.

Пример установки IP-адресов представлен на рис 4.10 - рис 4.11.



Свойство	Значение
IP адрес	192.168.10.121
SSH: логин	pi
SSH: пароль	123
Автозапуск	☑
Имя	PLC1
Номер	0
ОС	Linux RealLab! RSB v1
Пароль admin	<password>
Подключаться через	SSH
Пользовательское свойство 00	192.168.1.10
Пользовательское свойство 01	192.168.1.11
Порт отладчика	4840
Тип процессорной платы	Beckhoff CX1000
Версия изменений	69
Индекс	5
Принадлежит	1

Рис 4.10. – Свойства ПЛК 1

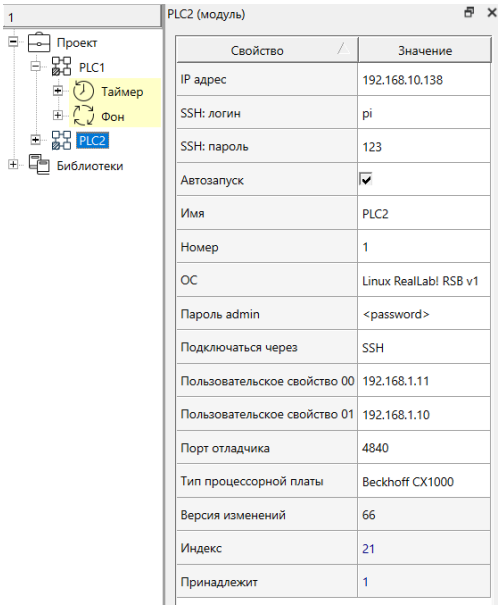


Рис 4.11. – Свойства ПЛК 2

4.5. Работа с протоколами

В проект-пример добавлены следующие модули:

- NLS-16DI;
- NLS-16DO;
- NLS-16DO-Ethernet;
- NLS-8AI-Ethernet.

На рис 4.12 – схема соединения резервируемых ПЛК, модулей и компьютера (клиента OPC UA).

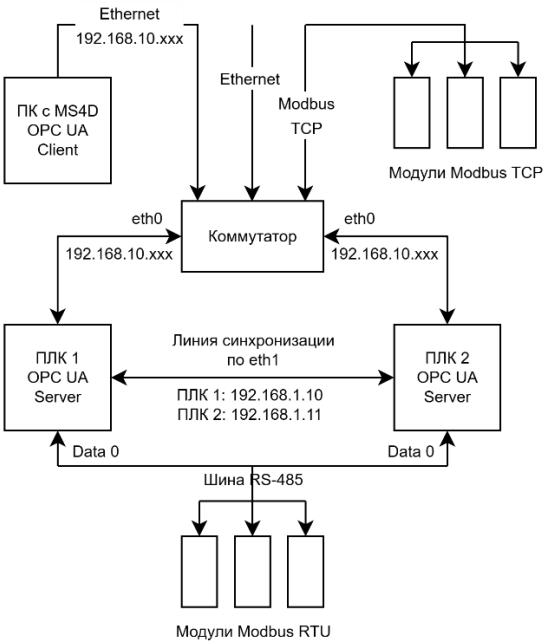


Рис 4.12. – Схема соединения ПЛК и модулей

4.5.1. Подключение по Modbus RTU

Настройка проекта одинакова как для топологии “**2 ПЛК, 1 общий набор модулей**”, так и для “**2 ПЛК, 2 индивидуальных набора модулей**”.

Для того, чтобы настроить резервирование модулей по протоколу Modbus RTU в Полигоне, необходимо выполнить следующее:

1. Создадим в модуле проекта поток для логики Modbus RTU, произвольно назовём его.
2. Создадим в созданном потоке программу для генератора импульсов, добавим туда блок “PulseGeneratorEx” из “Генераторы и таймеры” библиотеки “paCore” (рис 4.13).

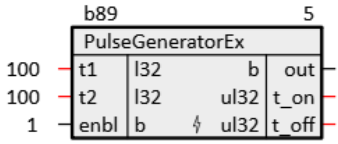


Рис 4.13. – Генератор импульсов

3. Создадим в созданном потоке программу на языке функциональных блоков (рис 4.14).

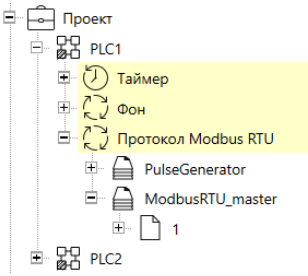


Рис 4.14. – Дерево проекта

4. Откроем страницу. Соединим с ПЛК модуль ввода-вывода RTU на примере NLS-16DO:

   - Определим скорость и id модуля. Для этого можно использовать конфигуратор NLConfig или другие инструменты - читать руководство пользователя конкретного модуля ввода/вывода.

   - Создадим блок “RealLabRS485” из библиотеки “paReallab”.

5. Выберем один из двух портов на входе rs485.

6. Установим скорость передачи на входе “speed”, такую же как на модуле.

   - Создадим блок “MRTU_Master” из библиотеки “paModbus”, создадим у него вход “inps” (по ПКМ).

   - Соединим выход и вход “cnc” созданных блоков друг с другом.

   - На вход “enbl” блока “MRTU_Master” задать глобальную ссылку на выход “L_Master”. Для этого необходимо кликнуть ПКМ по входу, нажать “задать глобальную.”, “ссылку на вход”, выбрать “L_Master”.

7. Таким образом, опрос модуля будет производиться только с ведущего ПЛК. Именно этот элемент обеспечивает резервирование опроса.

   - Создадим блок “NLS_16DO_DOwrite” из библиотеки “rlbModulesModbus”. Соединим его с блоком “MRTU_Master” через вход/выход “bo”. На вход “slv” установим id модуля.

   - Создадим блок “PulseGeneratorEx” из раздела “Генераторы и таймера” библиотеки “paCore” для генерации импульсов на выходы NLS-16DO. Установим входы “t1” и “t2” на 100(мс), вход enbl на 1. Соединим выход “out” с любым из выходов “NLS_16DO_DOwrite”.

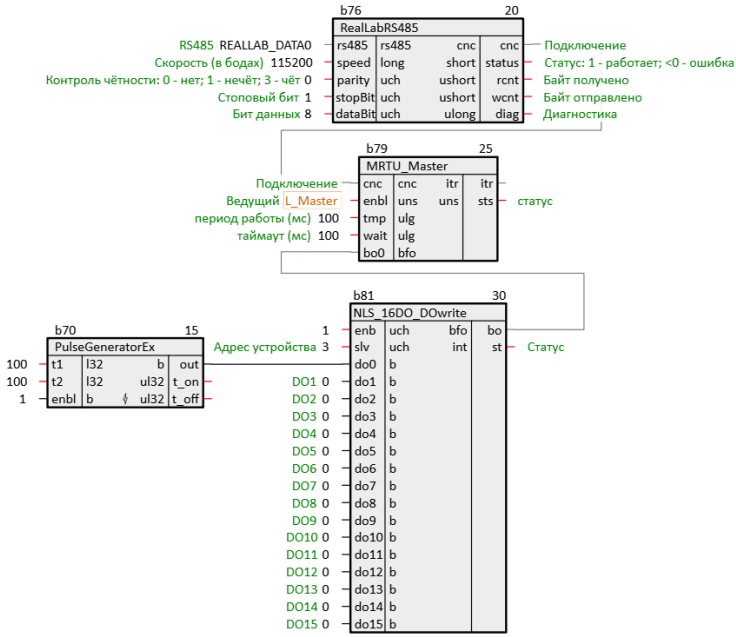


Рис 4.15. – Готовая программа Modbus RTU

4.5.2. Подключение по Modbus TCP

Для того, чтобы настроить резервирование модулей по протоколу Modbus TCP в Полигоне, необходимо выполнить следующее:

1. Создадим в модуле проекта поток, произвольно назовём его.
2. Добавим его в PLC2.
3. Создадим в созданном потоке программу для генератора импульсов, добавим туда блок “PulseGeneratorEx” из “Генераторы и таймеры” библиотеки “paCore” (рис 4.16).

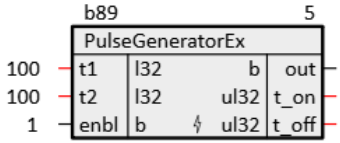


Рис 4.16. – Генератор импульсов

4. Создадим программу на языке функциональных блоков для логики Modbus TCP (рис 4.17).

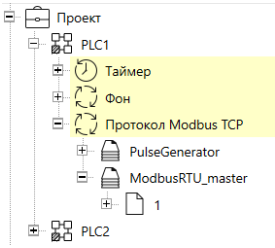


Рис 4.17. – Дерево проекта

5. Откроем страницу. Соединим с ПЛК модуль ввода-вывода TCP на примере NLS-16DO-Ethernet:

   - Присоединим модуль к той же локальной сети, что и ПЛК. Через ip сканнер найдём его IP, откроем веб-страницу по его IP-адресу и узнаем его ID. Это можно выполнить другими способами - читать руководство пользователя модуля.

   - Создадим блок “TcpIpCIA” из библиотеки “paCore”, раздел “Системные”.

6. Пропишем на входы IP-адреса ПЛК и модуля. IP-адрес ПЛК рекомендуется задавать как ссылку на свойства проекта:

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

Оба порта = 502.

   - Создадим блок “MTCP_Master” из библиотеки “paModbus”, создадим у него вход “inps” (по ПКМ).

   - Соединим выход и вход “cnc” созданных блоков друг с другом.

   - На вход “enbl” блока “MTCP_Master” задать глобальную ссылку на выход “L_Master”. Для этого необходимо кликнуть ПКМ по входу, нажать “задать глобальную..”, “ссылку на вход”, выбрать “L_Master”.

7. Таким образом, опрос модуля будет производиться только с ведущего ПЛК. Именно этот элемент обеспечивает резервирование опроса.

- Создадим блок “NLS_16DO_Enet_DOwrite” из библиотеки “rlbModulesModbus”. Соединим его с блоком “MTCP_Master” через вход/выход “bo”. На вход “slv” установим id модуля.
- Соединим выход “out” блока “PulseGeneratorEx” с любым из выходов “NLS_16DO_Enet_DOwrite”

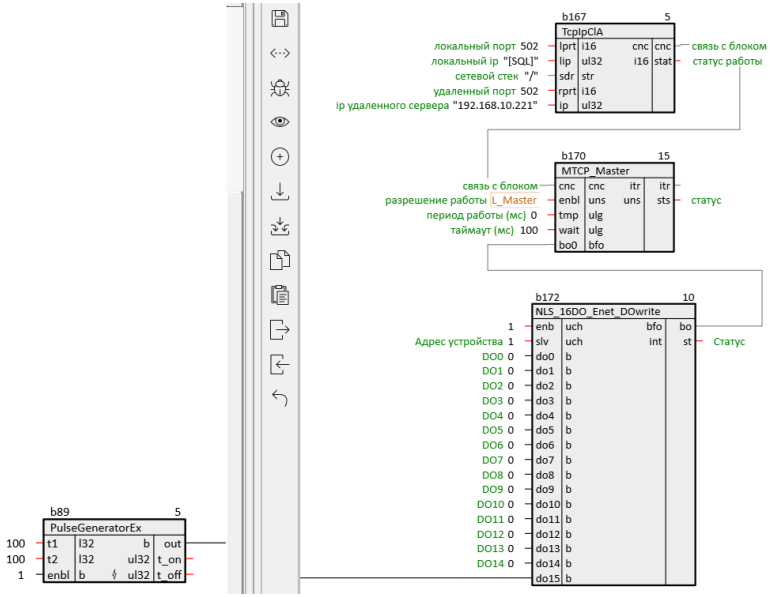


Рис 4.18. – Готовая программа Modbus TCP

4.5.1. Запуск проекта

Для запуска проекта загрузите 2 его модуля на каждый ПЛК по отдельности (рис 4.19).

Рис 4.19. – Панель запуска проекта

4.5.2. Синхронизация отладчика OPC UA

На каждом ПЛК (модуле проекта) работает свой OPC UA сервер со своим IP-адресом. Резервирование представлено в виде синхронизации отладчика.

То есть, при форсированном изменении переменной входа или выхода какого-либо функционального блока через OPC UA на любом из ПЛК, эта переменная так же изменится на другом ПЛК.

Количество синхронизируемых параметров ограничено (по умолчанию 100), число можно установить вручную – параметр “fnum” функционального блока “SyncMan”. При установке числа 0 синхронизация отключается.

К двум OPC UA серверам можно подключиться через среду MasterSCADA 4D для реализации визуализации с возможностью вывода переменных только с текущего ПЛК-мастера.

4.5.3. Синхронизация входов/выходов

Между двумя ПЛК возможна синхронизация входов-выходов функциональных блоков.

При такой синхронизации главным является значение на ведущем ПЛК. Оно форсировано дублируется на ведомый.

1. Установим в значение на входе fnum количество входов/выходов, которое собираемся синхронизировать. Можно взять значение с запасом (рис 4.20).

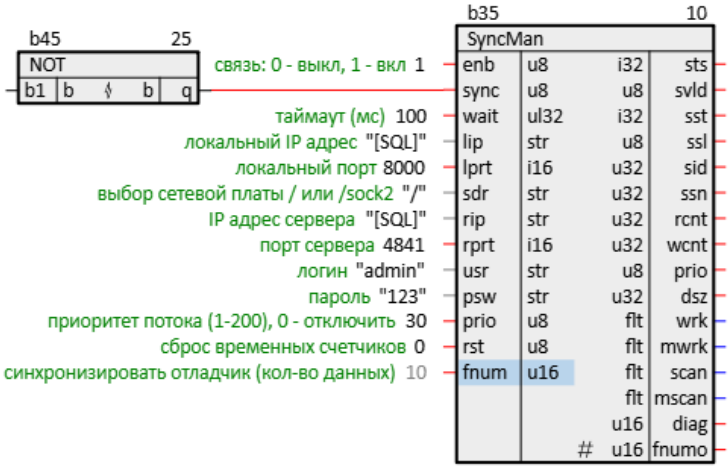


Рис 4.20. – Вход fnum

2. Зажав Ctrl на входе/выходе перенесём его на блок SuncMan и в контекстном меню выберем “добавить” (рис 4.21).

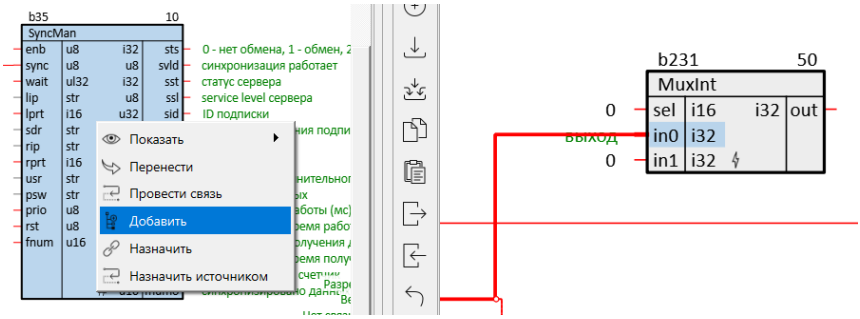


Рис 4.21. – Добавление входа в SyncMan

3. Теперь добавленный вход будет синхронизироваться.

4.5.4. Визуализация через MasterSCADA 4D

Также на компьютере – клиенте OPC UA реализована визуализация при помощи среды MasterSCADA (рис 4.22).

Для подключения к двум OPC UA серверам ПЛК с резервированием нужно в качестве сервера в настройках подключения установить 2 сервера через запятую: “opc.tcp://192.168.10.141,opc.tcp://192.168.10.145”.

Остальные настройки идентичны с настройками из примера “Обмен данными по протоколу OPC UA ПЛК NLScon-RSB-PLG”

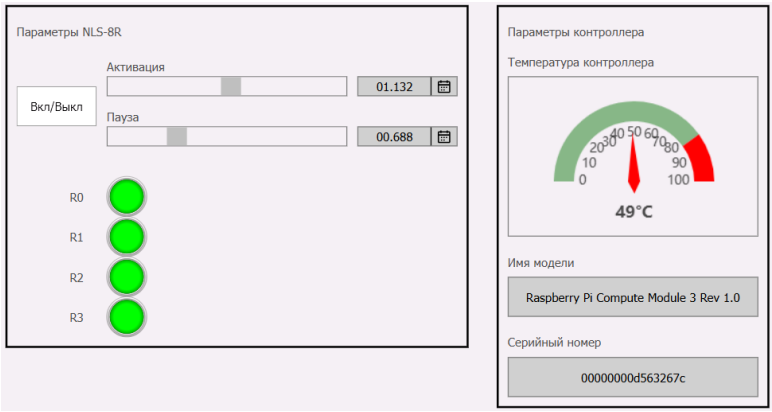


Рис 4.22. – Визуализация в проекте

5. Проверка и диагностика

Для проверки корректной работы горячего резервирования необходимо использовать существующий проект с настроенным горячим резервированием.

Исходное состояние: Резервируемые ПЛК соединены между собой линией синхронизации по Eth1, а каждый из них отдельно подключен к общей сети.

Отключение линии синхронизации: Отключите линию синхронизации между ведущим и ведомым ПЛК (рис 5.1). В результате оба ПЛК перейдут в режим автономного ведущего. Светодиоды индикации состояния на обоих устройствах начнут мигать зелёным светом.

Восстановление линии синхронизации: Восстановите соединение линии синхронизации. ПЛК автоматически проведут процедуру синхронизации, после чего восстановится исходное распределение ролей: один ПЛК вернётся в состояние «Ведущий», второй — в состояние «Ведомый». Светодиоды индикации вернутся к нормальному режиму работы.

Отключение питания, ведущего ПЛК: Отключите питание от ПЛК, находящегося в роли «Ведущий» (рис 5.2). Ведомый ПЛК должен автоматически переключиться в роль «Ведущий» и продолжить корректный опрос модулей ввода-вывода без прерывания процесса управления.

После этого восстановите питание ранее отключённого ПЛК. Дождитесь его полного запуска и завершения синхронизации с текущим ведущим ПЛК. После успешной синхронизации отключённый ПЛК автоматически перейдёт в роль «Ведомый».

Отключение питания, ведомого ПЛК: Отключите питание от ПЛК, находящегося в роли «Ведомый» (рис 5.3). Ведущий ПЛК не должен изменять своё состояние и продолжает штатную работу — опрос модулей ввода-вывода и управление процессом без прерываний.

После восстановления питания и завершения процедуры синхронизации отключённый ПЛК автоматически вернётся в роль «Ведомый».

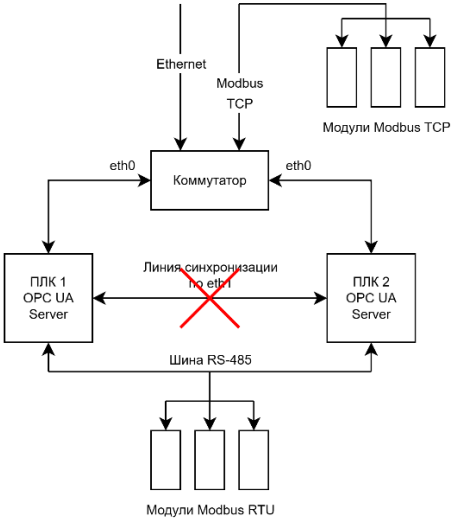


Рис 5.1. – Обрыв линии синхронизации

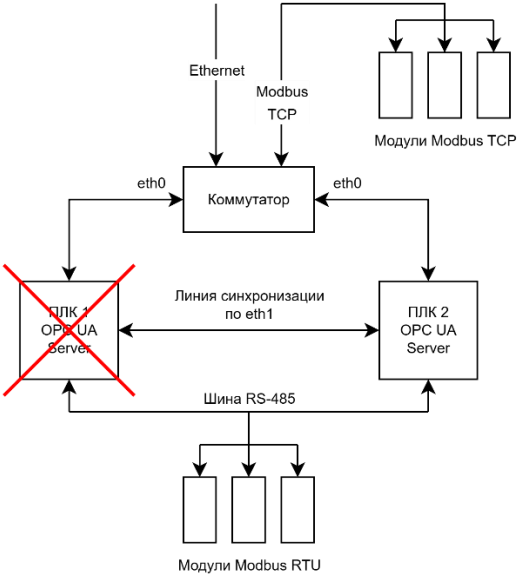


Рис 5.2. – Отключение ПЛК 1 (ведущего)

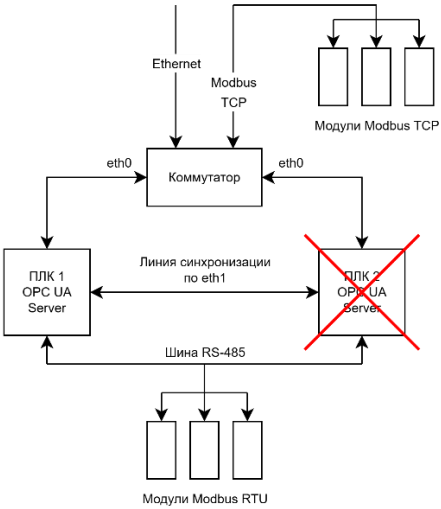


Рис 5.3. – Отключение ПЛК 2 (ведомого)

5.1. Переключатель Run-Stop

Переключатель Run-Stop, останавливает/запускает исполняемый файл полигон-проекта. При переводе переключателя в режим STOP – исполнение программы на ПЛК приостанавливается, резервный контроллер переходит в состояние Master.

При запуске ПЛК программа автозапуска проекта Полигон проверяет состояние переключателя. Если переключатель в положении Старт, проект запустится.

При отключения основного питания, ПЛК продолжает работать 10 секунд от аккумулятора (время можно настроить). Затем выключается. Отключение питания можно отследить в проекте и произвести необходимые действия: сохранить retain-переменные.

5.2. Мониторинг ролей и состояния резервирования через индикацию на корпусе ПЛК

На панели индикации на ПЛК находится светодиод “состояние”, отображающий текущую роль ПЛК при резервировании (при условии добавления в проект функционального блока RedundancyStateCtrl) (табл. 6).

Табл. 6 – Работа светодиодной индикации ПЛК при резервировании

Состояние ПЛК	Индикация
ПЛК-мастер	Постоянный зеленый
ПЛК-слейв	Постоянный красный
Автономный режим (нет связи со вторым ПЛК)	Мигающий зеленый
Аварийное состояние (включение осуществляется пользователем подачей значения “True” на вход “error” ФБ “RedundancyStateCtrl”)	Постоянный оранжевый

5.3. Мониторинг ролей и состояния резервирования через отладчик Полигона (OPC UA)

Отследить состояние резервирования можно подключившись к проекту отладчиком и найдя в проекте блок “MasterSel”.

Для переключения отладчика между ПЛК можно нажать ПКМ на пустом месте страницы -> показать -> выбрать другой ПЛК (рис 5.4).

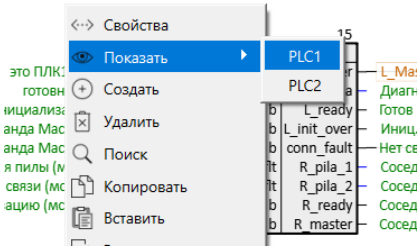


Рис 5.4. – Выбор другого ПЛК на странице при отладке

Выход L_Master сигнализирует о роли ПЛК в резервировании: 1 – ведущий, 0 – ведомый.

Выход conn_fault сигнализирует об отсутствии связи с соседним ПЛК: 1 – связи нет, 0 – связь есть.

Точно так же можно использовать любой OPC UA клиент. Например, MasterSCADA. Необходимо лишь дополнительно найти блок MasterSel по его названию (рис 5.5)

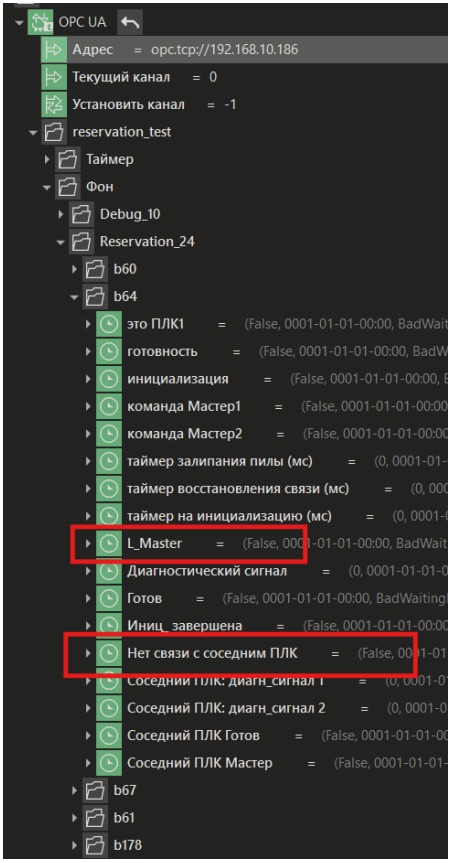


Рис 5.5. – Просмотр роли и состояния ПЛК через OPC UA клиент MasterSCADA

Лист регистрации изменений

Дата изменения	Описание изменения	Примечание