КОНФИГУРИРОВАНИЕ РЕЗЕРВИРОВАННОЙ СИСТЕМЫ НА КОНТРОЛЛЕРАХ СЕРИИ REGUL RX00

Руководство пользователя

DPA-302.4

Версия документа 2.8

Версия ПО 1.7.1.0

Август 2023

История изменений руководства пользователя

Версия руководства пользователя	Описание изменения
2.2	Добавлена история изменений руководства пользователя. Добавлены знаки с предупреждающей и поясняющей информацией. *Раздел «Алгоритм работы задачи резервирования»:* добавлены значения в таблицу зависимости SyncTime от DataSizeKb для разных типов модулей ЦП. *Раздел «Резервируемые переменные и программирование контроллера»:* добавлено описание ограничений в резервировании на этапе разработки. Добавлены новые разделы: – *«Особенности резервирования ПЛК с модулями ЦП II-го типа»;* – *«Обращение в службу технической поддержки»;* – *«Приложение А».* *Раздел «Журналы сообщений»:* обновлено описание вкладки *Сообщения* в редакторе. *Раздел «Обновление приложений резервируемого контроллера»:* добавлено описание ограничения при загрузке проекта на ведущий контроллер. Дополнительно по тексту внесены небольшие изменения с уточняющей информацией
2.3	*Раздел «Настройка IP-адресов для резервирования»:* обновлено описание настройки параметров IP-адресов. Добавлены новые разделы: – *«Конфигурация коммуникационных модулей Ethernet в резервированной системе»;* – *«Замена модуля ЦП резервированного контроллера».* *Раздел «Алгоритм работы задачи резервирования»:* добавлены значения приоритетов задач для разных типов модулей ЦП. *Раздел «Скорость реакции на аварийные события»:* обновлены формулы расчета таймаута модуля и мастера при активированном параметре «Автонастройка параметров». Дополнительно по тексту внесены небольшие изменения с уточняющей информацией
2.4	*Раздел «Добавление в проект компонента «Резервирование»»:* дополнено описание о доработке настроек задачи резервирования при создании конфигурации. Добавлен новый раздел: *«Конфигурация коммуникационных модулей RS-485 в резервированной системе».* *Раздел «Настройка и диагностика соединения»:* добавлено описание значков, информирующих о состоянии устройств. Отображение пассивного состояния модуля ЦП-партнера в онлайн режиме. *Раздел «Настройка резервирования»:* обновлено описание состояний, в которых может находиться соответствующий ЦП. Дополнительно по тексту внесены небольшие изменения с уточняющей информацией

Версия руководства пользователя	Описание изменения
2.5	Обновление в связи с выпуском среды разработки Astra.IDE. Добавлен новый раздел: «*Настройка резервирования с версии СПО 1.7».* *Раздел «Принципы резервирования Алгоритм работы задачи резервирования»:* добавлено описание об увеличении максимального размера синхронизируемых данных (до 500 КБ) для I-го типа ЦП. Дополнительно по тексту внесены небольшие изменения с уточняющей информацией
2.6	*Раздел «Скорость реакции на аварийные события»:* обновлены формулы расчета таймаута модуля и мастера при активированном параметре «Автонастройка параметров». *Приложение А «Типы модулей центрального процессора»:* добавлены новые модули. Добавлен новый раздел: *«Приложение Б. Библиотеки».* Дополнительно по тексту внесены небольшие изменения с уточняющей информацией
2.7	Добавлен новый раздел: «*Термины и определения».* Дополнительно по тексту внесены небольшие изменения с уточняющей информацией
2.8	*Приложение Б «Библиотеки»:* добавлена новая функция Synchronize4, взамен устаревшей Synchronize3 и исключен флаг xDenyBeActive (принудительный запрет быть ведущим)

АННОТАЦИЯ

Настоящий документ содержит сведения о создании и конфигурировании резервированных систем на основе программируемых логических контроллеров серии Regul RX00. Настройка осуществляется с помощью прикладного программного обеспечения Astra.IDE.

Данное руководство предназначено для эксплуатационного персонала и инженеров-проектировщиков АСУ ТП, которые должны:

- иметь, как минимум, среднее техническое образование;
- приступить к работе только после изучения данного руководства.

Обновление информации в Руководстве

Производитель ООО «РегЛаб» оставляет за собой право изменять информацию в настоящем Руководстве и обязуется публиковать более новые версии с внесенными изменениями. Обновленная версия Руководства доступна для скачивания на официальном сайте Производителя: https://reglab.ru/.

Для своевременного отслеживания выхода новой версии Руководства рекомендуется оформить подписку на обновление документа. Для этого необходимо на сайте Производителя: https://reglab.ru/ кликнуть на кнопку «Подписаться на обновления» и оставить свои контактные данные.

В руководстве присутствуют знаки с предупреждающей и поясняющей информацией. Каждый знак обозначает следующее:

ПРЕДУПРЕЖДАЮЩИЕ ЗНАКИ

	**ВНИМАНИЕ!** Здесь следует обратить внимание на способы и приемы, которые необходимо в точности выполнять во избежание ошибок при эксплуатации или настройке

ИНФОРМАЦИОННЫЕ ЗНАКИ

	**ИНФОРМАЦИЯ** Здесь следует обратить внимание на важную информацию

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Использование функции «горячего» резервирования позволяет обеспечить непрерывное выполнение технологического процесса в случае какого-либо отказа в системе управления в общем и в контроллере – в частности.

В резервированном контроллере используются два модуля центрального процессора (далее ЦП), называемые ЦП-партнерами, которые работают по одинаковому алгоритму и постоянно синхронизируют свои данные друг с другом.

Обмен данными между модулем ЦП и модулями ввода/вывода происходит по одной линии (шине RegulBus), а синхронизация данных между ЦП-партнерами осуществляется по другой линии (линия синхронизации), независимой от первой. При этом каждый модуль ЦП общается и производит обмен с модулями ввода/вывода независимо от другого модуля ЦП по своей собственной шине RegulBus.

i	**ИНФОРМАЦИЯ** Начиная с версии СПО 1.7.0.0, доступна нативная версия внутренней шины данных контроллера (**RegulBus OS**), реализованная на уровне операционной системы. Описание перехода на другую версию шины приведено в документе «Программное обеспечение Astra.IDE. Руководство пользователя» в разделе «Редактор шины»)

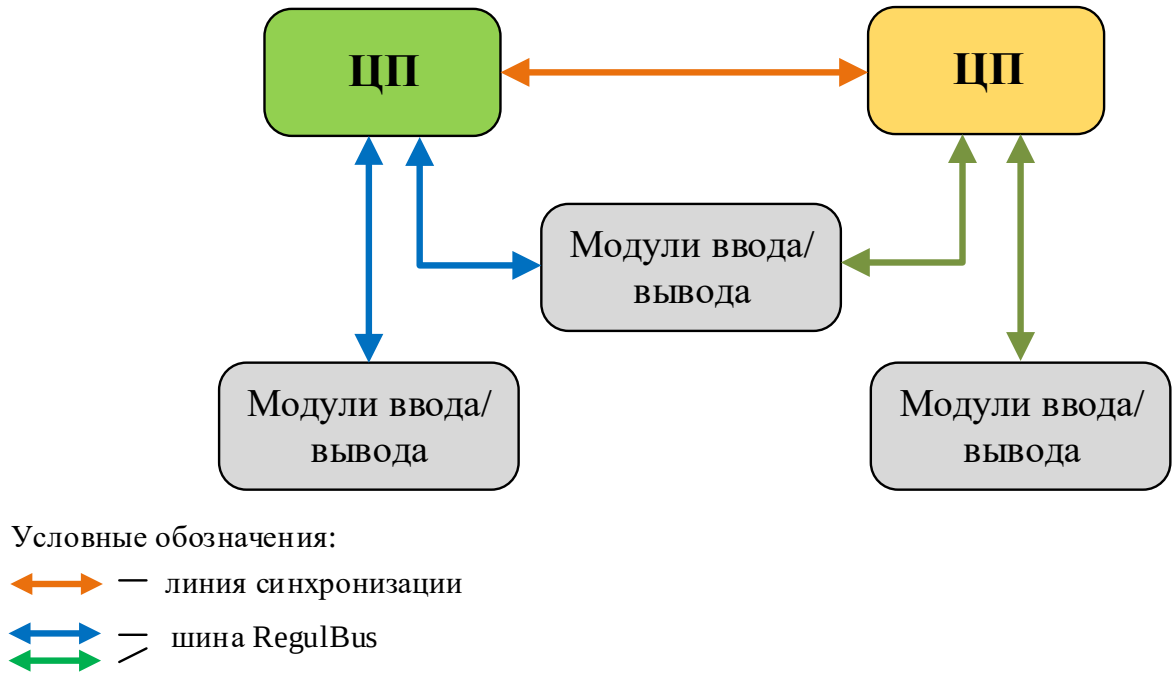

Рисунок 1 – Обмен данными по разным шинам

Термины и определения

Программируемый логический контроллер (ПЛК) — микропроцессорное устройство в промышленном исполнении, используемое для сбора, преобразования, обработки, хранения информации и выработки команд управления, имеющее конечное количество входов и выходов, подключенных к ним датчиков, ключей, исполнительных механизмов к объекту управления, и предназначенное для работы в режимах реального времени.

Astra.IDE - среда разработки с набором компонент для настройки/программирования контроллеров серии Regul RX00, выпускается компанией «РегЛаб».

Крейт – обособленная сборка из нескольких модулей, объединенных в общем корпусе или на DIN-рейке, с независимым электропитанием.

CPU – центральное процессорное устройство (ЦПУ), также модуль центрального процессора (ЦП) программируемого логического контроллера REGUL RX00. В резервированном контроллере используются два модуля центрального процессора, называемые ЦП-партнерами (**CPU A и CPU B**).

Проект – совокупность программного кода и настроек устройств, необходимая для выполнения контроллером своих функций.

POU – компонент организации программы. В большинстве случаев под этим термином понимается пользовательская программа и функциональный блок.

Устройство - специализированное аппаратное средство, на котором должна запускаться ПЛК-программа (приложение).

Приложение – это набор программных объектов, необходимых для запуска конкретного экземпляра пользовательской программы на конкретном устройстве.

Задача – часть приложения, выполняющая блок подпрограмм в отдельном цикле контроллера. Для каждой задачи можно определить контроль времени выполнения.

Система исполнения – это часть системного программного обеспечения контроллера, предназначенное для выполнения на контроллере пользовательской программы. Устанавливается в контроллер в процессе его изготовления.

Плагины – модули, подключаемые к среде разработки Astra.IDE, обеспечивающие настройку и конфигурирование контроллеров серии Regul RX00. Каждый плагин имеет имя, версию и ограничения версии.

Профиль – набор плагинов конкретных версий, которые будут подгружены к среде разработки в момент запуска.

АППАРАТНАЯ РЕАЛИЗАЦИЯ РЕЗЕРВИРОВАНИЯ

Схемы резервирования

Обмен данными **между центральными процессорами** организован посредством линии синхронизации. Для построения систем резервирования используются модули ЦП с двумя и более портами Ethernet. Один порт для построения линии синхронизации, а второй порт для подключения к контроллеру из среды Astra.IDE.

Резервирование линии синхронизации

Для линии синхронизации предусмотрена возможность резервирования, т.е. использование дублирующей линии на случай выхода из строя основной. При дублировании линии синхронизации образуется два физических соединения между модулями ЦП, обеспечивая тем самым взаимодействие ЦП в штатном режиме даже при обрыве одной линии связи.

Для линий синхронизации можно использовать два любых (одинаковых по номеру) порта Ethernet на борту каждого модуля ЦП (Рисунок 2).

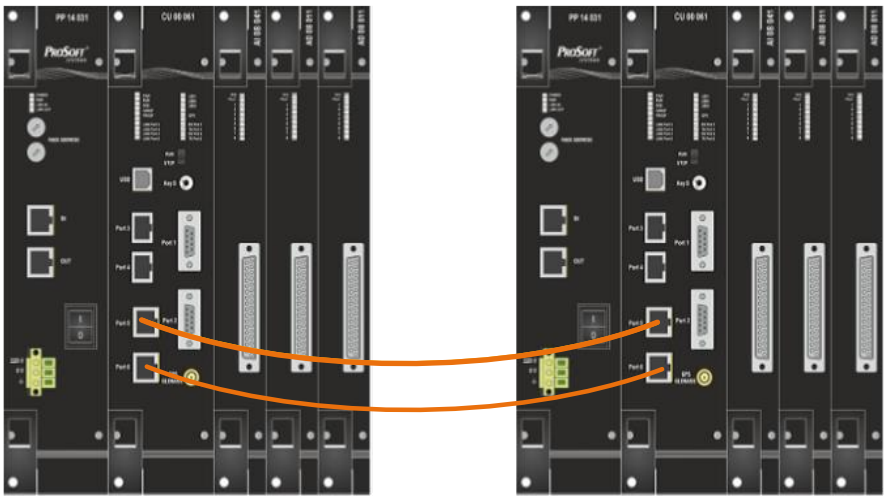

Рисунок 2 – Пример резервированной линии связи на контроллерах R600

Резервирование модулей контроллера

Предусмотрены следующие схемы резервирования модулей:

- **полное резервирование** – в схеме присутствуют два идентичных набора крейтов, объединенных в один резервированный контроллер. Каждый базовый крейт имеет модуль ЦП со своим собственным набором модулей ввода/вывода, представляя собой «зеркало» другого;
- **частичное резервирование** – в схеме резервирования два модуля ЦП работают с одним общим набором модулей ввода/вывода;
- **комбинированная схема резервирования** – в схеме часть модулей ввода/вывода дублируется и работает только с одним модулем ЦП, как в случае полного

резервирования, а часть присутствует в единичном экземпляре и осуществляют обмен данными с обоими модулями ЦП.

Условные обозначения:

— линия синхронизации
— первая шина RegulBus
— вторая шина RegulBus

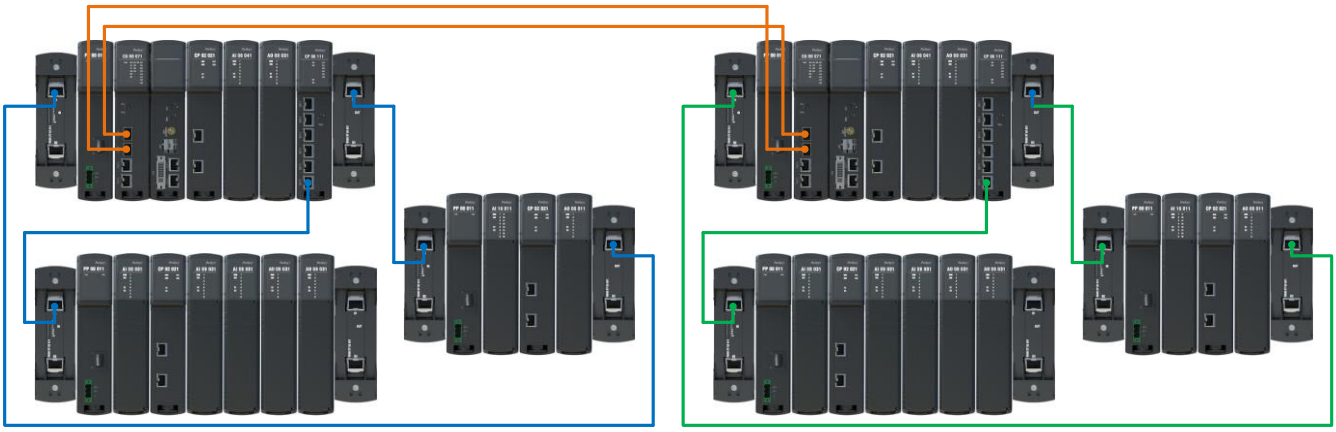

Рисунок 3 – Пример схемы полного резервирования, Regul R500

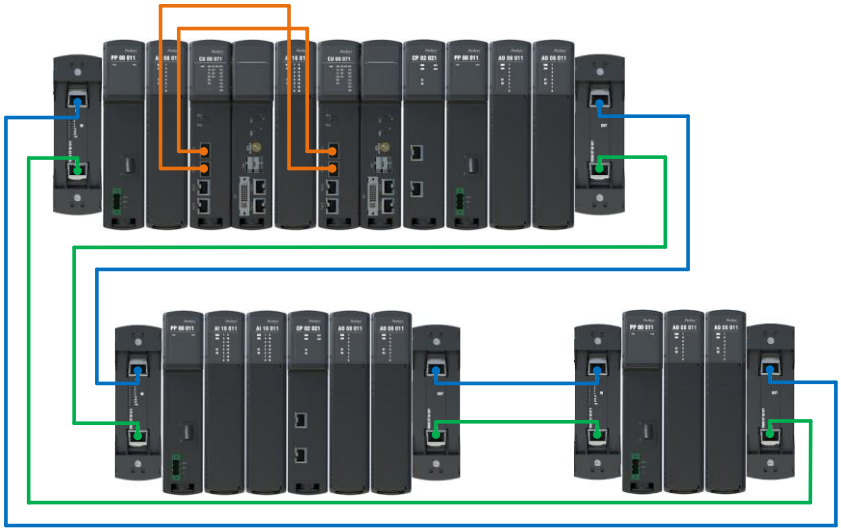

Рисунок 4 – Пример схемы частичного резервирования, Regul R500

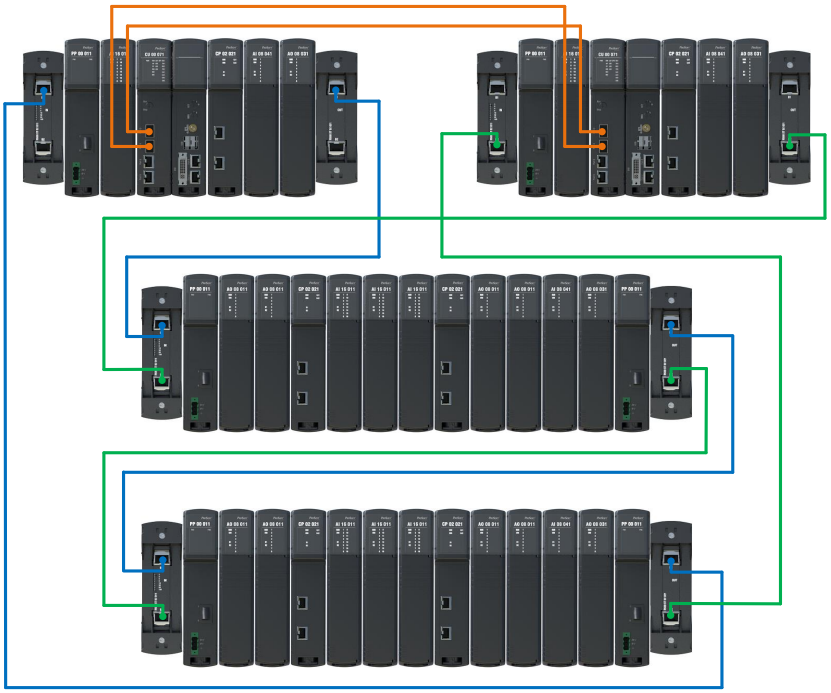

Рисунок 5 – Пример комбинированной схемы резервирования, Regul R500

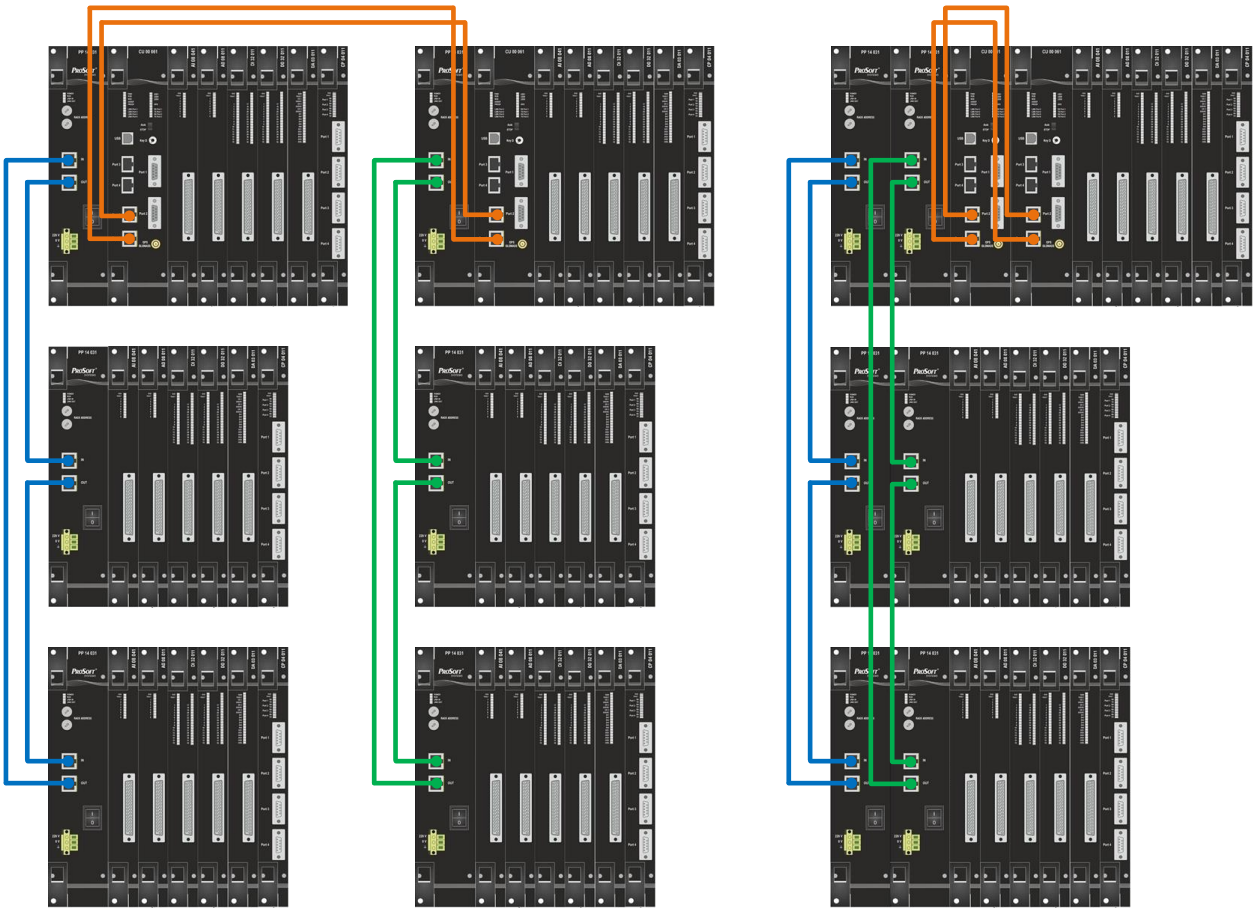

Рисунок 6 – Примеры схем полного и частичного резервирования, Regul R600

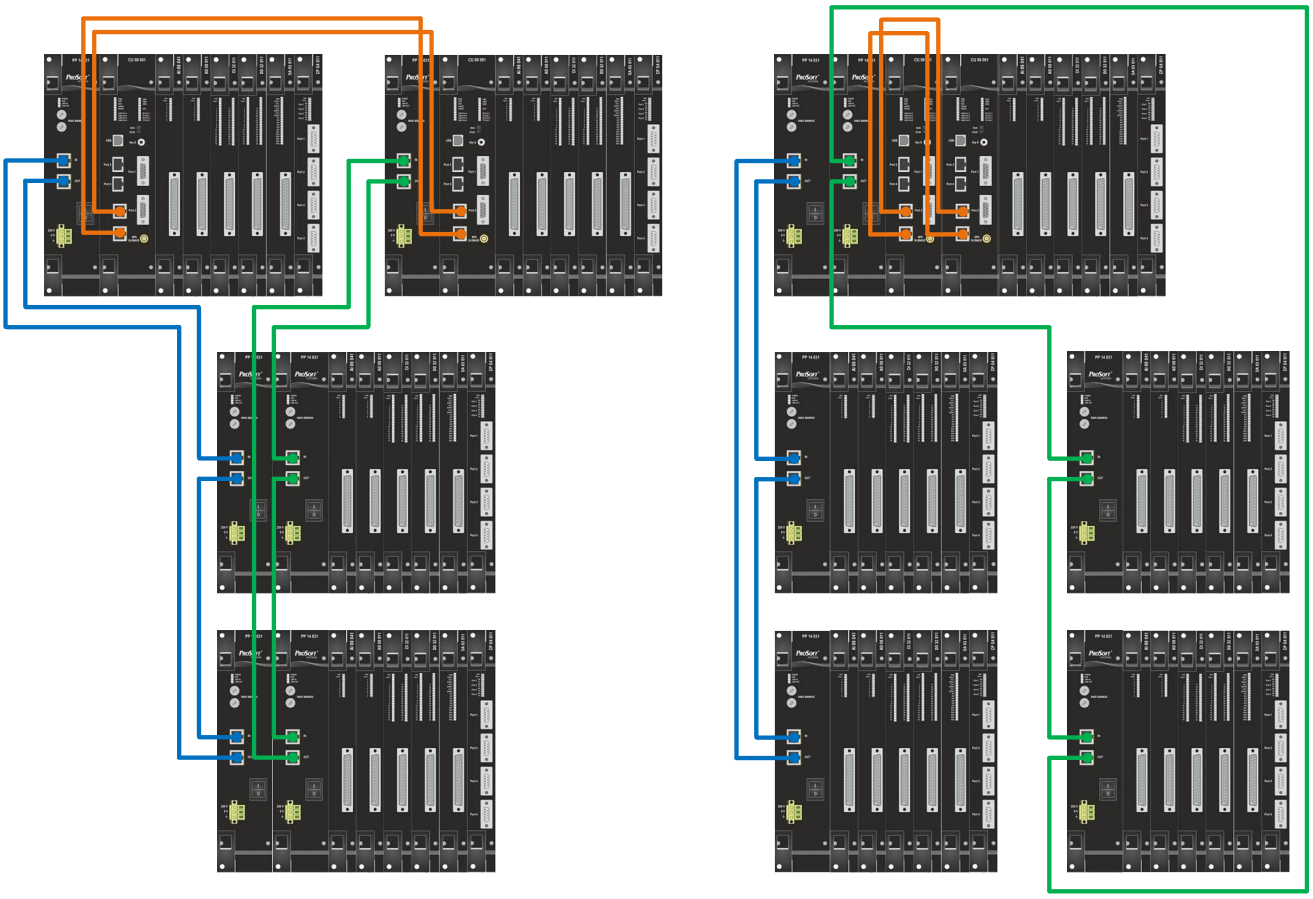

Рисунок 7 – Примеры комбинированных схем резервирования, Regul R600

При использовании частичной или комбинированной схемы резервирования в контроллере присутствуют так называемые общие модули – это модули ввода/вывода, которые одновременно работают с двумя модулями ЦП. При чем, каждый из двух модулей ЦП осуществляет обмен с общими модулями ввода / вывода по своей шине RegulBus, независимой от шины RegulBus ЦП–партнера.

В контроллере R500 выбор номера шины RegulBus, по которой модуль ЦП осуществляет связь с модулями ввода/вывода, производится переключателем MBS (см. главу Переключатели).

В контроллере R600 аналогичный выбор определяется местом установки модуля ЦП и модуля источника питания. При этом, в случае установки в одном крейте, модуль ЦП, осуществляющий связь с модулями ввода/вывода по первой шине RegulBus, устанавливается левее от модуля ЦП, осуществляющего связь с модулями ввода/вывода по второй шине RegulBus (см. Рисунки 6, 7). Аналогичное правило действует и для модулей источника питания.

При реализации схемы резервирования возможно использование модулей ввода/вывода из различных линеек REGUL: R600, R500, R200 (модули R200 только в схемах с полным резервированием). В качестве модулей ЦП могут использоваться процессорные модули R200, R500, R600.

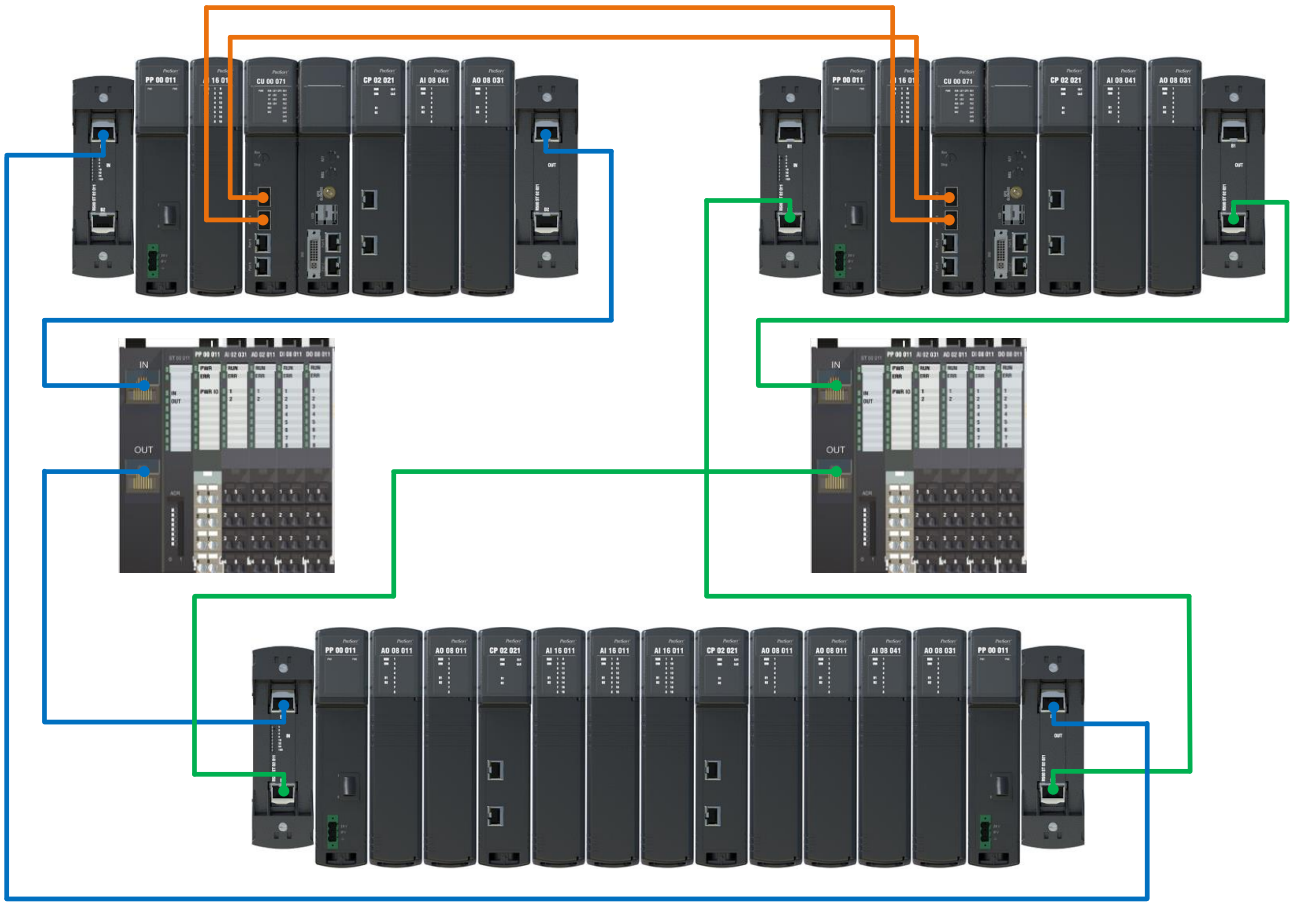

Рисунок 8 – Комбинированная схема, использующая модули ввода/вывода из линейки R200, R500 (общий) и базовый крейт R500

Модули ЦП делятся на типы (I/II/III). Для оценки функциональных и вычислительных возможностей резервированной системы следует учитывать тип применяемого модуля ЦП. Распределение по типам модулей ЦП серии REGUL RX00 представлено в Приложении А.

Переключатели и индикация

Переключатели

Переключатели расположены на передней панели модуля центрального процессора.

Модуль ЦП может находиться в одном из двух режимов: *Работа* (RUN) или *Стоп* (STOP). Переключение между режимами работы ЦП – *Работа*/*Стоп* производится переключателем RUN / STOP. Также запуск / остановку приложения можно производить с помощью кнопок Старт / Стоп (Рисунок 9) в среде разработки прикладного программного обеспечения Astra.IDE (подробно описание в документе «Программное обеспечение Astra.IDE. Руководство пользователя» в разделе «Запуск и мониторинг приложения»).

Рисунок 9 – Кнопки Старт / Стоп в среде разработки прикладного программного обеспечения Astra.IDE

В случае необходимости пользователь может запретить управление режимами работы из Astra.IDE. Для этого в секции PsLed конфигурационного файла */etc/runtime.cfg* необходимо активировать опцию:

```
[PsLed] RunStopButtonSuperior = 1
```

Переключатель KEY управляет автозапуском прикладной программы. В модуле R600 переключатель обозначен как Key D, в R500 и R200 – KEY. Положение I – автозапуск выключен, II – включен. Положение III, имеющееся в модуле R500 и R200, не используется.

В модуле R500 переключатель MBS определяет, мастером какой из двух шин RegulBus является модуль. Он имеет два положения:

- в положении I центральный процессор обеспечивает обмен данными между модулями по первой шине RegulBus;
- в положении II – по второй шине RegulBus.

Индикация резервированного контроллера

Описание всех индикаторов светодиодной панели модулей контроллера приведено в документах «Regul R200. Системное руководство», «Regul R500. Системное руководство», «Regul R600. Системное руководство».

Ниже приведено описание тех индикаторов, которые помогают пользователю ориентироваться в работе резервированного контроллера (Рисунок 10).

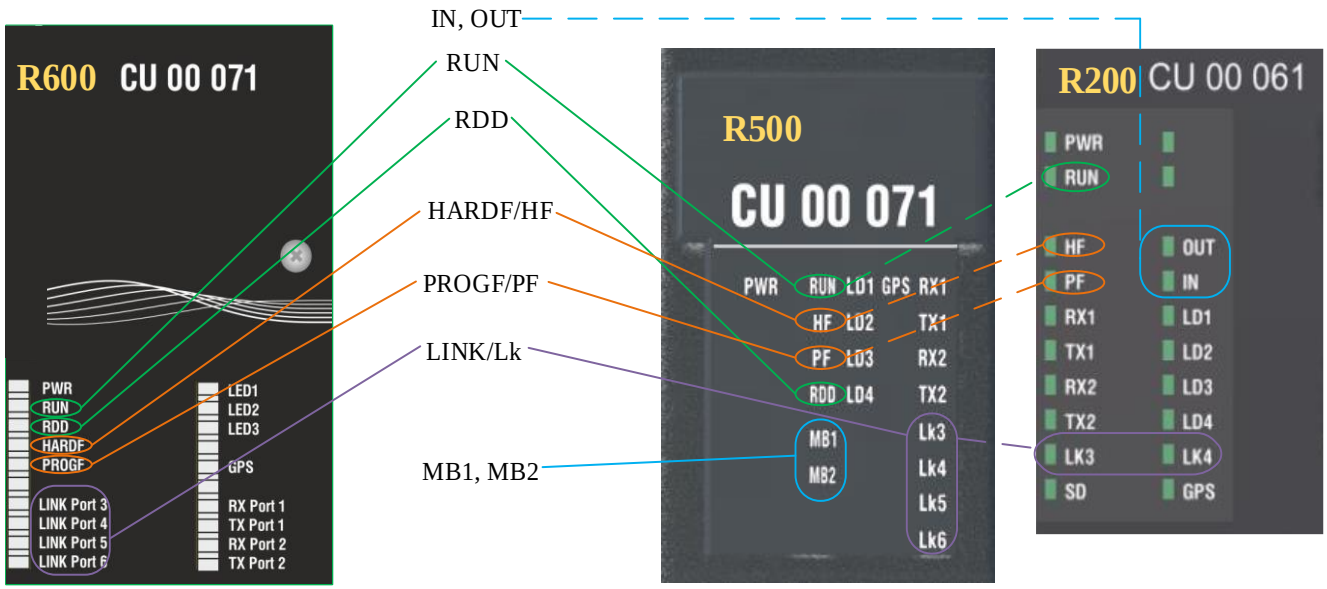

Рисунок 10 – Фрагменты лицевых панелей модулей ЦП R600, R500 и R200. Индикаторы

RUN – индикатор горит при выполнении прикладной программы в центральном процессоре, если не горит, то пользовательская программа не выполняется (не загружена или переключатель RUN/STOP находится в положении STOP).

RDD (R600 и R500) – индикатор информирует о работе модуля в составе резервированного контроллера:

- горит при работе модуля в качестве ведущего центрального процессора в составе резервированного контроллера,
- редко мигает (1 Гц) – при работе модуля в качестве ведомого центрального процессора в составе резервированного контроллера,
- часто мигает (5 Гц) – любое другое состояние модуля ЦП, которое показывает о логической неготовности данного ЦП к управлению (например, идет обновление проектов, или ожидается синхронизация, проекта нет или он остановлен и т.д.).

HARDF (R600), **HF** (R500 и R200) – индикатор горит в случае отсутствия или неисправности одного из модулей контроллера (неисправность в шине).

PROGF (R600), **PF** (R500 и R200) – индикатор:

- горит - присутствует программная ошибка в модуле;
- медленно моргает (1 Гц) - пользовательская программа не загружена.

LINK Port X, (R600), **Lk X** (R500 и R200), где X – номер порта (3-6) – индикатор мигает при наличии обмена через соответствующие Ethernet-порты.

MB1 и **MB2** (R500) – индикаторы показывают, мастером какой из шин RegulBus в данный момент является этот модуль (свечение зеленым цветом).

IN и **OUT** (R200) – индикаторы мигают при наличии обмена через порт IN и OUT соответственно;

B1 и **B2** (модули ввода/вывода R500) – индикаторы показывают состояние обмена данными по шине:

- не горит – с момента включения и по настоящие время обмен по шине не производился и модуль на ней не инициализировался,
- мигает желтым – идет инициализация модуля по данной шине,
- горит желтым – модуль сконфигурирован и осуществляет обмен по шине, но не выдает на выходные клеммы сигналы, соответствующие данным, полученным по ней (шина, как и центральный процессор, подключенный к ней, находятся в резерве),
- горит зеленым – модуль сконфигурирован и осуществляет обмен по данной шине, а в соответствии с данными, полученными по ней, подаются сигналы на выходные клеммы модуля (шина, как и центральный процессор, подключенный к ней, являются ведущими),
- мигает красным – несоответствие типа модуля конфигурации шины,
- горит красным – с момента включения по шине производился обмен, но впоследствии связь по ней была утрачена.

ПРИНЦИПЫ РЕЗЕРВИРОВАНИЯ

Понятия ведущего и ведомого модуля ЦП

В резервированном контроллере в работе одновременно находятся два модуля ЦП. При этом один модуль ЦП является **ведущим** (Active) – он непосредственно обеспечивает управление технологическим процессом, а его партнер – **ведомым** (StandBy) – он находится в «горячем» резерве.

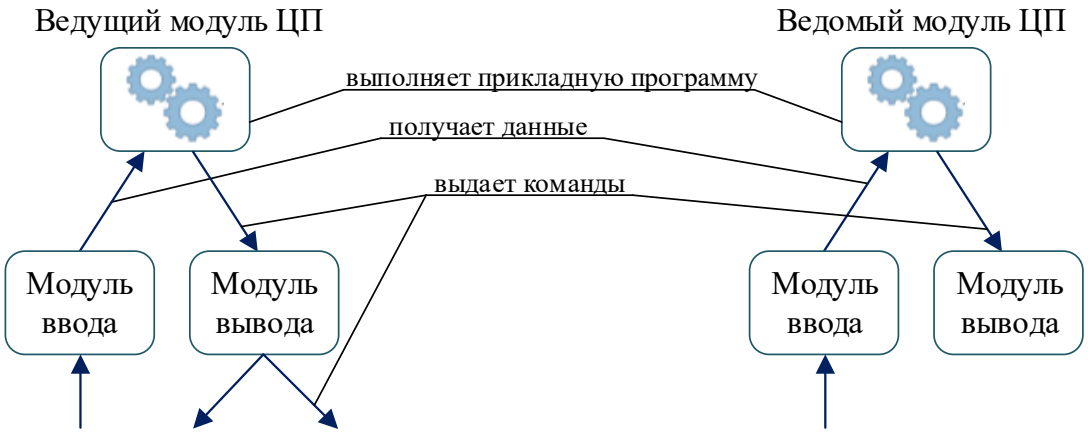

Рисунок 11 – Одновременная работа модулей ЦП

Оба модуля ЦП производят опрос модулей ввода, выполнение прикладной программы пользователя и выдачу результатов этого выполнения в модули вывода.

Модули вывода формируют управляющие команды, только если они поступили от ведущего модуля ЦП, при этом:

- если модуль вывода общий (работает с двумя модулями ЦП), он формирует на выход сигналы только в соответствии с данными от ведущего модуля ЦП,
- если модуль вывода не является общим (работает с одним модулем ЦП), он формирует сигналы на выход, только если этот модуль ЦП ведущий. Если модуль ЦП ведомый, то модуль вывода не устанавливает выхода.

Роли модулей ЦП в составе резервированного контроллера

С целью исключения конфликтов приоритет одного модуля ЦП (CPU_A) по умолчанию выше приоритета ЦП–партнера (CPU_B). Поэтому, в случае нахождения обоих ЦП в один и тот же момент времени в одном и том же состоянии (ведущий или ведомый), управление на себя принимает CPU_A.

Выбор роли модуля ЦП производится на этапе настройки IP-адресов для линий синхронизации. При конфигурировании порта связи необходимо выбрать, к какому из ЦП-партнеров относится данный порт, при этом, после определения, происходит автоматическая установка IP-адресов портов. Стоит отметить, что в схемах частичного или комбинированного

резервирования CPU_A должен работать по первой шине RegulBus, CPU_B – по второй шине RegulBus.

Алгоритм работы задачи резервирования

Все задачи реализуются с вытесняющей многоприоритетной многозадачностью. Значение приоритета задачи резервирования задается в зависимости от типа применяемого модуля ЦП:

- для модуля ЦП типа I / III– приоритет 1;
- для модуля ЦП типа II – приоритет 4.

В самом цикле задачи резервирования существует последовательность системных и пользовательских вызовов.

В начале каждого цикла вызывается системная процедура **Read Inputs** – чтение данных с входных каналов модулей ввода/вывода и данные, полученные по цифровой линии связи (Рисунок 12).

Далее выполняется синхронизация данных (вызов функции **Synchronize**). При синхронизации происходит копирование данных между ЦП–партнерами, коррекция интервала вызова задачи на ведомом модуле ЦП. По завершении этих действий выполнение задач приостанавливается до окончания контрольного времени синхронизации.

По истечении времени синхронизации выполняется пользовательское приложение (**User application**).

Сразу после окончания выполнения пользовательского приложения вызывается системная процедура **Write Outputs** – запись обновленных значений переменных в каналы вывода. Это могут быть управляющие сигналы для модулей DO/AO либо данные, передаваемые по цифровой линии связи.

Задача на **ведомом** модуле ЦП, участвующая в резервировании, синхронизируется с одноименной задачей **ведущего** модуля ЦП. В рамках одного резервированного контроллера может быть синхронизирована только одна задача.

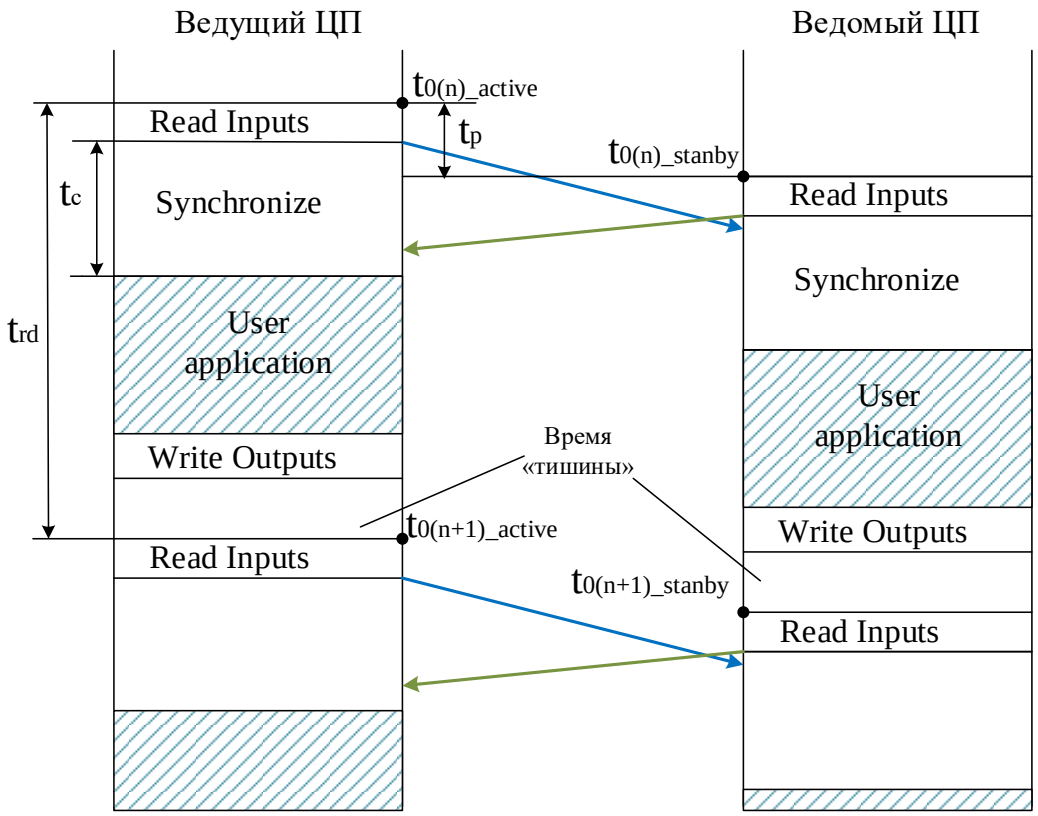

Рисунок 12 – Алгоритм работы задачи резервирования

t_0 – время начала очередного цикла задачи резервирования. Время различается для **ведущего** и **ведомого** модуля ЦП из-за погрешности планировщика задач, разница значений может составлять несколько миллисекунд.

t_{rd} – время цикла задачи резервирования, значение не должно превышать 2500 мс.

t_p – время рассинхронизации задач, т.е. смещения ведомой задачи от ведущей. Данный параметр по умолчанию установлен равным 25% от времени цикла задачи, но может быть изменен пользователем.

t_c – время синхронизации. По умолчанию рассчитывается автоматически, но может быть изменено пользователем, исходя из объема резервируемых данных (общий объем блоков данных, заданных в экземплярах **SharedMemory** и **CrossMemory**) (Таблица 1). При этом время синхронизации должно быть не более 75% от времени цикла. Кроме того, если пользователь установит значение **t_c** меньшее, чем рассчитано по умолчанию, это может привести к ошибкам синхронизации.

Таблица 1 – Зависимость времени синхронизации от объема синхронизируемых данных

Размер данных, Кб	**Параметр t_c, мс**	
	I / III тип модулей ЦП	**II тип модулей ЦП**
5	5	5
10	10	10
15	10	15

Размер данных, Кб	Параметр t_c, мс	
	I / III тип модулей ЦП	II тип модулей ЦП
20	15	20
50	25	50
75	40	70
100	50	90
125	60	–
150	70	–
175	80	–
200	90	–
300	135	–
400	180	–
500	220	–

Время «тишины» – время бездействия задачи, когда весь пользовательский код уже выполнен (включая резервирование) и ожидается очередной вызов задачи от планировщика.

t_{0+1} – время начала нового цикла задачи резервирования.

Во время работы модули ЦП обмениваются сигналами (heartbeat) с периодичностью 5 мс. Отсутствие сигналов в течение двух периодов расценивается ЦП как потеря связи с партнером.

Параметры, настраиваемые пользователем для задачи синхронизации, приведены на вкладке ***Резервирование*** (см. в разделе «Обслуживание резервированного контроллера. Настройка резервирования»).

ИНФОРМАЦИЯ

Настроечные параметры необходимо задавать на вкладке **Резервирование.** Если добавлять в код программы экземпляр PsRedundancy.TConfigControl2 (определяющий параметры), то при компиляции будет появляться сообщение: «Обнаружена декларация структуры TConfigControl2, для настройки используйте редактор объекта резервирования вместо структуры»

Состояния модулей ЦП резервированного контроллера

Модуль ЦП в составе резервированного контроллера может находиться в одном из следующих состояний:

- **Инициализация**;
- **Синхронизация;**
- **Ведомый**;

- **Ведущий**;
- **Автономный**;
- **Ошибка соединения**.

Состояние **Инициализация** – это промежуточное состояние модуля ЦП, в котором он находится в момент запуска, перезагрузки или после восстановления связи с модулем ЦП-партнером. В состоянии **Инициализация** модуль ЦП производит сравнение своей версии программного обеспечения с версией, запущенной на модуле ЦП-партнере. В случае необходимости ЦП обновляет свое прикладное программное обеспечение до версии, полученной от ЦП-партнёра.

Если в состоянии **Инициализация** модуль ЦП не обнаруживает модуль ЦП-партнера, то он переходит в состояние **Автономный.**

Из состояния **Инициализация** модуль ЦП переходит в состояние **Синхронизация,** в котором пытается синхронизировать данные с модулем ЦП-партнера. При успешной синхронизации ЦП переходит в состояние **Ведомый**.

Если в состоянии **Синхронизация** модуль ЦП теряет связь с модулем ЦП-партнером, то он переходит в состояние **Автономный**.

Состояние **Ведомый** – это одно из двух штатных состояний модуля ЦП резервированного контроллера. В этом состоянии происходит постоянный обмен данными между модулями ЦП (как служебными, так и пользовательскими) и ведомый модуль ЦП в любой момент времени готов принять управление от ведущего модуля ЦП. В этом состоянии модуль ЦП находится до тех пор, пока штатно не перейдет в состояние **Ведущий**.

Если в состоянии **Ведомый** наступает ошибка синхронизации, то ЦП переходит в состояние **Синхронизация.** Ошибка синхронизации возникает при превышении допустимого времени рассинхронизации между задачами (t_p). В общем случае это происходит из-за превышения общего времени выполнения у задачи (время выполнения больше времени интервала задачи).

Если в состоянии **Ведомый** модуль ЦП теряет связь с модулем ЦП-партнером, то он переходит в состояние **Автономный**.

В состоянии **Ведущий** модуль ЦП осуществляет управление технологическим процессом до тех пор, пока не наступит одно из следующих событий:

- потеря связи с одним или несколькими модулями ввода/вывода (при условии, что у модуля ЦП-партнера не должно быть подобной неисправности);
- останов приложения (с помощью переключателя RUN/STOP; произошло событие EXCEPTION; обновление приложения в модуле ЦП (см. раздел «Обновление приложений резервируемого контроллера»));
- выполнена пользовательская команда по переключению в состояние **Ведомый** (из прикладной программы или из среды разработки Astra.IDE).

В этом случае произойдет штатная передача управления технологическим процессом модулю ЦП-партнеру, а ведущий модуль ЦП перейдет в состояние **Ведомый**.

Если в состоянии **Ведущий** модуль ЦП теряет связь с модулем ЦП - партнером, то он переходит в состояние **Автономный**.

В состоянии **Автономный** модуль ЦП находится в том случае, если он потерял связь с модулем ЦП-партнером. При восстановлении связи с модулем ЦП-партнером модуль ЦП может уйти в состояние **Ведущий** или в состояние **Инициализация**.

Если резервированный контроллер собран по схеме частичного или комбинированного резервирования, то модуль ЦП из состояния **Автономный** может перейти в состояние **Ошибка соединения**.

Состояние **Ошибка соединения** – это особое состояние ЦП в резервированном контроллере, при котором ЦП потеряли связь по линии синхронизации, но есть связь хотя бы с одним модулем ввода/вывода, общим для обоих модулей ЦП. При этом модуль ЦП, находящийся в состоянии **Ошибка соединения**, получает информацию от общих модулей ввода/вывода о нахождение модуля ЦП-партнера в состоянии **Автономный** (т.е. получает информацию об управлении технологическим процессом ЦП-партнером).

При восстановлении связи по линии синхронизации модуль ЦП из состояния **Ошибка соединения** переходит в состояние **Инициализация**.

Если в состоянии **Ошибка соединения** модуль ЦП получает информацию от модулей ввода/вывода о прекращении управления технологическим процессом ЦП-партнером (в том числе и по причине потери связи с общими модулями ввода/вывода), то он переходит в состояние **Автономный**.

Таблица 2 – Состояние индикаторов при нахождении модуля ЦП в определенном состоянии

Состояние модуля ЦП	**Выполнение прикладной программы**	**Управление технологическим процессом**	**Готовность взять на себя управление при отказе ведущего модуля ЦП**	**Состояние индикаторов**
Инициализация	нет	нет	нет	RUN – не горит; RDD – не горит
Синхронизация	да	нет	да*	RUN – горит; RDD – часто мигает; Lk** - мигает
Ведомый	да	нет	да	RUN – горит; RDD – редко мигает

Состояние модуля ЦП	Выполнение прикладной программы	Управление технологическим процессом	Готовность взять на себя управление при отказе ведущего модуля ЦП	Состояние индикаторов
Ведущий	да	да	-	RUN – горит; RDD – горит
Автономный	да	да	-	RUN – горит; RDD – не горит
Ошибка соединения	да	нет	да*	RUN – горит; RDD – не горит; Lk** – не горит

Примечание:

***** - безударная передача управления невозможна, так как отсутствует синхронизация данных прикладных задач между ЦП;

****** - индикатор мигает при наличии обмена через порты подключения линии синхронизации.

Особенности резервирования ПЛК с модулями ЦП II-го типа

В резервированной системе с применением модулей ЦП II-го типа накладываются следующие функциональные ограничения:

- минимальный цикл задачи резервирования равен 20 мс;
- объем резервируемых данных не должен превышать 100 Кб;
- приоритет задачи резервирования устанавливать только – 4.
- запрещены к использованию приоритеты в диапазоне 1-3 (для любых задач).

Отсутствует возможность резервирования линии синхронизации, т.к. на борту модуля ЦП ограниченное количество портов Ethernet.

В модулях ЦП II-го типа обмен по линии синхронизации чувствителен к наличию внешнего трафика по другому порту Ethernet. В связи с этим, при наличии интенсивного обмена по этому порту, возможно появление сообщений об ошибках синхронизации.

Пример реализации схемы резервирования с использованием модулей ЦП II-го типа модели R200 (Рисунок 13).

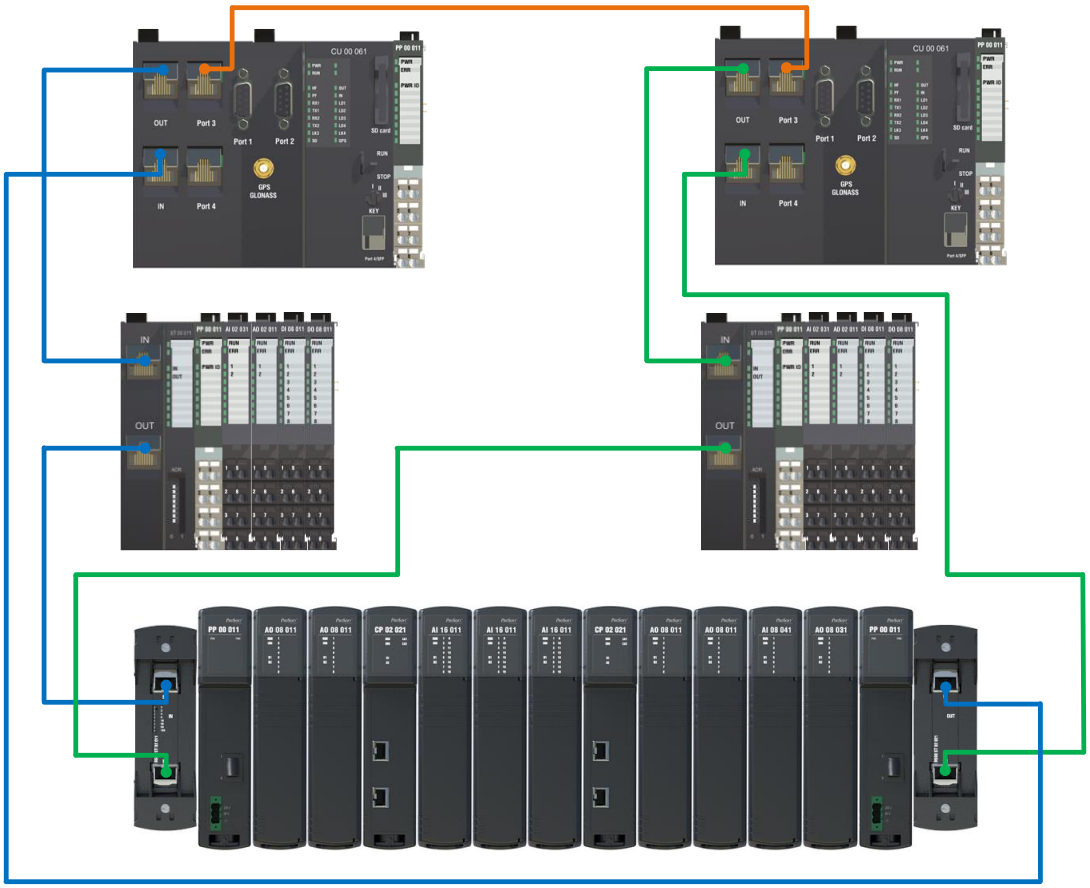

Рисунок 13 – Схема с резервированием, использующая модули ввода/вывода из линейки R200, R500 (общий) и базовый крейт R200

СКОРОСТЬ РЕАКЦИИ НА АВАРИЙНЫЕ СОБЫТИЯ

Скорость реакции на аварийные события регулируется следующими временными интервалами:

- **Таймаут модуля** (TO_mo) – временной интервал, по истечении которого модуль ЦП (мастер) бракует модуль ввода / вывода;
- **Таймаут мастера** (TO_ms) – временной интервал, по истечении которого модуль ввода / вывода бракует модуль ЦП (мастер).

Значение таймаута мастера определяет максимальный временной интервал, в течение которого модули вывода будут выдавать сигнал от ЦП, с которым уже потеряна связь (не актуальный сигнал). Этот период будет зависеть от того, в какой момент времени выполнения цикла прикладной задачи возникает неисправность. Если неисправность происходит в начале цикла, то период задержки управления будет минимальным (или его вообще не будет). Если же такая неисправность возникнет в самом конце цикла прикладной задачи, то задержка управления будет максимальная, т.е. равная таймауту мастера (пример поведения сигнала при разных таймаутах мастера, в случае аппаратной неисправности модуля CPU_A(мастера) в комбинированной схеме резервирования – Рисунок 14).

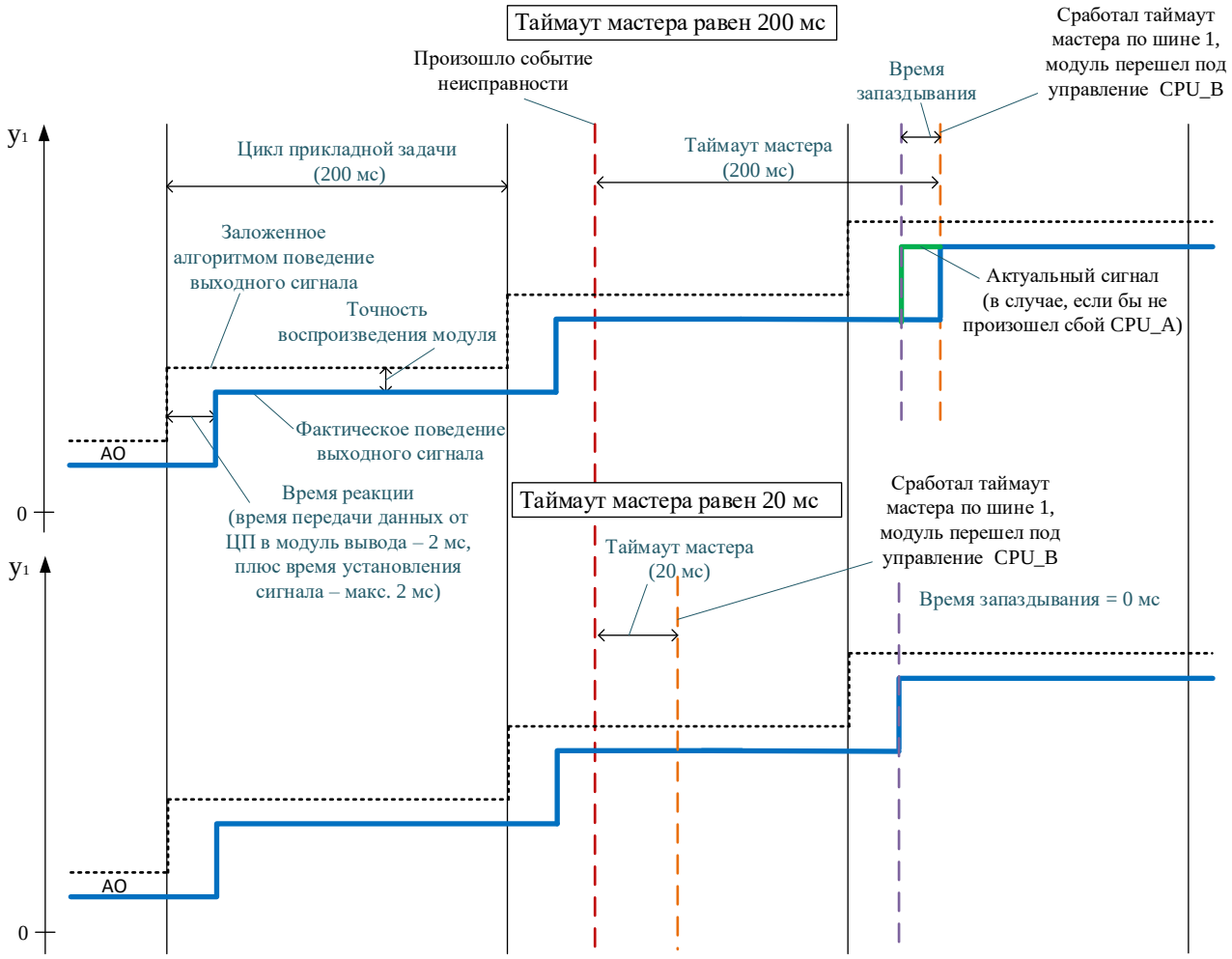

Рисунок 14 – Поведение сигнала при разных таймаутах мастера

Таймаут модуля, по аналогии с таймаутом мастера, определяет, какой максимальный интервал времени модуль ЦП будет использовать в прикладной программе неактуальное значение сигнала от модуля ввода/вывода, с которым уже потеряна связь. И так же, как и в случае с таймаутом модуля, реальная задержка будет зависеть от момента возникновения неисправности относительно цикла прикладной задачи.

Настройка таймаута модуля и таймаута мастера на модулях ЦП I/II -го типа

Таймаут модуля и таймаут мастера настраивается в **редакторе параметров шины Regul** (Рисунки 15, 16).

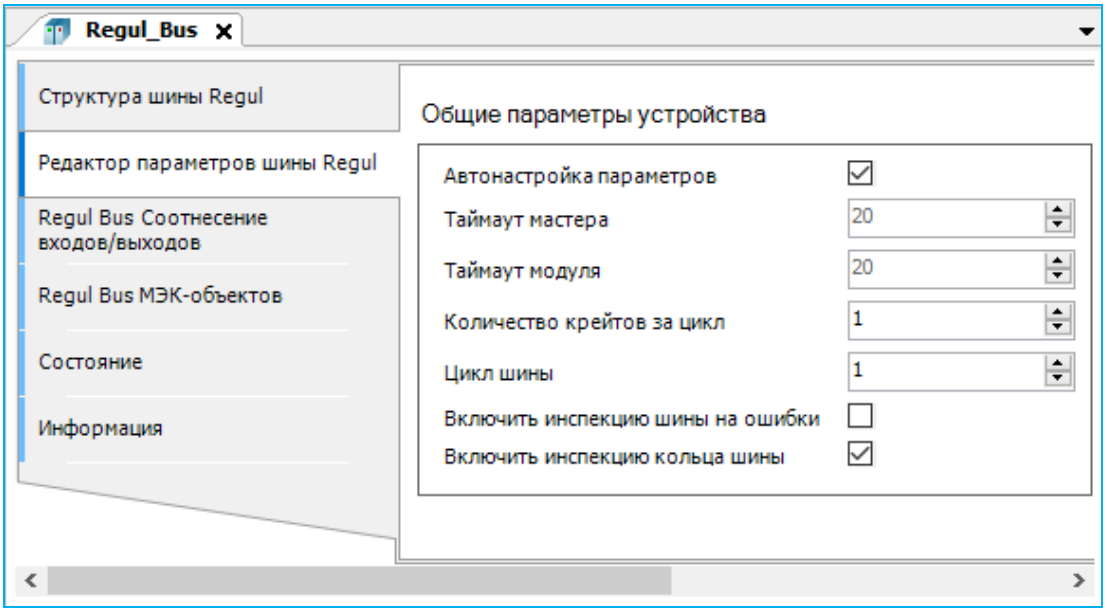

Рисунок 15 - Редактор параметров шины RegulBus

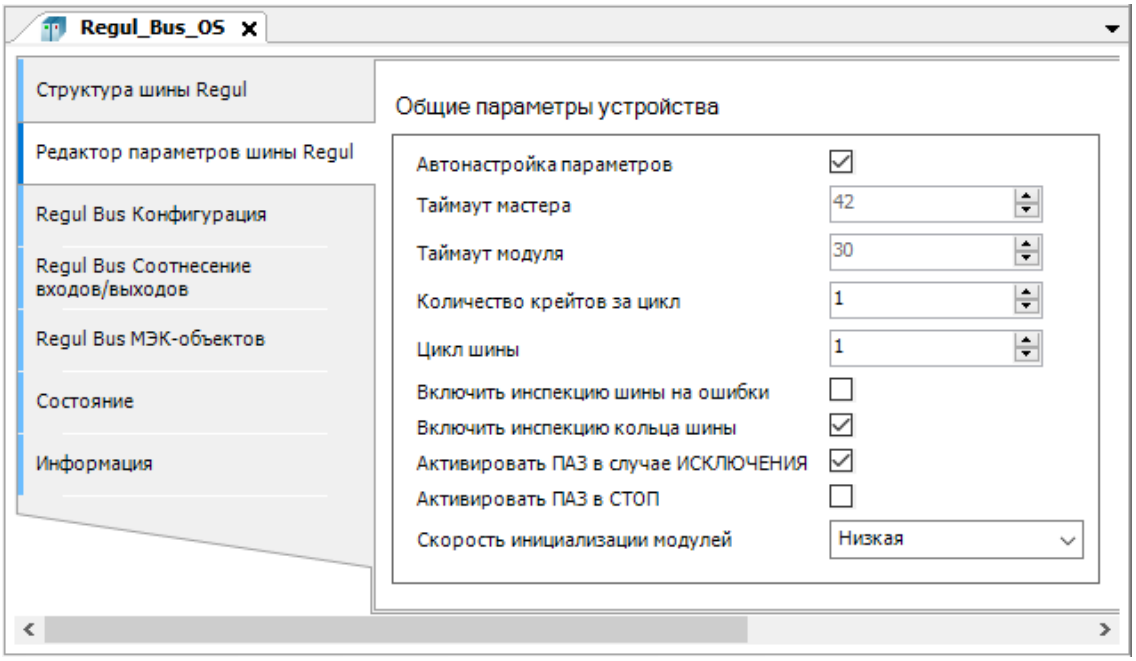

Рисунок 16 - Редактор параметров шины RegulBus OS

При активированном параметре «Автонастройка параметров» таймаут модуля и таймаут мастера рассчитываются по следующим формулам:

$$TO\_mo = Trd, \text{ но не менее 20 мс}, \quad (1)$$

где **Trd** – время цикла задачи резервирования. При этом должно выполняться следующее условие:

$$TO\_mo \geq 3*CI, \quad (2)$$

где **CI** – интервал опроса модулей;

$$TO\_ms = TO\_mo+2BI+ HwErrorSwitchDelayMs+10, \quad (3)$$

где BI (**Цикл шины**) – интервал работы шины, в мс (по умолчанию 1 мс);

HwErrorSwitchDelayMs – задержка передачи управления по аппаратной ошибке (смотри более подробное описание параметра в разделе «Настройка резервирования»).

При этом должно выполняться следующее условие:

$$TO\_ms \geq CI, \quad (4)$$

В свою очередь, вычисление интервала опроса модулей производится по следующей формуле:

$$CI = BI*CN/CPC, \quad (5)$$

где CPC (**Количество крейтов за цикл**) – количество крейтов, опрошенных за один цикл (по умолчанию 1);

CN (**Количество крейтов**) – количество крейтов в контроллере.

Полученное значение CI, если оно является дробным, округляется до целого числа в большую сторону.

Пользователь может изменить параметр **Цикл шины** (только в сторону увеличения) в случае, если не требуется частого опроса модулей (прикладная задача имеет существенно больший цикл).

Также пользователь может изменить параметр **Количество крейтов за цикл**. При увеличении этого параметра опрос всех модулей контроллера произойдёт за меньшее время. Так как увеличение параметра отразится на пиковой загрузке ЦП, то необходимо проконтролировать, чтобы задача RegulBusTask выполнялась за отведенное ей время.

При установке значения параметра **Количество крейтов за цикл** равному «0» все крейты контроллера будут опрашиваться в одном цикле, т.е. CI будет равняться BI.

Пользователь может самостоятельно установить требуемые значения таймаута мастера и таймаута модуля. Если значения таймаутов будут противоречить выше описанным условиям, то, после компиляции проекта, в окне сообщений появится информация об ошибке или предупреждение.

При установке TO_mo менее 20 мс или менее трех интервалов опроса модуля будет предупреждение вида: «Не выполняется условие TO_mo (15) ≥ 20ms и TO_mo (15) >= 3*CI (150). Система неустойчива к коммутационным и ЭМС воздействиям».

При установке TO_ms менее одного интервала опроса модуля будет ошибка вида: «Ошибка в структуре шины ПЛК: «Не выполняется условие TO_ms (20) ≥ CI (50). Необходимо откорректировать соответствующие параметры».

При установке TO_ms менее одного интервала опроса модуля будет ошибка вида: «Ошибка в структуре шины ПЛК: «Не выполняется условие TO_ms (20) ≥ CI (50). Необходимо откорректировать соответствующие параметры».

Следует иметь в виду, что увеличение значения таймаутов уменьшает чувствительность контроллера к внешним воздействиям, приводящим к потерям связи по шине RegulBus. Это приводит к уменьшению количества случаев передачи управления между ведущим и ведомым ЦП.

С другой стороны, увеличение таймаутов увеличивает время реакции резервированного контроллера на аварийное событие (например, отказ одного из модулей ввода/вывода, отказ модуля ЦП).

Напротив, уменьшение таймаутов ускоряет реакцию контролера на возникновение аппаратной ошибки, что бывает критически важно в ответственных системах с быстрым циклом управления. Но, в свою очередь, это приводит к увеличению количества случаев передачи управления между ведущим и ведомым ЦП.

	**ВНИМАНИЕ!** Изменять параметры, установленные по умолчанию, рекомендуется только в обоснованных случаях и при полной уверенности в работоспособности контроллера с измененными параметрами, так как неверно установленные параметры могут привести к некорректной работе контроллера. При наличии сбоев в работе контроллера необходимо вернуться к параметрам по умолчанию!

Настройка таймаута модуля и таймаута мастера на модулях ЦП III-го типа

На модулях III-го типа доступна только версия внутренней шины данных RegulBus OS. Для модулей этого типа *интервал опроса модулей* (*CI*) в большинстве случаев не превышает 1 мс. Настройка параметров **Таймаут мастера**, **Таймаут модуля** отличается от настройки, описанной выше.

Редактор параметров шин представлен на рисунке 17 (внутренняя вкладка **Редактор параметров шины Regul**).

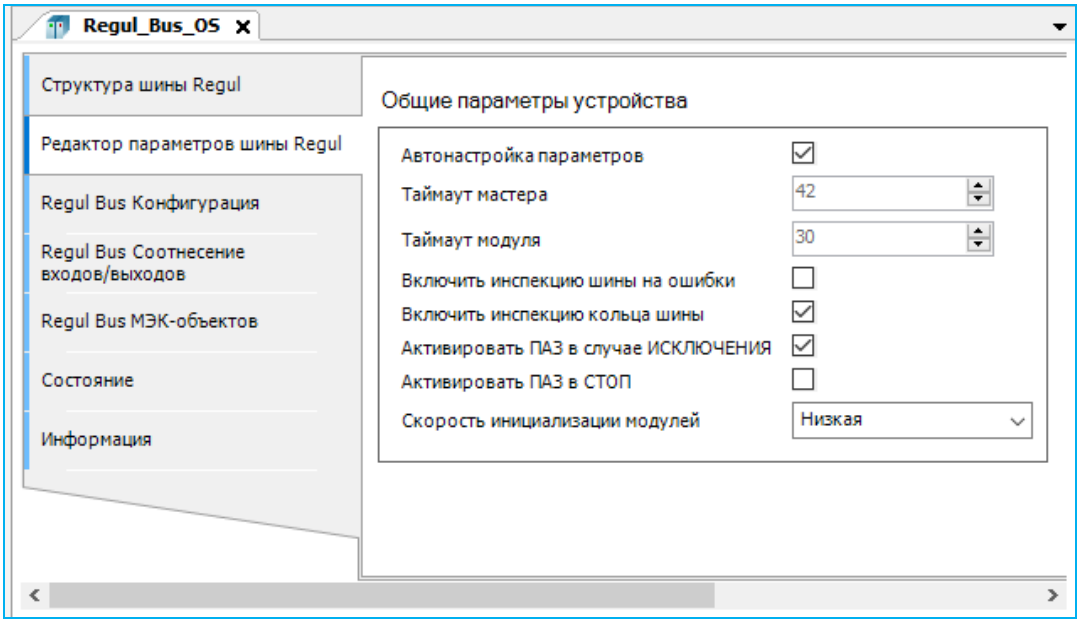

Рисунок 17 – Редактор параметров шины **RegulBus OS** (для III-го типа модулей)

Установка флажка в поле **Автонастройка параметров** блокирует возможность изменять значения параметров **Таймаут мастера**, **Таймаут модуля**. При активированном параметре **Автонастройка параметров** происходит автоматическая подстановка следующих значений:

$$TO\_mo = Trd, \text{ но не менее } 20 \text{ мс}, \quad (6)$$

где **Trd** – время цикла задачи резервирования.

$$TO\_ms = TO\_mo + HwErrorSwitchDelayMs + 10, \quad (7)$$

где **HwErrorSwitchDelayMs** – задержка передачи управления по аппаратной ошибке (смотри более подробное описание параметра в разделе «Настройка резервирования»).

Пользователь может самостоятельно установить требуемые значения таймаута мастера и таймаута модуля. Если значения таймаутов будут противоречить выше описанным условиям, то, после компиляции проекта, в окне сообщений появится информация об ошибке или предупреждение.

При установке TO_mo менее 20 мс или менее трех интервалов опроса модуля будет предупреждение вида: «Не выполняется условие TO_mo (15) ≥ 20ms и TO_mo (15) >= 3*CI (150). Система неустойчива к коммутационным и ЭМС воздействиям».

Поведение резервированного контроллера при наступлении аварийных событий

Поведение резервированного контроллера при различных вариантах отказов будет представлено на контроллере серии R500 с комбинированной схемой резервирования (Рисунок 18).

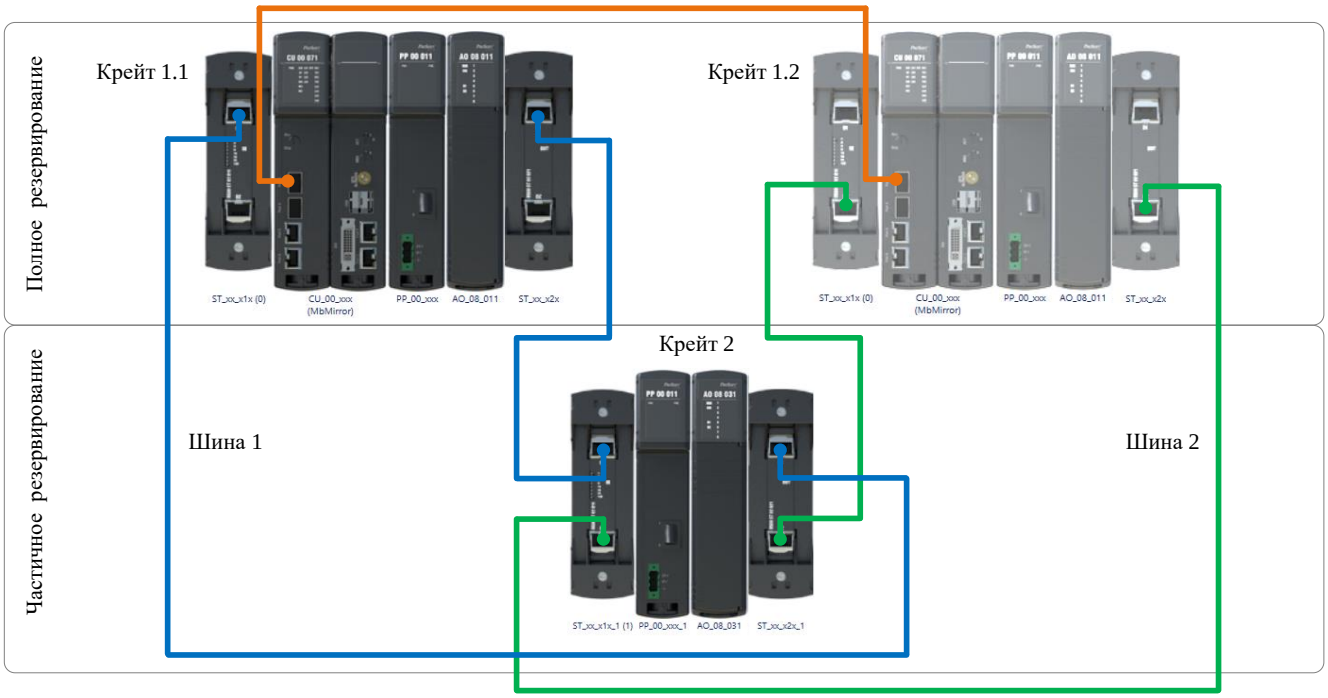

Рисунок 18 – Комбинированная схема резервирования

Состав резервированного контроллера:

- крейт 1.1 – базовый крейт, в составе которого: модуль источника питания, модуль ЦП (CPU_A) и модуль аналогового вывода (AO1);
- крейт 1.2 – базовый крейт, идентичный крейту 1.1(модуль источника питания, модуль ЦП (CPU_B) и модуль аналогового вывода (AO2));
- крейт 2 – крейт расширения, общий крейт для обоих ЦП, в составе которого: модуль источника питания и модуль аналогового вывода (AO).

Начальные условия: модуль CPU_A (крейт 1.1) – **ведущий**, модуль CPU_B (крейт 1.2) – **ведомый**.

Аппаратная неисправность ведущего модуля ЦП

Произошло аварийное событие: вышел из строя модуль CPU_A.

Индикация:

- крейт 1.1 – на модуле аналогового вывода AO1 горит красным индикатор B1;
- крейт 1.2 – на модуле CPU_B горит индикатор RUN. На модуле аналогового вывода AO2 горит зеленым индикатор B2, означающий, что модуль воспроизводит сигналы управления, переданные по второй шине RegulBus;
- крейт 2 – на модуле аналогового вывода AO горит красным индикатор B1 и зеленым индикатор B2.

Состояние модулей ЦП: модуль CPU_B перешел из состояния *Ведомый* (*StandBy*) в состояние *Автономный* (*Active_Standalone*).

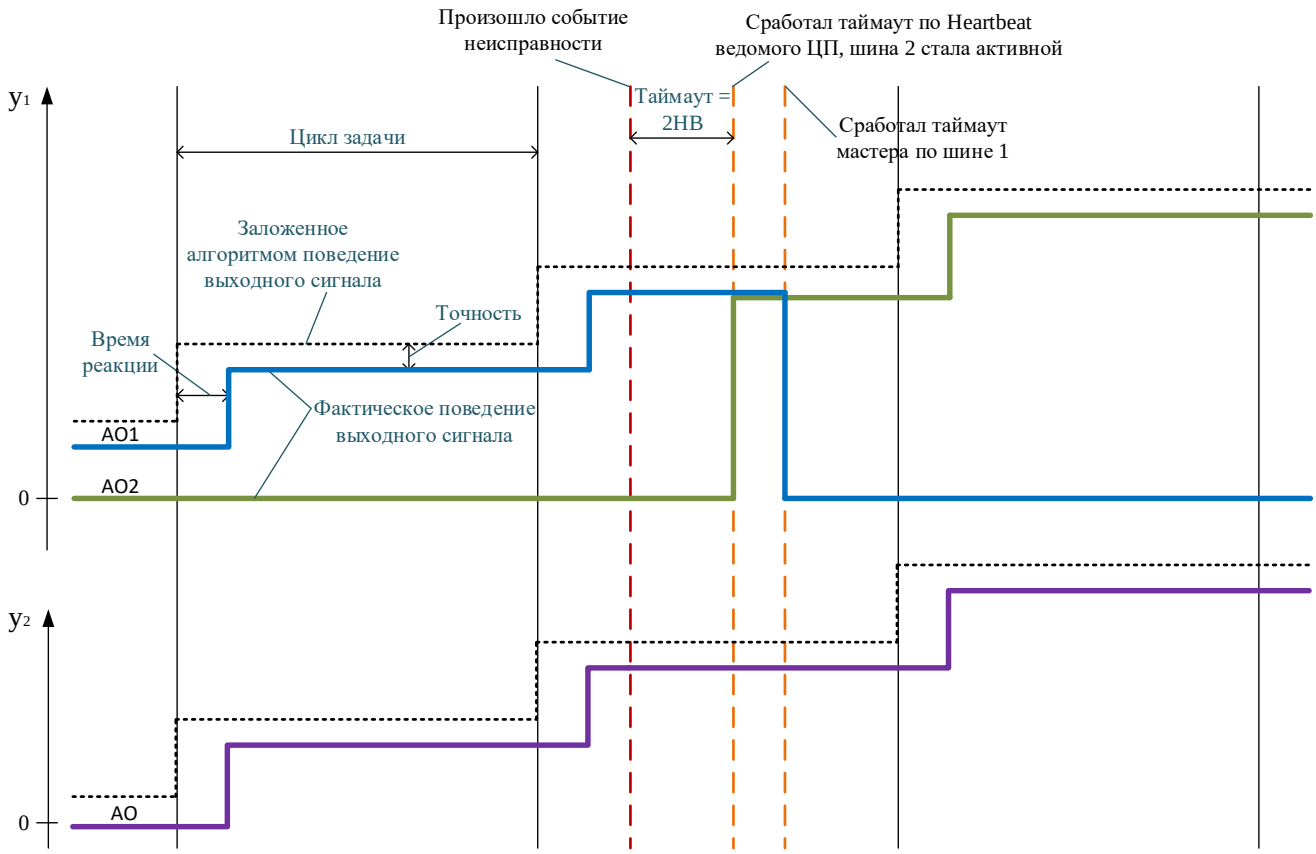

Рисунок 19 – Поведение сигнала при выходе из строя модуля ЦП1

При восстановлении модуля CPU_A его состояние изменится следующим образом: *Инициализация (Initialization)* → *Синхронизация* (*Sync*) → *Ведомый* (*StandBy*). При этом модуль CPU_B сменит состояние с *Автономный* (*Active_Standalone*) на *Ведущий* (*Active*).

Потеря входного питания модуля РР в крейте ведущего модуля ЦП

Произошло аварийное событие: нет питания в крейте 1.1.

Индикация:

– крейт 1.2 – на модуле CPU_B горит индикатор RUN. На модуле аналогового вывода AO2 горит зеленым индикатор B2;

– крейт 2 – на модуле аналогового вывода AO горит красным индикатор B1 и зеленым индикатор B2.

Состояние модулей ЦП: Модуль CPU_B перешел из состояния *Ведомый* (*StandBy*) в состояние *Автономный* (*Active_Standalone*).

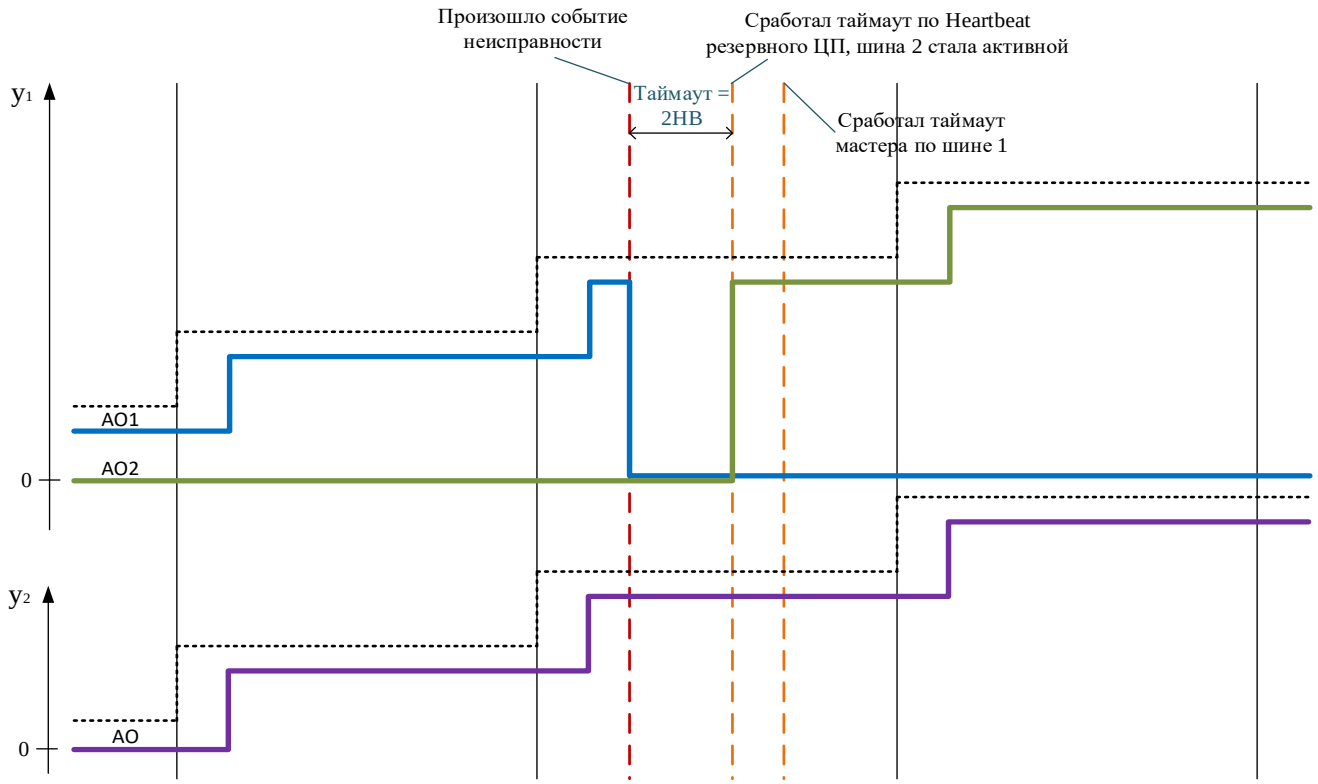

Рисунок 20 – Поведение сигнала при отсутствии питания в крейте 1.1

При восстановлении питания в крейте 1.1 состояние модуля CPU_A изменится следующим образом: *Инициализация (Initialization)* → *Синхронизация* (*Sync*) → *Ведомый* (*StandBy*). При этом модуль CPU_B сменит состояние с *Автономный* (*Active_Standalone*) на *Ведущий* (*Active*).

Аппаратная неисправность модуля AO в крейте ведущего модуля ЦП

Произошло аварийное событие: вышел из строя модуль аналогового вывода AO1 в крейте 1.1.

Индикация:

- крейт 1.1 – на модуле CPU_A редко мигает индикатор RDD, горит индикатор HF, показывающий, что отсутствует или неисправен один из модулей контроллера;
- крейт 1.2 – на модуле CPU_B горит индикатор RDD. На модуле аналогового вывода AO2 горит зеленым индикатор B2;
- крейт 2 – на модуле аналогового вывода AO горит красным индикатор B1 и зеленым индикатор B2.

Состояние модулей ЦП: модуль CPU_A перешел из состояния *Ведущий* (*Active*) в состояние *Ведомый* (*StandBy*). Модуль CPU_B перешел из состояния *Ведомый* (*StandBy*) в состояние *Ведущий* (*Active*).

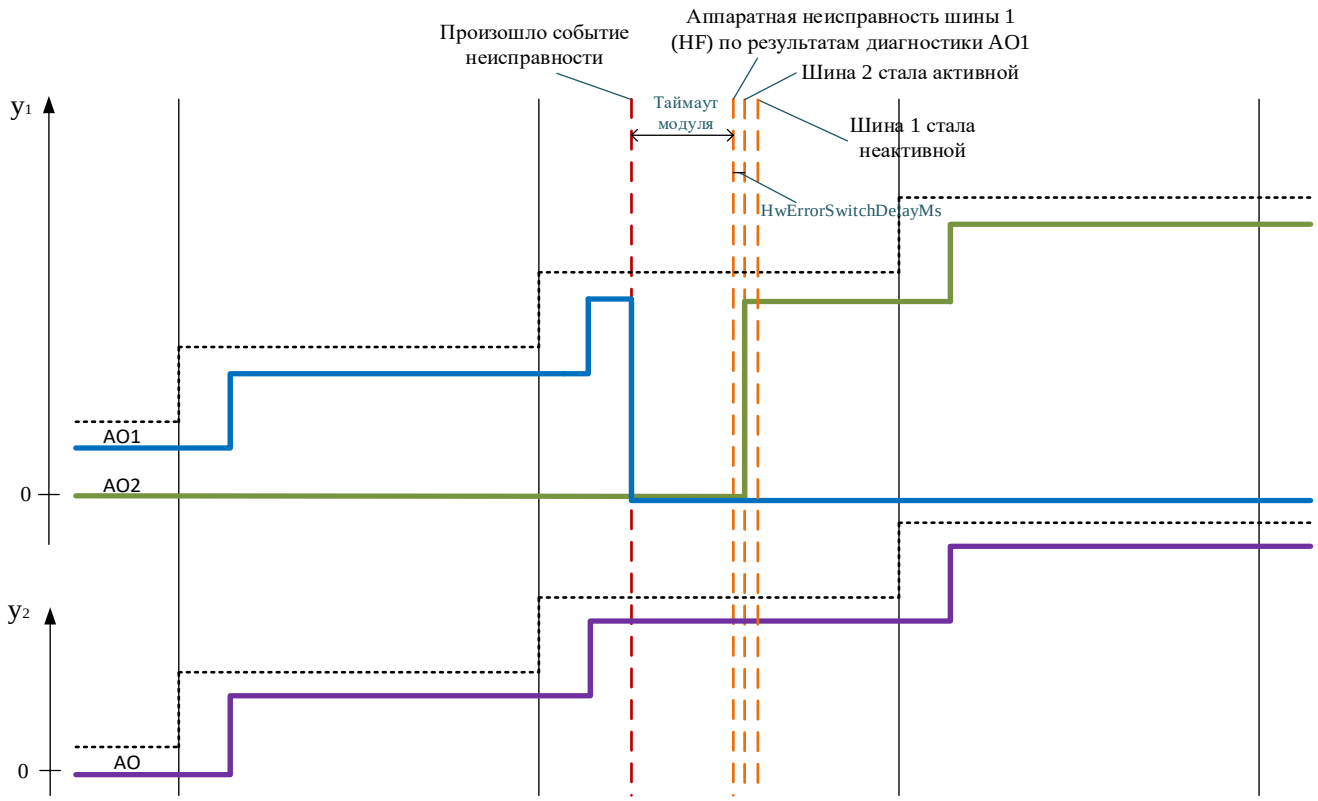

Рисунок 21 – Поведение сигнала при аппаратной неисправности модуля AO1 в крейте 1.1

По истечении **Таймаута модуля** ведущий модуль CPU_A бракует модуль аналогового вывода AO1 и в результате сформируется признак аппаратной ошибки.

При восстановлении модуля аналогового вывода AO1 в крейте 1.1 – изменений в состоянии модулей ЦП не произойдет.

Аппаратная неисправность модуля AO в общем крейте резервирования

Произошло аварийное событие: вышел из строя модуль аналогового вывода AO в крейте 2.

Индикация:

- крейт 1.1 – на модуле CPU_A горит индикатор RDD, горит индикатор HF;
- крейт 1.2 – на модуле CPU_B редко мигает индикатор RDD, горит индикатор HF;
- крейт 2 – на модуле AO не горят индикаторы или горит индикатор ERR.

При выходе из строя общего модуля не происходит переключения управления между модулями ЦП, если заранее установлена задержка передачи управления по аппаратной ошибке (HwErrorSwitchDelayMs). Значение задержки должно быть равным или больше интервала опроса модулей (CI).

Аппаратная неисправность линии синхронизации между модулями ЦП

Произошло аварийное событие: обрыв связи между модулями ЦП.

Индикация:

- крейт 1.1 – на модуле CPU_A индикатор RUN горит, не горят индикатор RDD и Lk3. На модуле аналогового вывода AO1 индикатор B1 горит зеленым;
- крейт 1.2 – на модуле CPU_B индикатор RUN горит и индикатор PF (красный) часто мигает, не горят индикатор RDD и Lk3. На модуле аналогового вывода AO2 индикатор B2 горит желтым, означающим, что идет обмен по второй шине RegulBus, но модуль не воспроизводит сигналы управления;
- крейт 2 – на модуле аналогового вывода AO горит зеленым индикатор B1 и желтым индикатор B2.

Состояние модулей ЦП: модуль CPU_A перешел из состояния *Ведущий* (*Active*) в состояние *Автономный* (*Active_Standalone*). Модуль CPU_B перешел из состояния *Ведомый* (*StandBy*) в состояние *Автономный* (*Active_Standalone*), далее в состояние *Ошибка соединения* (*Cn_Error*). Управление будет осуществляться модулем CPU_A.

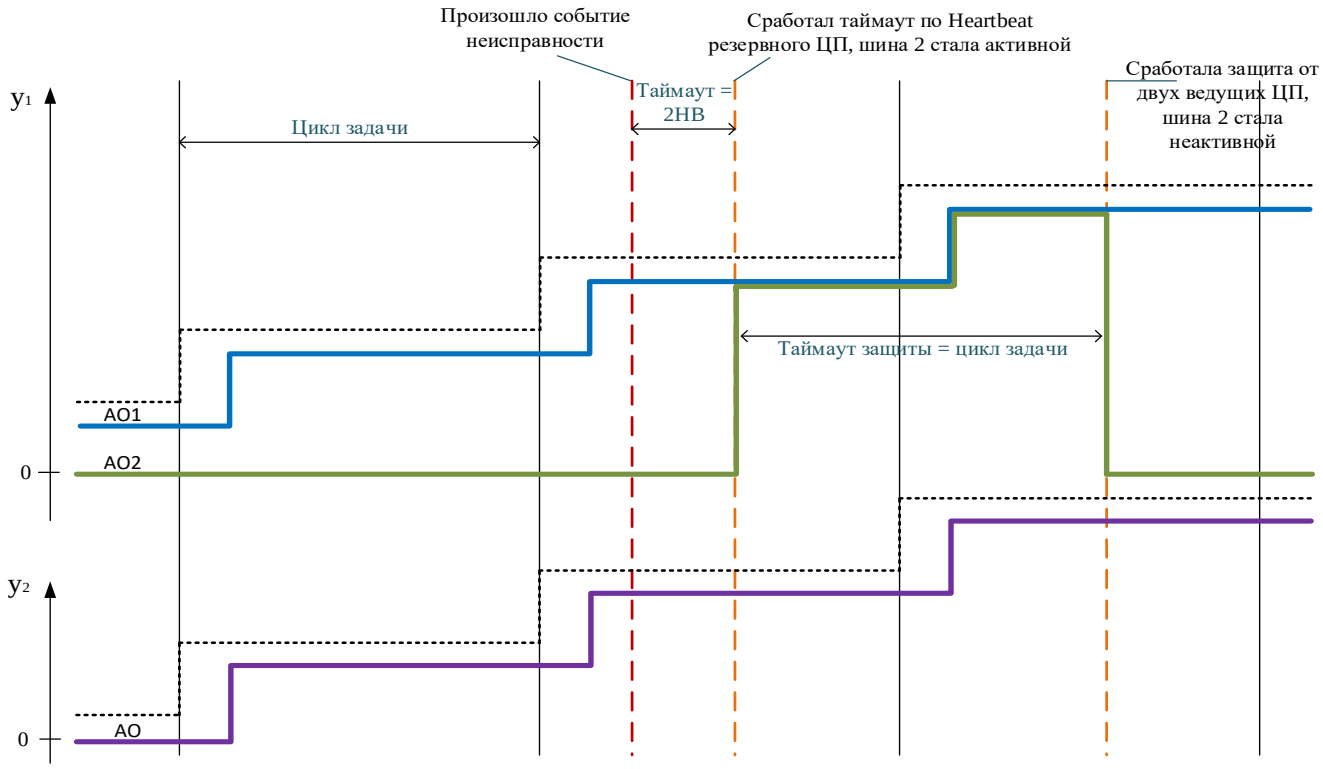

Рисунок 22 – Поведение сигнала при аппаратной неисправности линии связи между модулями ЦП

Для того, чтобы после обрыва связи и срабатывания защиты от двух ведущих ЦП вернуться к штатной работе системы резервирования, необходимо восстановить линию синхронизации.

Следует учесть, что в контроллерах с полным резервированием (где отсутствует крейт с общими модулями для обоих ЦП) при обрыве линии синхронизации оба модуля ЦП переходят в состояние *Автономный* (*Active_Standalone*) и осуществляют управление независимо друг от друга.

Для исключения данной ситуации рекомендуется передавать признак активности ЦП-партнера посредством физической линии связи. Для этого необходимо с помощью одного из

дискретных выходов передавать сигнал активности (сигнал всегда TRUE при запущенном прикладном проекте) на дискретных вход ЦП-партнера. При этом к данному дискретному входу должна быть привязана переменная xOtherBusInOperationalExtra, с помощью которой параметру Regul_Bus.OtherBusInOperationalExtra передается признак активности от физического входа.

Аналогичную связь требуется установить и со стороны ЦП-партнера.

НАСТРОЙКА РЕЗЕРВИРОВАНИЯ В СРЕДЕ

Настройка резервирования (Redundancy)

Начало работы

Запустите среду разработки. Создайте или откройте проект.

Настройка резервирования состоит из следующих шагов:

- добавление в проект и настройка компонента **Резервирование** (Redundancy);
- описание в проекте аппаратной конфигурации резервируемого контроллера;
- настройка IP-адресов для резервирования на каждом из модулей ЦП (также на коммуникационных модулях в составе контроллера);
- добавление в проект программного кода, указывающего экземпляры резервируемых данных и вызывающего синхронизацию в рабочем цикле одной из задач.

Добавление в проект компонента «Резервирование» (Redundancy)

При создании нового проекта с помощью Мастера конфигурации Regul на этапе **Дополнительные настройки** поставьте переключатель в поле **Резервирование**. Компонент **Резервирование (Redundancy)** будет автоматически добавлен в дерево устройств (Рисунок 23).

Также будет создана задача **TASK_PLC_PRG** с приоритетом 1 для I/III-го типа ПЛК, либо 4 для II-го типа, в программный код будет добавлен вызов функции синхронизации (Synchronize4) и один байт памяти (SharedMemory).

ВНИМАНИЕ!

Начиная с версии 1.7.1.0 на этапе создания нового проекта будет автоматически добавлен компонент **Резервирование** (**Redundancy OS)** в дерево устройств (Рисунок 24)

Подробное описание работы с Мастером конфигурации Regul приведено в документе «Программное обеспечение Astra.IDE. Руководство пользователя».

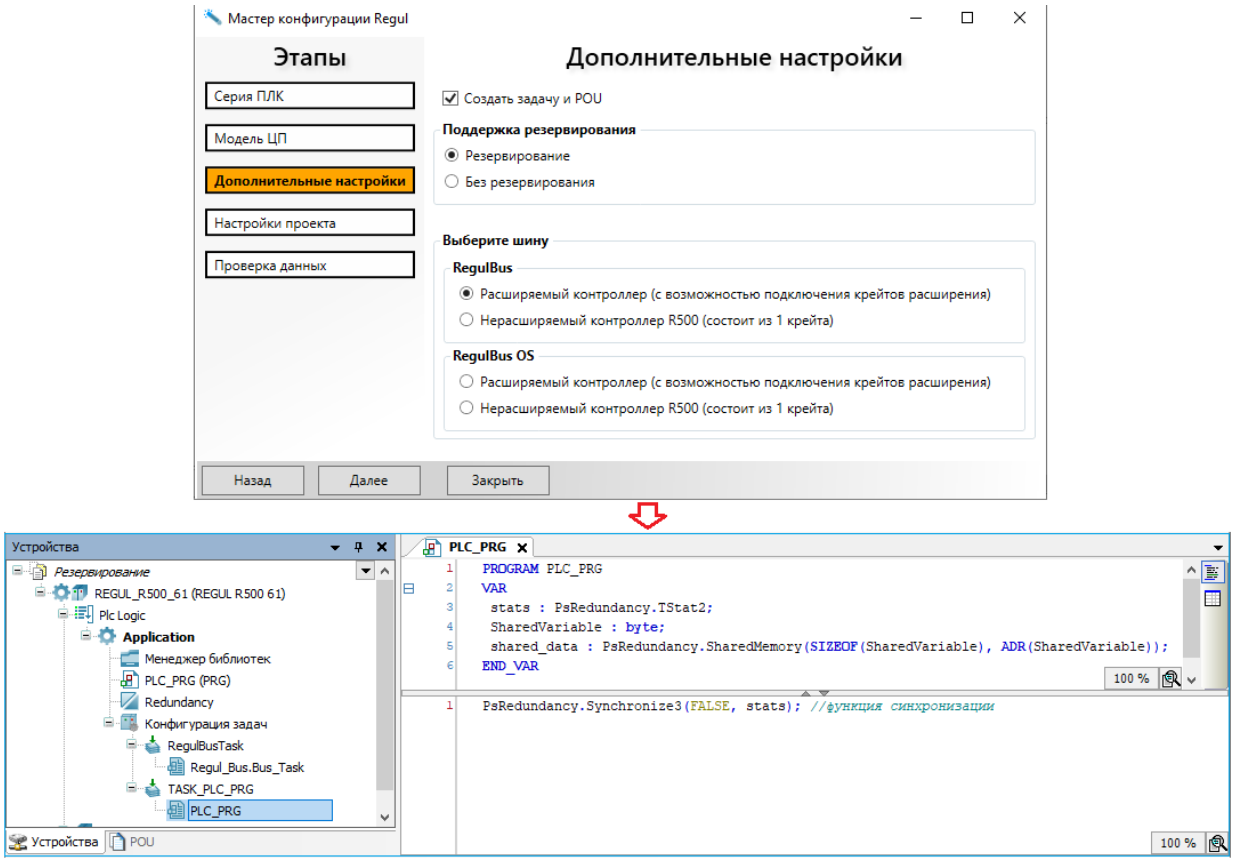

Рисунок 23 – Автоматическое добавление компонента «Резервирование» при создании нового проекта до версии 1.7.1.0

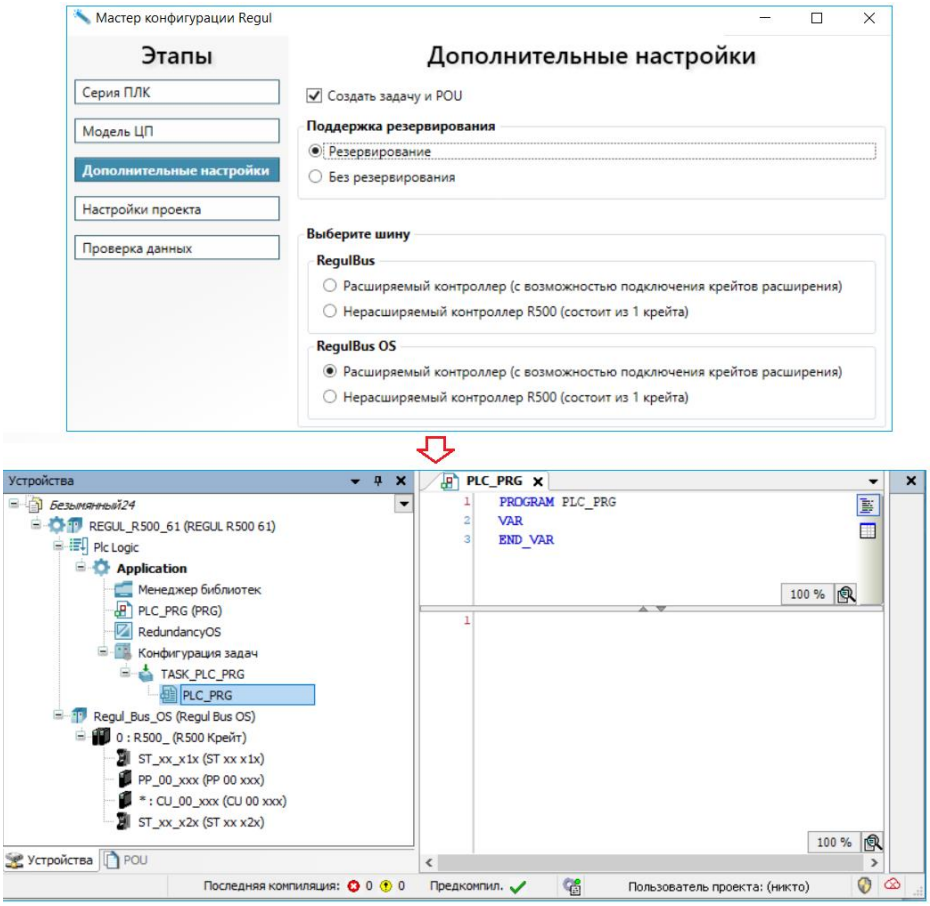

Рисунок 24 – Автоматическое добавление компонента «Резервирование» при создании нового проекта начиная с версии 1.7.1.0

Для добавления компонента **Резервирование** в существующий проект поставьте курсор на название приложения (Application), нажмите правую кнопку мыши. Появится контекстное меню, в котором выберите **Добавить объект ▸ ⇨ Резервирование…** Откроется окно **Добавить Резервирование**, где нажмите кнопку ***Добавить***. Компонент **Резервирование** будет помещен в дерево устройств.

При добавлении компонента **Резервирование** в проект автоматически добавляется библиотека PsRedundancy, которая осуществляет связь между модулями ЦП в схеме резервирования, предоставляет интерфейс для описания данных, требующих синхронизации, осуществляет синхронизацию рабочего цикла и предоставляет диагностическую информацию для компонента **Резервирование**.

Построение аппаратной конфигурации резервируемого контроллера

Построение аппаратной конфигурации стандартного контроллера, а также принципы добавления крейтов и модулей подробно описаны в документе «Программное обеспечение **Astra.IDE.** Руководство пользователя».

Построение аппаратной конфигурации для резервированного контроллера аналогично, за исключением наличия поля **Режим резервирования** в редакторе крейта (Рисунок 25). В поле **Режим резервирования** отображается, какой выбран режим резервирования (*Полное* или *Частичное*). Визуальная конструкция схемы резервирования отобразится в редакторе шины Regul_Bus во вкладке **Структура шины Regul**. Если выбрать в поле **Режим резервирования** значение *Полное,* то в редакторе шины создастся клон данного крейта, который определяется более тусклым цветом (Рисунки 26, 27). Клон крейта не подлежит редактированию, а все параметры и конфигурацию он принимает с исходного крейта.

Соответственно, при выборе значения *Частичное* резервирование в соответствующем поле, крейт будет определяться как общий для обоих модулей ЦП.

Для создания комбинированной схемы резервирования необходимо к базовому крейту (с модулем ЦП), с выбранным режимом резервирования *Полное,* добавить в проекте к шине Regul Bus крейт общий, с выбранным режимом резервирования *Частичное* (Рисунок 28).

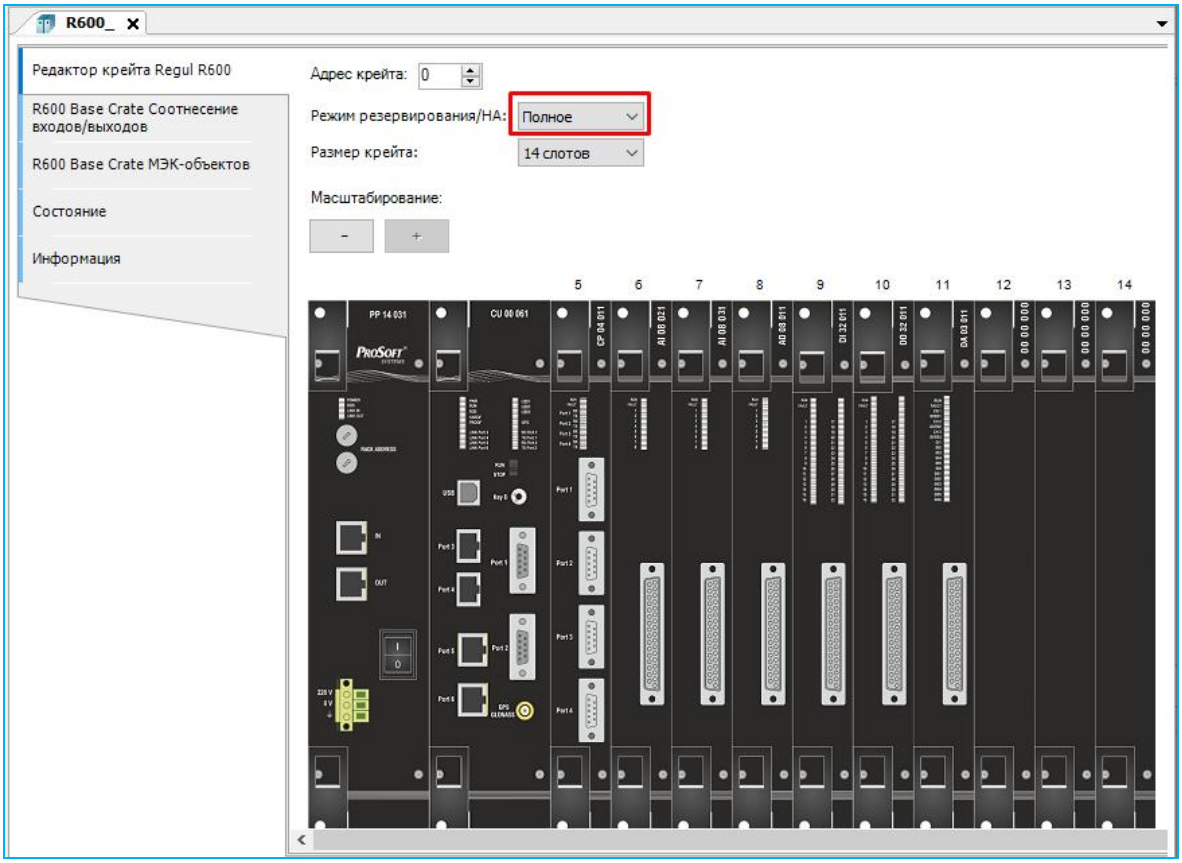

Рисунок 25 – Пример редактора крейта (контроллер Regul R600)

Рисунок 26 – Редактор шины контроллера R500 (схема с полным резервированием)

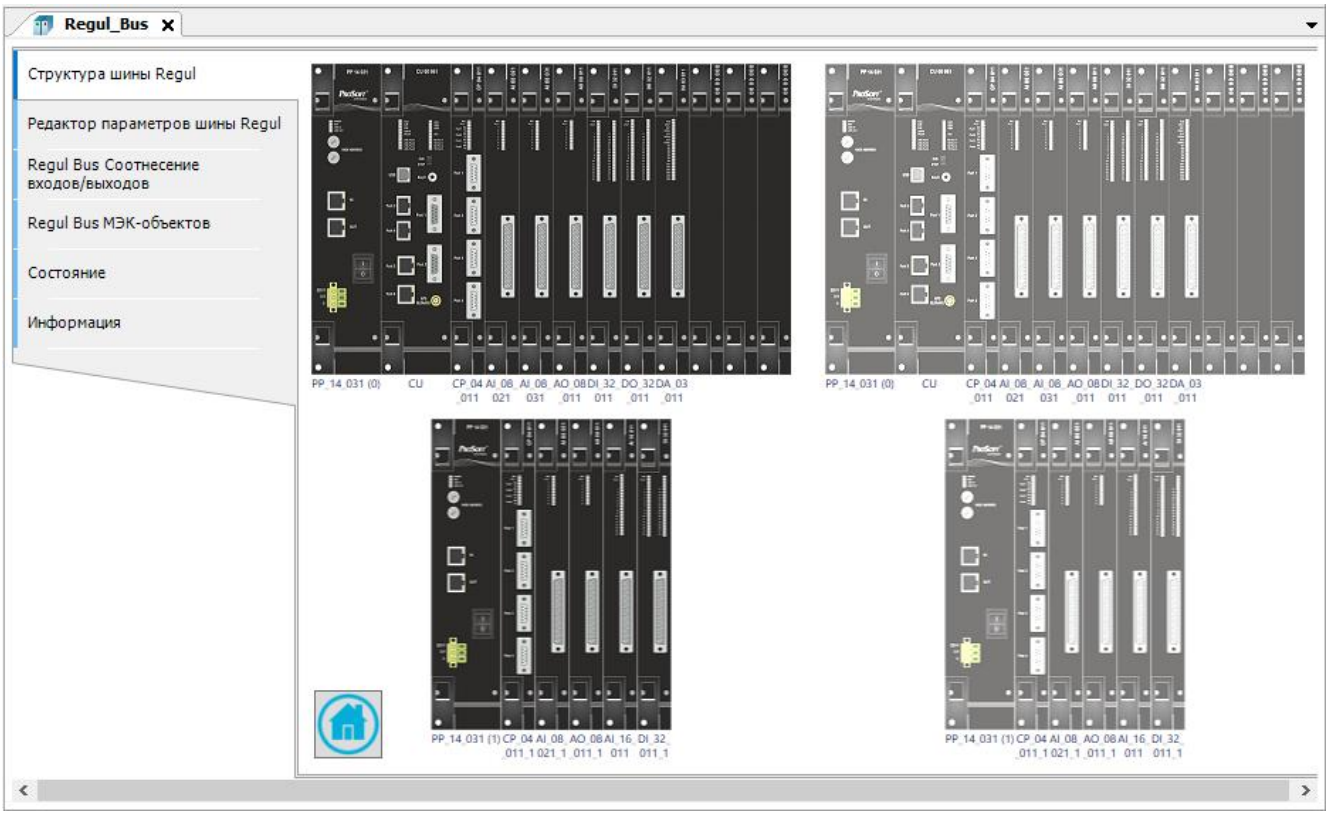

Рисунок 27 – Редактор шины контроллера R600 (2 крейта, с полным резервированием)

Рисунок 28– Редактор шины контроллера R500 (полное + частичное резервирование)

Настройка IP-адресов модулей ЦП в резервированном контроллере

При построении систем резервирования необходимо настроить порты для подключения линий синхронизации (канал 1 и/или 2) через Сканер сети (выберите в основном меню **Инструменты** ⇨ **Сканер сети**). Более подробное описание настройки основных параметров приведено в документе «Программное обеспечение **Astra.IDE.** Руководство пользователя», раздел «Подключение контроллера к сети».

Линии синхронизации могут быть настроены на любом порту Ethernet модуля ЦП.

ВНИМАНИЕ!

Не настраивайте линию синхронизации на порту, который отмечен зеленым цветом, иначе будет потеряна связь с контроллером из среды разработки

Для выбора режима работы порта, достаточно установить переключатель в соответствующее поле (другой режим на порту будет автоматически заблокирован):

- ***Статический адрес*** – IP-адрес порта, с которого контроллер будет отвечать на сетевые запросы. В полях **IP** и **Mask**, введите IP-адрес и маску, соответствующие подсети, в которой будет находиться контроллер (и ваш компьютер).
- ***Линия синхронизация*** – канал модуля ЦП для организации линии синхронизации в резервированном контроллере. Выберите из выпадающего списка необходимое значение (Рисунок 29) и нажмите кнопку ***Записать***.

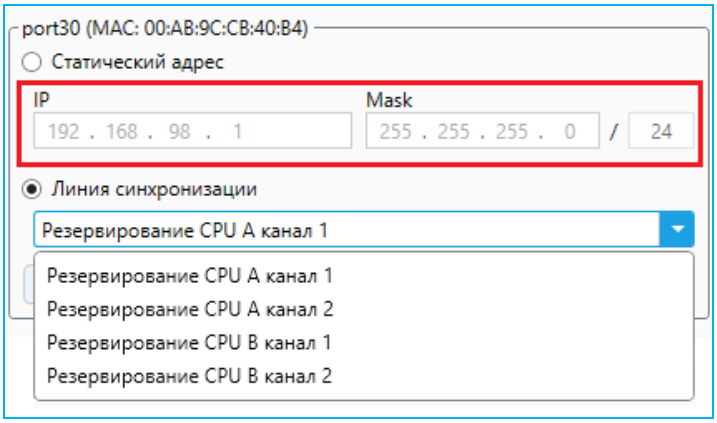

Рисунок 29 – Определение параметров линии синхронизации

Если используется одна линия синхронизации (канал 1), то достаточно выбрать роль модуля ЦП (CPU A или CPU B) в резервированном контроллере (смотри раздел «Роли модулей ЦП в составе резервированного контроллера») и будет автоматически проставлен зарезервированный IP-адрес (Таблица 3).

Таблица 3 - IP-адреса портов линии синхронизации

	CPU_A	**CPU_B**
Канал 1	192.168.98.1	192.168.98.2
Канал 2	192.168.99.1	192.168.99.2

При наличии второй линии синхронизации (канал 2), значение в выпадающем списке будет проставлено автоматически (Рисунок 30)

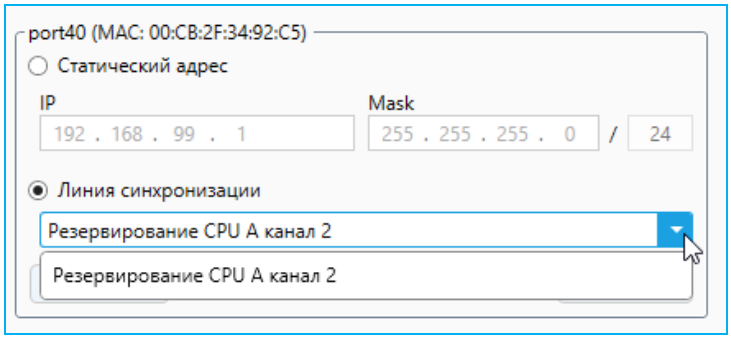

Рисунок 30 - Определение параметров второй линии синхронизации

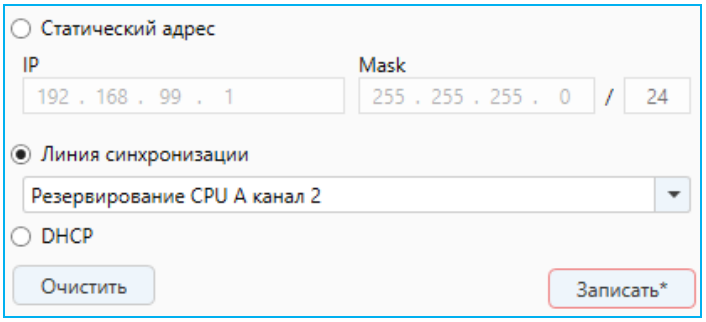

Рисунок 31 – Сохранение изменений

	**ИНФОРМАЦИЯ** Для сохранения внесенных изменений на ПЛК необходимо нажать кнопку ***Записать*** (при этом, будет появляться напоминание - «*» и контур кнопки будет окрашен в красный цвет). После назначения IP-адресов и нажатия кнопки ***Записать***, потребуется перезагрузить ЦП для вступления в силу сетевых настроек

Далее необходимо подключиться ко второму модулю ЦП в резервированном контроллере и произвести аналогичные настройки (Рисунок 32).

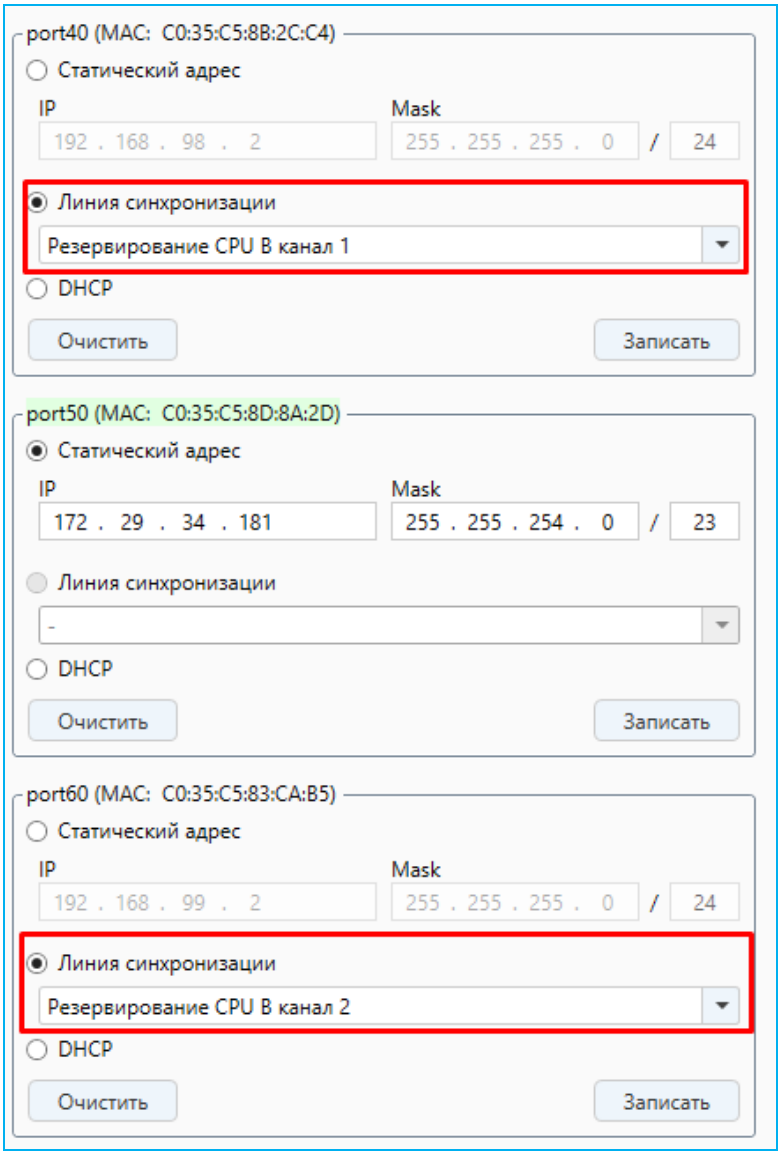

Рисунок 32 - Пример настройки второго модуля ЦП (CPU B) в резервированном контроллере с двумя линиями синхронизации

Конфигурация коммуникационных модулей Ethernet в резервированной системе

Коммуникационные модули могут работать в схеме полного, частичного и комбинированного резервирования.

Во всех схемах резервирования данные транслируются с обоих модулей ЦП-партнёров. Разница состоит лишь в том, что в схеме полного резервирования каждый ЦП транслирует данные через свой коммуникационный модуль, а в схемах частичного/комбинированного резервирования через общий коммуникационный модуль. Трансляция данных в различных схемах резервирования приведена на рисунке 33.

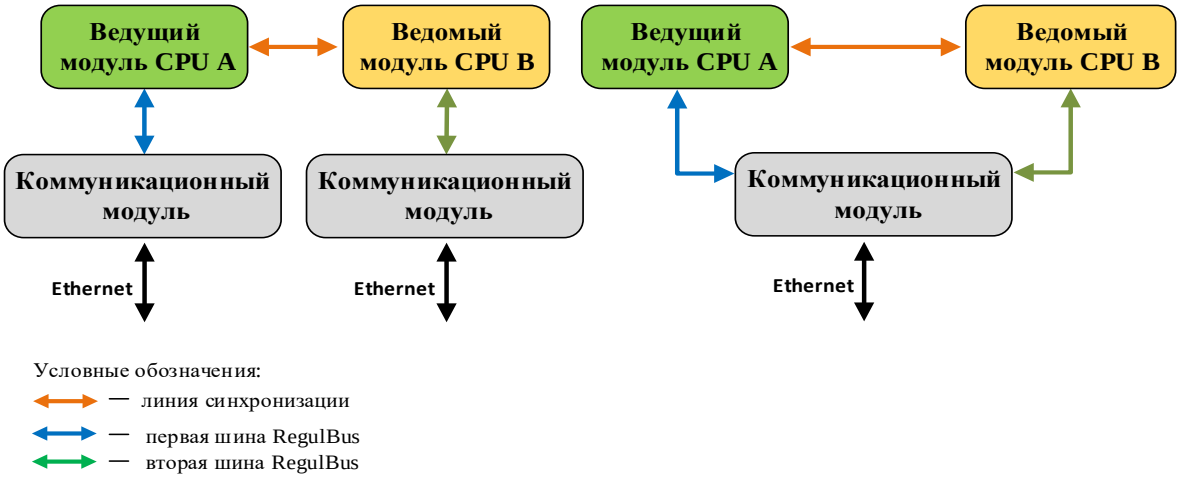

Рисунок 33 – Трансляция данных с модулей ЦП в различных схемах резервирования

Для настройки параметров, зайдите на вкладку **Редактор модуля CP 0x 021**, где будут представлены все параметры сетевой конфигурации. Общее описание параметров коммуникационных модулей приведено в документе «Программное обеспечение Astra.IDE. Руководство пользователя», раздел «Задание параметров модулей коммуникационного процессора».

Пример настройки IP и MAC адресов в коммуникационном модуле приведен на рисунке 34.

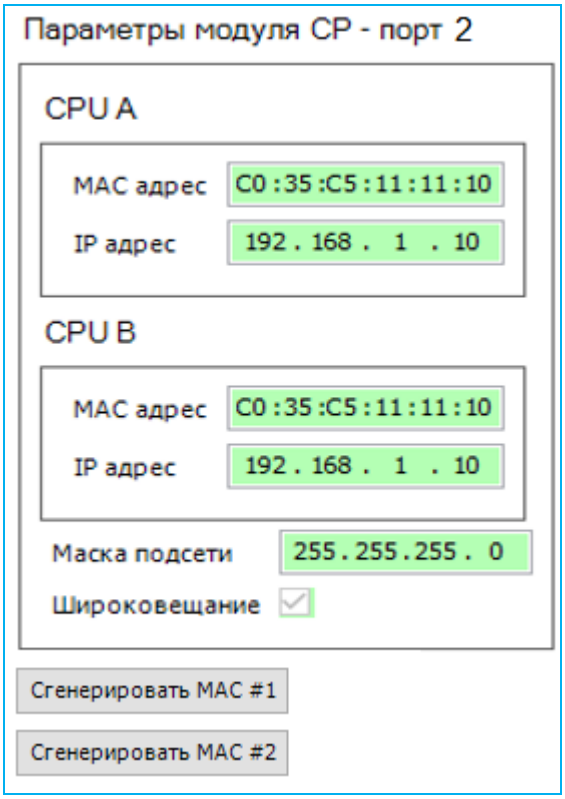

Рисунок 34 – Пример настройки порта 2 на модуле CP 0x 021 в схеме частичного резервирования

При нажатии на кнопку *Сгенерировать MAC #1 (#2) в* редакторе модуля генерируются MAC адреса для CPU A и CPU B в схемах:

- **полного резервирования** – разные MAC адреса;
- **частичного резервирования** – одинаковые MAC адреса.

	**ВНИМАНИЕ!** В схеме полного резервирования крейта, в котором расположен коммуникационный модуль, настройка одинаковых (или из одной подсети) IP и/или одинаковых MAC адресов на портах данных модулей и объединение их в единую сеть может вызвать конфликт сети, что приведет к потере связи с обоими коммуникационными модулями

	**ВНИМАНИЕ!** В схеме частичного резервирования крейта, в котором расположен коммуникационный модуль, при назначении одинаковых MAC-адресов на одном порту данного модуля, обмен данными будет проводиться только с ведущим модулем ЦП. Назначение одинаковых IP-адресов с различными MAC-адресами может вызвать конфликт сети, что приведет к ошибкам в ее работе

	**ИНФОРМАЦИЯ** Работа коммуникационного модуля с ранней версией СПО (до 1.0.25.0), по схеме частичного/комбинированного резервирования, возможна только при установке одинаковых MAC адресов для одного и того же порта в обоих модулях ЦП

Конфигурация коммуникационных модулей RS-485 в резервированной системе

Коммуникационные модули могут работать в схеме полного, частичного и комбинированного резервирования.

В зависимости от схемы резервирования данные будут транслироваться следующим образом:

– при полном резервировании – данные транслируются с обоих ЦП-партнеров через свой коммуникационный модуль;

– при частичном резервировании – данные транслируются с ведущего модуля ЦП через общий коммуникационный модуль.

Трансляция данных в различных схемах резервирования приведены на рисунках 35, 36.

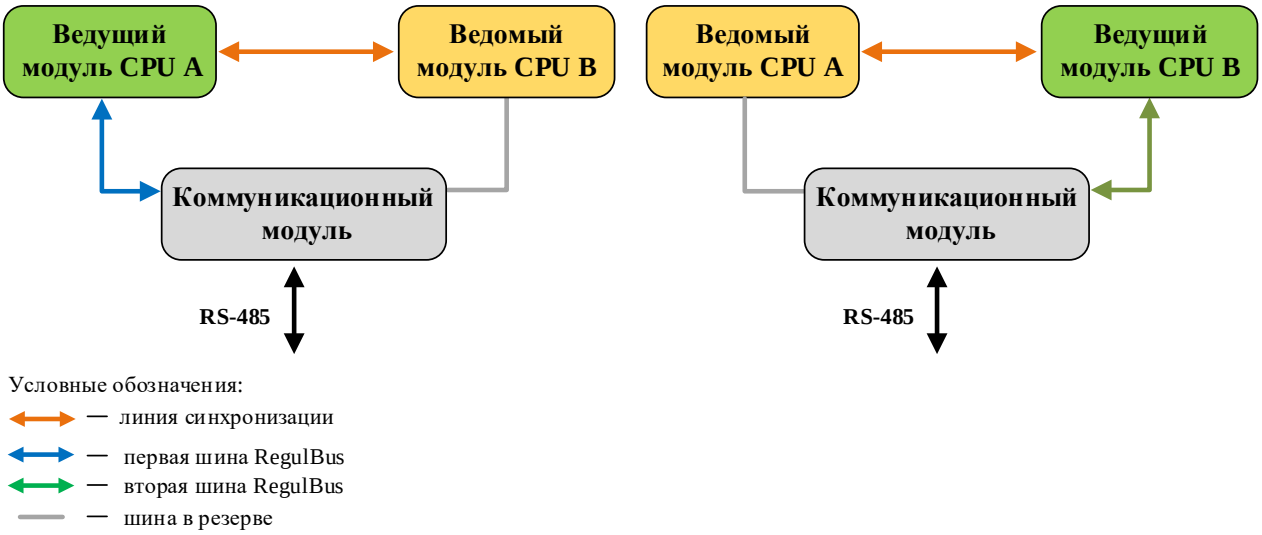

Рисунок 35 – Трансляция данных с модулей ЦП в схеме частичного резервировании через коммуникационные модули RS-485

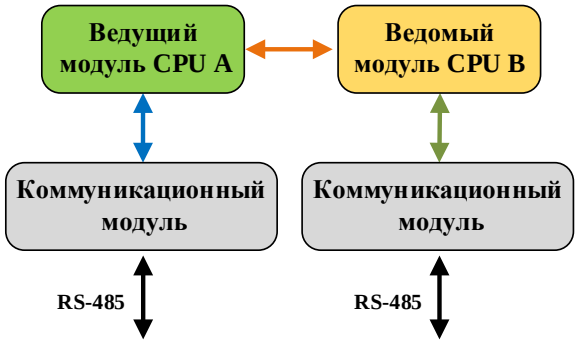

Рисунок 36 – Трансляция данных с модулей ЦП в схеме полного резервировании через коммуникационные модули RS-485

Резервируемые переменные и программирование контроллера

Список резервируемых переменных формируется пользователем на этапе разработки прикладной программы.

	**ВНИМАНИЕ!** **POINTER**-переменные нельзя объявлять в областях данных, которые контролируются резервированием. Так как значения, контролируемые резервированием, передаются на другой ПЛК во время синхронизации, то значения указателя будут недействительны на другом ПЛК, потому что память там может быть распределена иначе

При компиляции функция резервирования проверяет, расположены ли переменные-указатели в области, контролируемой резервированием. Для каждой переменной-указателя, обнаруженной в такой области, выдается предупреждение. В частности, в окне сообщений появляются строки с предупреждающей информацией следующего типа: «Указатели запрещены для резервирования…» (Рисунок 37).

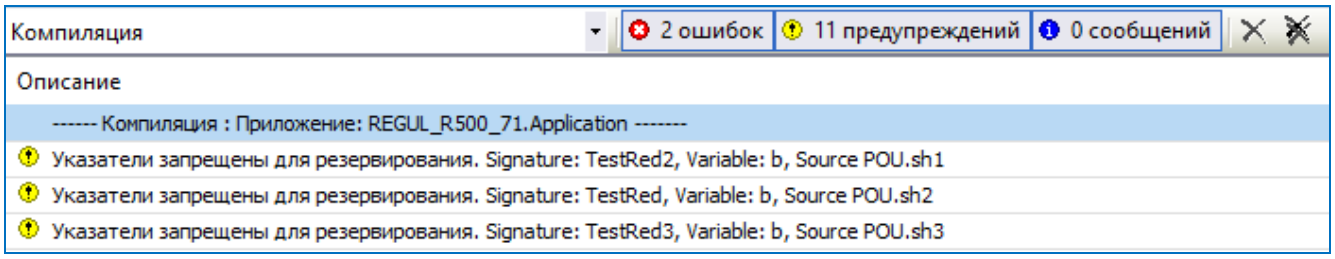

Рисунок 37 – Предупреждающие сообщения о попадании указателей в область резервируемых данных

	**ВНИМАНИЕ!** Будьте осторожны при использовании дескрипторов файлов и дескрипторов операционной системы, поскольку они могут привести к разным данным на каждом из ПЛК

Пользователю необходимо определиться с необходимым и достаточным набором резервируемых данных, учитывая, что необоснованное включение большого объема данных в задачу синхронизации может привести к увеличению цикла задачи резервирования. Рекомендуется предварительно объединить синхронизируемые переменные в структуры, чтобы уменьшить количество объектов синхронизации.

Необходимо помнить, что корректно синхронизуются исходные данные только для одной задачи – для той, в которой осуществляется вызов функции **Synchronize**. Исходные данные для остальных задач, выполняемых на резервированном контроллере, могут быть замещены данными, полученными в результате работы других задач.

Максимальный суммарный объем резервируемой памяти (блоков **SharedMemory** и **CrossMemory**) составляет:

– **500 Кб** (**200 Кб** - до версии СПО 1.6.5.4) - для контроллеров с модулями ЦП I/III-го типа**;**

– **100 Кб** - для контроллеров с модулями ЦП II-го типа.

Ниже приведено описание компонентов, которые потребуются при написании пользовательской программы.

В момент вызова **Synchronize** каждый модуль ЦП синхронизирует области резервируемых данных. Существуют два типа данных резервирования: **SharedMemory** и **CrossMemory**, которые регистрируются с помощью одноименных функциональных блоков (ФБ) из библиотеки **PsRedundancy**.

Блок **SharedMemory** при синхронизации всегда копирует указанные данные от ведущего модуля ЦП к ведомому, заменяя данные, которые были на ведомом модуле ЦП до синхронизации (Рисунок 38).

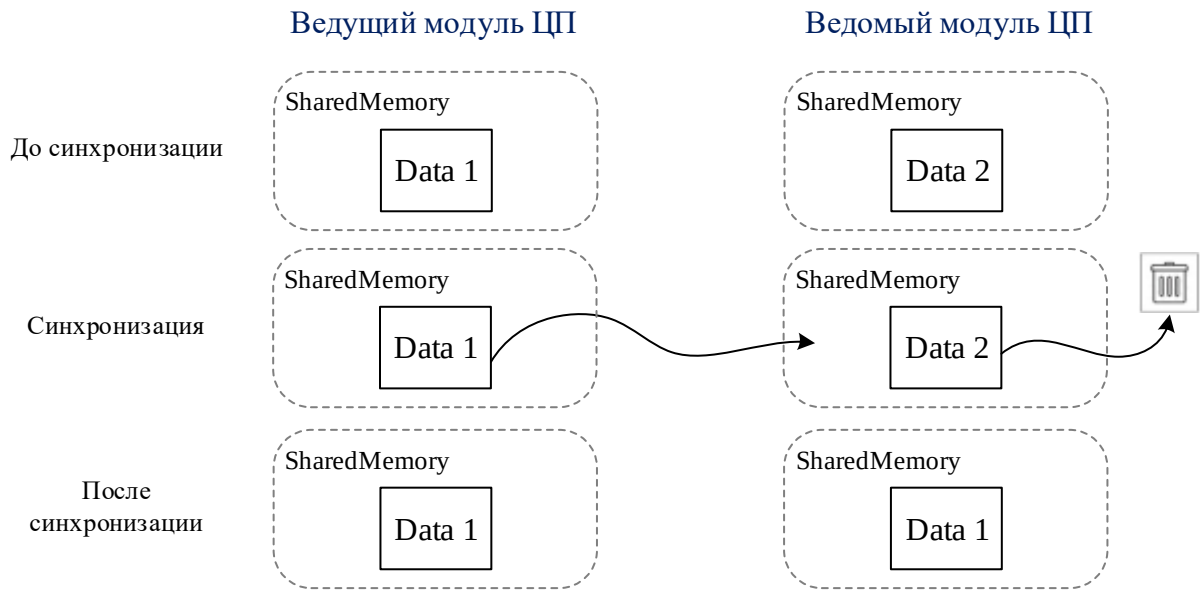

Рисунок 38 – Блок SharedMemory

Блок **CrossMemory** при синхронизации осуществляет обмен данными между модулями ЦП (Рисунок 39). С его помощью, например, в схеме полного резервирования можно сравнить локальное состояние входа N с состоянием входа N на модуле ЦП-партнере.

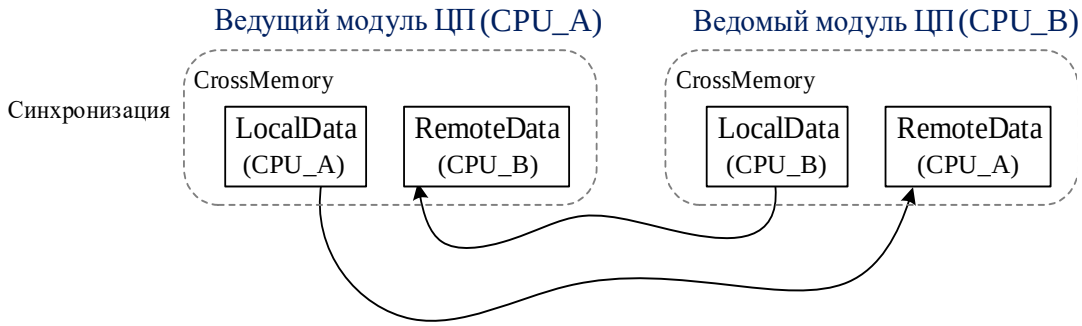

Рисунок 39 – Блок CrossMemory

Каждый вызов функции **Synchronize** приводит к обмену данными между модулями ЦП и синхронизации цикла для задачи, в которой вызван **Synchronize**.

Если требуется статистика о работе алгоритма синхронизации, то можно воспользоваться функцией **Synchronize4,** аналогичной **Synchronize**. Она дополнительно требует указания еще одного входного параметра – ссылки на объект типа **PsRedundancy.TStat2** (статистика). После вызова **Synchronize4** в этот объект будет помещена информация о времени, затраченном на синхронизацию, объеме синхронизируемых данных, состоянии и т.п. Статистику также можно посмотреть в настройках объекта **Redundancy** на вкладке **Мониторинг**.

Функция **SwitchToStandby** – это предложение партнеру взять управление на себя. Возвращает логическое значение *True*, если передача успешна. Значение *False* возвращается, если модуль ЦП-партнер отказался стать ведущим, например, из-за того, что поднят флаг ошибки по шине (неисправен какой-либо из модулей).

Текущий режим модуля ЦП можно увидеть с помощью функции **GetMode**. Возвращаемые значения описаны в структуре **RedMode** библиотеки **PsRedundancy**, текстовое представление также показано в примере.

IsCpuA признак идентификации ЦП как CPU_A или CPU_B. Значение флага не зависит от текущего режима модуля ЦП в схеме резервирования и на одном модуле ЦП всегда *True*, на другом – *False*.

Пример программного кода:

Объявление

```
PROGRAM PLC_PRG
VAR
    stats : PsRedundancy.TStat2;
    redmode : SINT;

SharedVariable : STRING(80);
    SharedVariableAfterSync : STRING(80);
//значение общей переменной после синхронизации
    LocalVariable : STRING(20);
    RemoteVariable : STRING(20);
//данные копируются всегда с ведущего модуля ЦП
    shared_data : PsRedundancy.SharedMemory(SIZEOF(SharedVariable),
ADR(SharedVariable));
//обмен данными между модулями ЦП
cross_data  : PsRedundancy.CrossMemory(SIZEOF(LocalVariable),
ADR(LocalVariable), ADR(RemoteVariable));
//указание синхронизации области М-памяти
    shMWRedMem : PsRedundancy.SharedMemory(ADR(%MW10000) - ADR(%MW0)
+ SIZEOF(%MW10000), ADR(%MW0));

    wsModeDescription : WSTRING;
IsStateActive : BOOL;
IsDefaultPlc : BOOL;
END_VAR
```

Реализация

```
IF PsRedundancy.IsCpuA() THEN
    SharedVariable:= 'DefaultPLC';
    LocalVariable := 'LocalVariable1';
ELSE
    SharedVariable:= 'Non DefaultPLC';
    LocalVariable := 'LocalVariable2';
```

```
END_IF

PsRedundancy.Synchronize4(FALSE, stats); //функция синхронизации
SharedVariableAfterSync := SharedVariable;
Redmode     := PsRedundancy.GetMode();
                        //состояние модуля ЦП в схеме синхронизации
CASE redmode OF
    PsRedundancy.CONN_ERROR : wsModeDescription := "CONN_ERROR/Ошибка соединения ";
    PsRedundancy.DISABLED     : wsModeDescription := "DISABLED/Выключено";
    PsRedundancy.BOOTUP : wsModeDescription := "BOOTUP/Включение";
    PsRedundancy.SYNC         : wsModeDescription := "SYNC/Синхронизация";
    PsRedundancy.STANDBY: wsModeDescription := "STANDBY/Ведомый";
    PsRedundancy.ACTIVE_STANDALONE  : wsModeDescription :=
                                    "ACTIVE_STANDALONE/Автономный";
    PsRedundancy.ACTIVE : wsModeDescription := "ACTIVE/Ведущий";
ELSE
    wsModeDescription   := "????";
END_CASE
IsStateActive     := (redmode=PsRedundancy.ACTIVE) OR
            (redmode=PsRedundancy.ACTIVE_STANDALONE);
IsDefaultPlc      := PsRedundancy.IsCpuA();  //для отличия модулей ЦП
```

Настройка резервирования с версии СПО 1.7 (Redundancy OS)

Установите на компьютер прикладное программное обеспечение **Astra.IDE**. Описание процесса установки программы, а также инструкции по работе с программой приведены в документе «Программное обеспечение Astra.IDE. Руководство пользователя». Программа установки и документация доступны на сайте www.reglab.ru.

ВНИМАНИЕ!

Перед обновлением СПО до версий 1.7.0.x и 1.7.1.x, обязательно ознакомьтесь с описанием в документе «Программное обеспечение Astra.IDE. Руководство пользователя» в разделе «Обновление ПО контроллера. Особенности обновления СПО до 1.7.0.0 и до 1.7.1.0»

Отличительные особенности механизма резервирования с версии СПО 1.7

Поддержка компонента Резервирование OS (Redundancy OS).

Основные моменты:

- основной функционал резервирования вынесен в отдельный поток, выполняемый непосредственно на операционной системе;
- конфигурирование набора резервируемых переменных может производиться через вкладку **Резервирование OS** (**Redundancy OS**) ⇨**Переменные для резервирования**, без необходимости внесения изменений в исходный код проекта. При этом имеется возможность указать резервированные переменные в коде посредством соответствующего атрибута;
- добавлен функционал синхронизации времени между ведущим и ведомым ЦП на базе NTP;
- добавлена возможность резервирования переменных в нескольких задачах, но только одна задача полностью синхронизируется по времени исполнения с ЦП. Для «асинхронных» задач не гарантируется соответствие значений резервированных переменных и циклов задач на обоих ЦП. Т.е. значение переменной, полученной из второго ЦП, может не соответствовать текущему циклу задачи на первом ЦП.

Начало работы

Настройка резервирования состоит из следующих шагов:

1. Добавление в проект и настройка компонента **Резервирование OS** (RedundancyOS);
2. Описание в проекте аппаратной конфигурации резервируемого контроллера;
3. Настройка IP-адресов для резервирования на каждом из модулей ЦП (также на коммуникационных модулях в составе контроллера);
4. Определение синхронизируемых переменных (в коде и/или на соответствующей вкладке).

Настройка п. 2,3 выполняется по аналогии с предыдущей версией резервирования (см. описание выше), изменения коснулись п. 1,4 (см. описание ниже).

Добавление в проект компонента «Резервирование OS»

	**ИНФОРМАЦИЯ** При создании нового проекта с помощью **Мастера конфигурации Regul** на этапе **Дополнительные настройки**, поставив переключатель в поле **Резервирование**, будет автоматически добавлен компонент резервирования – **Резервирование** (**Redundancy**). Начиная с версии 1.7.1.0 автоматически будет добавлен компонент резервирования **Резервирование** (**Redundancy OS**)

Добавить компонент «Резервирование OS» на версиях 1.7.0.x можно одним из следующих способов:

– для добавления компонента **Резервирование OS** в существующий проект поставьте курсор на название приложения (**Application**), нажмите правую кнопку мыши. Появится контекстное меню, в котором выберите **Добавление объекта** ▸ ⇨ **Резервирование OS…** Откроется окно **Добавить Резервирование OS**, где нажмите кнопку ***Добавить*** (Рисунок 40). Компонент **Резервирование OS** будет помещен в дерево устройств.

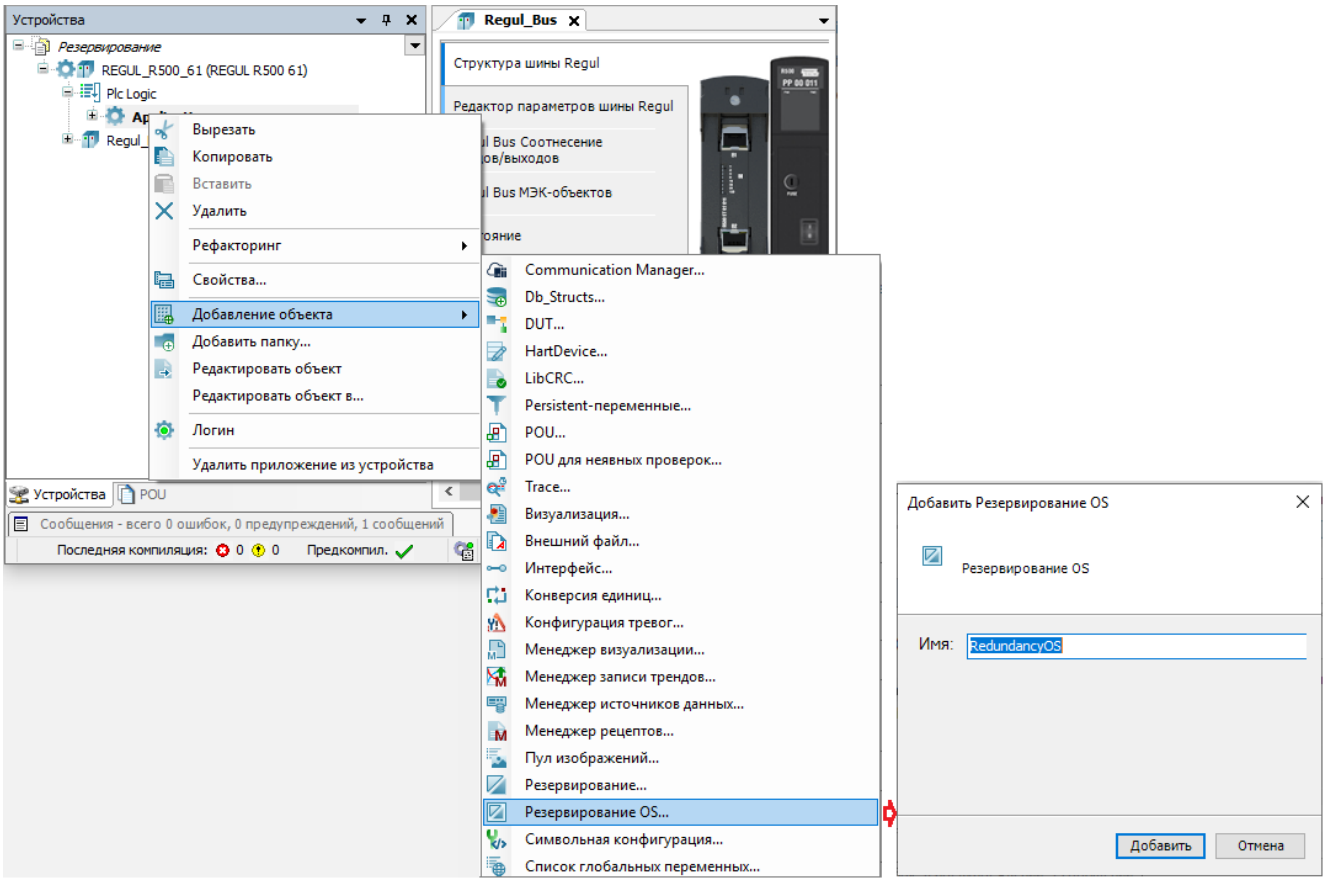

Рисунок 40 – Добавление компонента Redundancy OS

– для добавления компонента **Резервирование OS** в проект, созданный в предыдущей версии СПО (до версии 1.7.0.0), необходимо будет удалить компонент

Резервирование… и добавить новый **Резервирование OS** (как описано выше), осуществив при этом все необходимые настройки заново.

При добавлении компонента Резервирование OS в проект автоматически добавляется библиотека PsRedundancy_OS.

Запуск различных версий резервирования

При загрузке проекта, содержащего объект резервирования, производится проверка на соответствие значения параметра *RunDriverType* в секции [PsRedundancy] конфигурационного файла *runtime.cfg* с объектом резервирования:

– для компонента **Резервирование (Redundancy) -** параметр *RunDriverType* может отсутствовать или иметь значение *RT*:

```
[PsRedundancy]
RunDriverType=RT
```

– для компонента **Резервирование OS (RedundancyOS) -** параметр *RunDriverType* имеет значение *OS*:

```
[PsRedundancy]
RunDriverType=OS
```

Если значение параметра не совпадает с объектом резервирования, то происходит перезапись значения в конфигурационном файле *runtime.cfg* и выводится сообщение: «Тип драйвера резервирования был изменен. Перезагрузите ПЛК»/«Redundancy driver type was changed. Reboot the PLC», т.е. для вступления в силу изменений перезагрузите контроллер путем выключения/включения питания либо командой *reboot* на вкладке **Оболочка ПЛК**.

Резервируемые переменные

При проектировании прикладной программы и определении резервируемых переменных необходимо учитывать, что синхронизируются циклы только одной задачи, имеющей максимальный приоритет после задачи шины. Для остальных задач, если они участвуют в резервировании, моменты синхронизации данных будут определяться их циклами в ведущем ПЛК. При этом совпадение циклов в ведущем и ведомом ПЛК не гарантируется.

Синхронизация данных осуществляется двумя способами:

– **SourceData** (аналог **SharedMemory**) - резервирование данных ведущего модуля ЦП в ведомом модуле ЦП перед стартом цикла,

– **CrossData** (аналог **CrossMemory**) - использование данных в ведущем модуле ЦП из ведомого модуля ЦП, синхронизация перед стартом цикла;

Переключение между режимами производится автоматически – при наличии кросс переменных (RVE) работает алгоритм **CrossData**, в остальных случаях **SourceData**.

Кросс переменная (RVE) – переменная, которая содержит значение резервированной переменной второго ЦП перед началом работы цикла резервирования. Используется в

сочетании с обычной резервированной переменной. Пара: резервированная переменная и RVE переменная - это аналог записи:

```
cross data:
PsRedundancy.CrossMemory(SIZEOF(LocalVariable),ADR(LocalVariable),ADR(RemoteVariable));
```

ВНИМАНИЕ!

Нежелательно резервировать одни и те же данные в разных задачах

При необходимости программной обработки входных данных, до выполнения цикла резервирования в задаче, можно использовать обработчик системного события «AfterReadingInputs» (Рисунок 41).

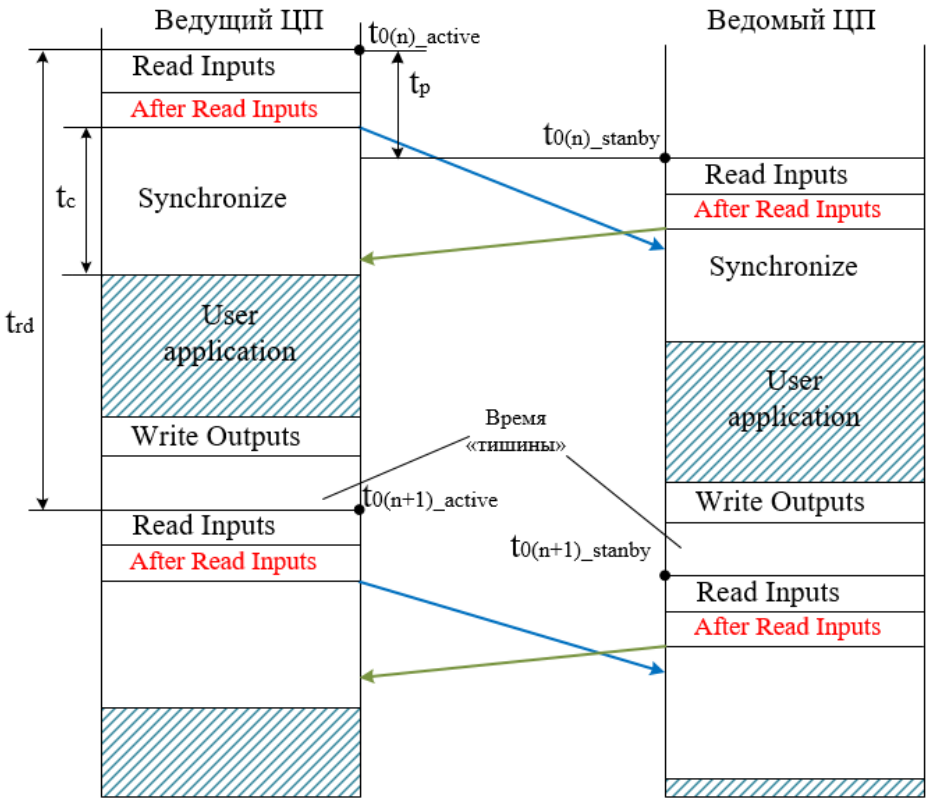

Рисунок 41 – Алгоритм работы задачи резервирования с AfterReadingInputs

Для добавления события откройте вкладку **Конфигурация задач ⇨ Системные события ⇨ +Добавить обработчик событий⇨ AfterReadingInputs** (Рисунок 42)**.**

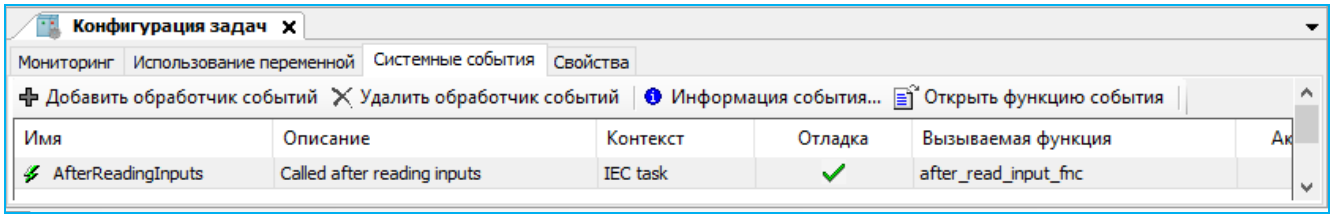

Рисунок 42 – Добавление обработчика события

Для конфигурирования переменных и настройки задач резервирования перейдите на вкладку **RedundancyOS ⇨Переменные для резервирования** (Рисунок 43).

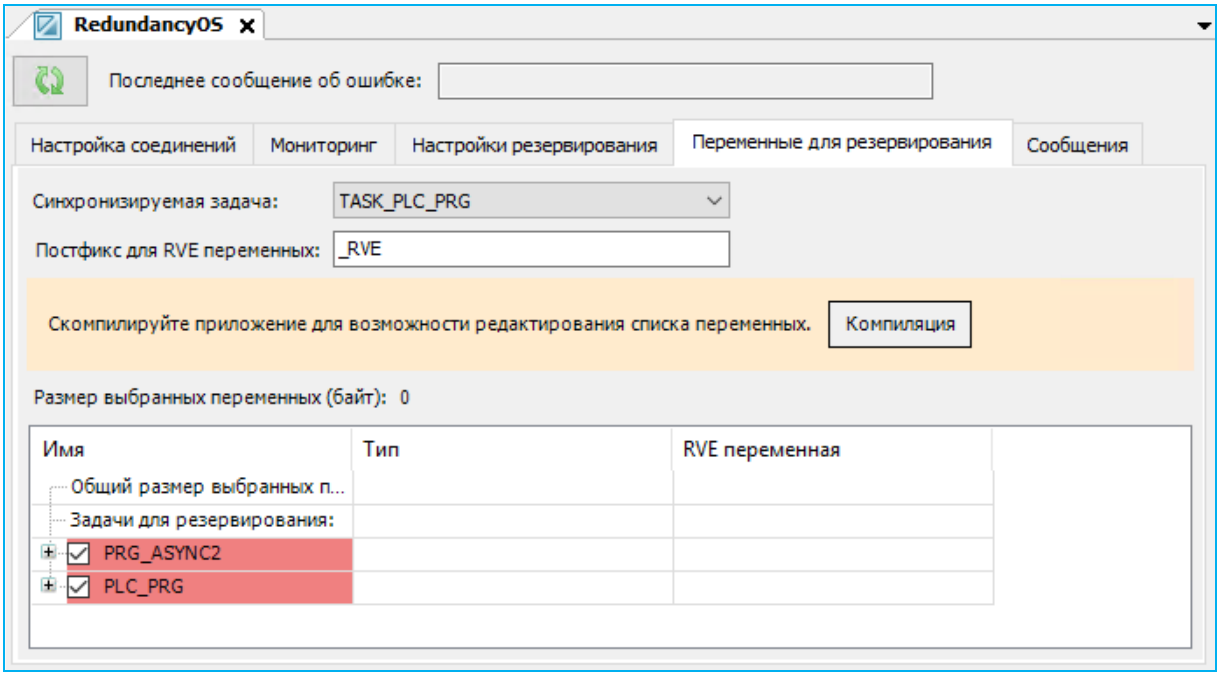

Рисунок 43 – Резервирование. Переменные для резервирования

В поле **Синхронизируемая задача:** из выпадающего списка, выберите синхронизируемую задачу – задача с максимальным приоритетом, после задачи RegulBus. В случае использования RegulBus_OS, задача резервирования будет иметь максимальный приоритет во всем проекте (например: TASK_PLC_PRG; см. рисунок 44).

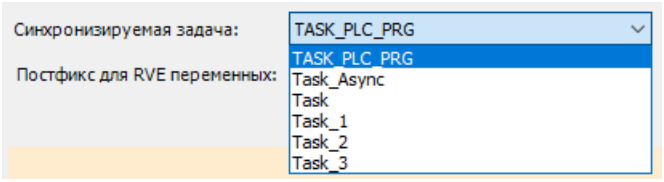

Рисунок 44 – Синхронизируемая задача

Для проверки на синтаксические ошибки и осуществления редактирования списка переменных нажмите на кнопку *Компиляция* (либо основном меню **Компиляция⇨ Генерировать код**, либо используйте клавишу **F11**). Информация, предупреждения и сообщения об ошибках будут отображаться в окне сообщений, которое по умолчанию располагается в нижней части окна.

Задайте постфикс для переменных в поле **Постфикс для RVE переменных**. Чтобы в таблице автоматически проставились RVE переменные, согласно выбранному постфиксу, нажмите кнопку *Компиляция (F11).*

Можно отметить «асинхронные» задачи, в которых будут резервироваться переменные («значок синхронизации» изменится с  на ). Для этого установите флажок ☑ в строке **Задачи для резервирования** напротив необходимой «асинхронной» задачи (Рисунок 45). В этом случае не гарантируется соответствие значений резервированных переменных в этих задачах и циклов задач на обоих ЦП. Из-за вытеснения более приоритетной синхронной задачей, значение переменной, полученной из одного ЦП, может не соответствовать текущему циклу задачи на другом.

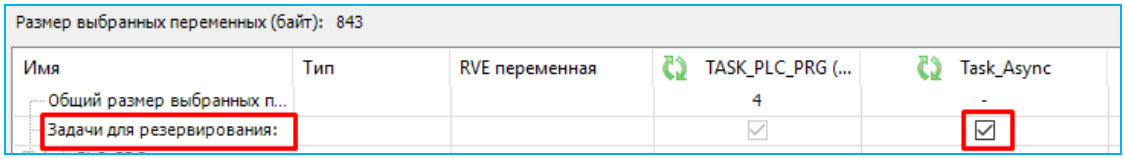

Рисунок 45 – Задачи для резервирования

Выберите переменные для резервирования, установив флажок ☑ в строке с названием соответствующей переменной (Рисунок 46)

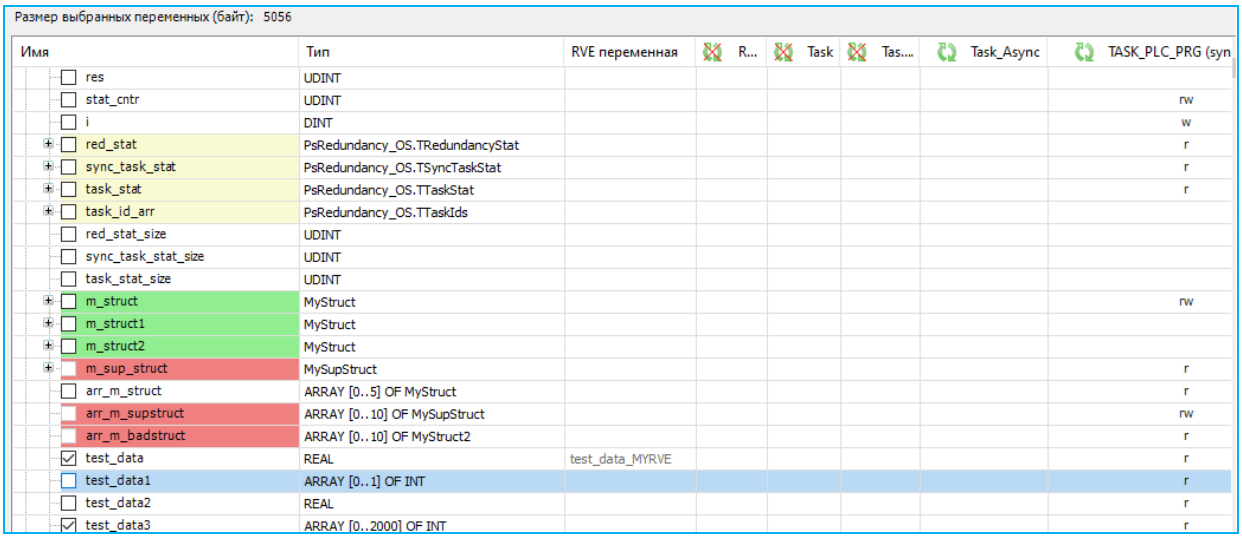

Рисунок 46 – Таблица переменных

Над таблицей отображается строка с указанием размера резервируемых данных (байт), а в первой строке таблицы (**Общий размер выбранных переменных по задачам:**), в колонке каждой задачи отображается размер синхронизируемых данных задачи (Рисунок 47).

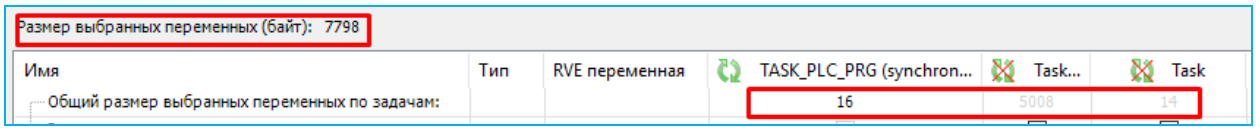

Рисунок 47 – Размер резервируемых и синхронизируемых данных

Участвовать в резервировании могут только переменные, имеющие один из атрибутов r/w/rw в колонке соответствующей задачи.

Резервированные переменные проекта в таблице могут быть выделены цветом по следующим признакам:

– зеленым цветом выделены экземпляры типов/ФБ, которые были проверены разработчиками ПО резервирования («РегЛаб») и гарантированно применимы для резервирования. Для таких типов и ФБ предусмотрен атрибут «ps.sync» (например, см. рисунок 48)

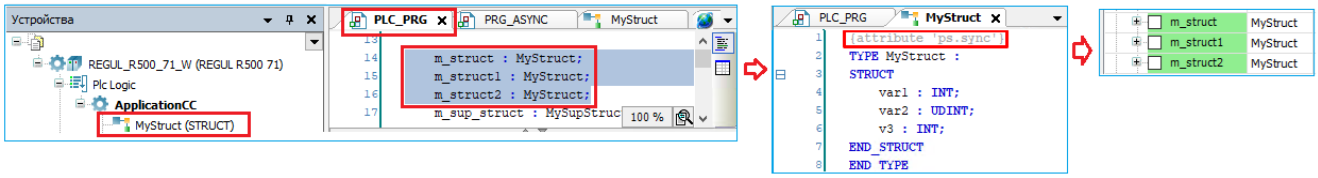

Рисунок 48 – Пример. Атрибут «ps.sync»

– желтым цветом выделены экземпляры типов/ФБ, которые прошли только автоматическую проверку на наличие недопустимых объектов для резервирования;

– красным цветом выделены экземпляры, в которых компилятор обнаружил недопустимые объекты для резервирования. Такие переменные не могут быть отмечены, как резервируемые. Массивы переменных таких типов также недоступны для резервирования.

ИНФОРМАЦИЯ

Не резервируются: REFERENCE, POINTER, массивы этих типов, массивы структур содержащих эти типы

– экземпляры простых типов не окрашены.

Можно указать переменные резервирования в коде проекта, для этого применяется атрибут «ps.add_redundancy». Переменные с таким атрибутом будут автоматически отмечены в таблице осветлённым флажком без возможности редактирования (Рисунок 49). При этом сохраняются все правила относительно недопустимых объектов для синхронизации.

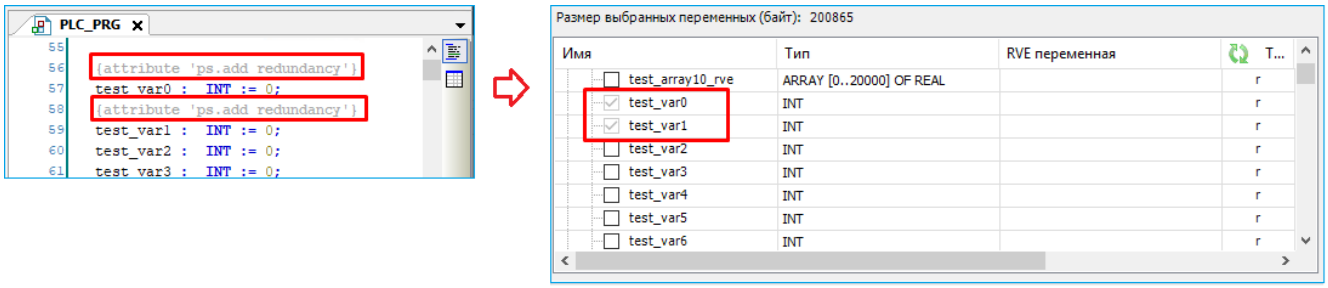

Рисунок 49 – Пример. Атрибут «ps.add_redundancy»

Работа с типами

Для работы со всеми экземплярами одного типа, поместите курсор на ячейку с наименованием типа переменной, нажмите правую кнопку мыши. В появившемся контекстном меню выберите пункт «Выберите компоненты типа»(«Select type’s components») (Рисунок 50).

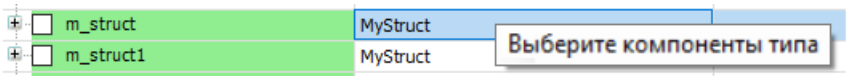

Рисунок 50 – Контекстное меню

Откроется окно с перечнем всех типов объекта. Выберите часть или весь объект типа целиком. Резервирование будет применено ко всем переменным данного типа в проекте.

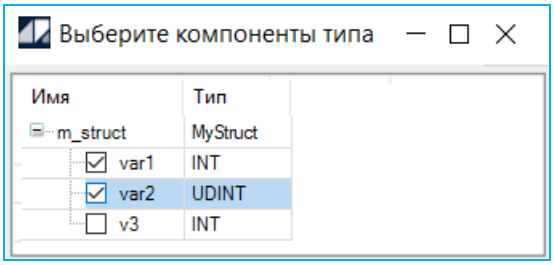

Рисунок 51 – Окно выбора типа объекта/компонента

Работа с кросс переменными (RVE)

Для работы с кросс переменными (RVE) необходимо объявить переменную в коде и указать ее в столбце RVE для соответствующей ей резервированной переменной (Рисунок 52). При этом производится автоматическая проверка на соответствие типов.

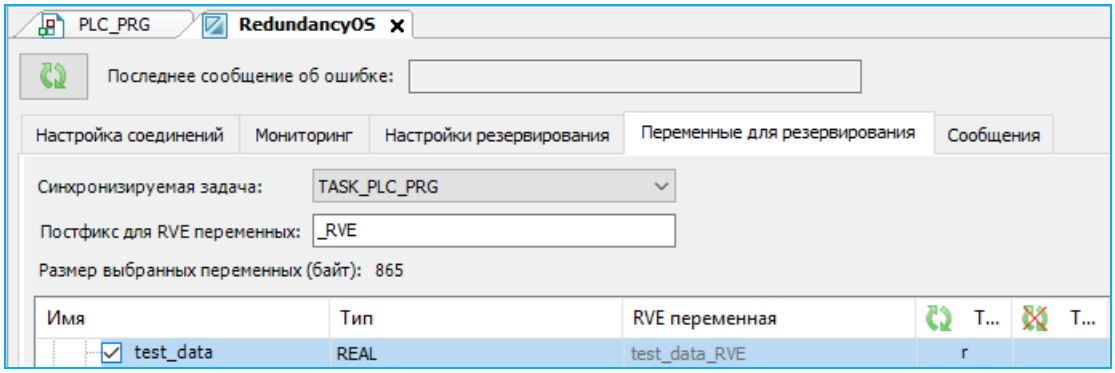

Рисунок 52

В таблице будет заблокирована возможность отметить кросс переменную RVE как резервированную (Рисунок 53).

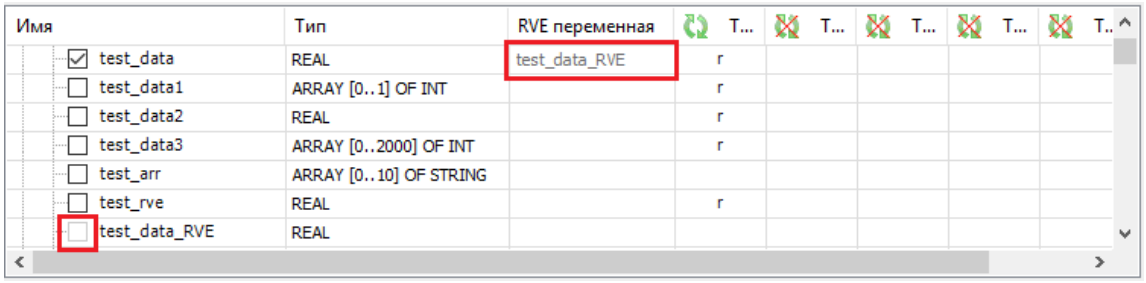

Рисунок 53

Если имя RVE переменной в проекте соответствует паттерну [имя резервированной переменной] [постфикс RVE], то такие переменные автоматически подставляются в колонку **RVE переменная**. Постфикс для RVE переменных может быть задан в поле **Постфикс для RVE переменных** (Рисунок 54).

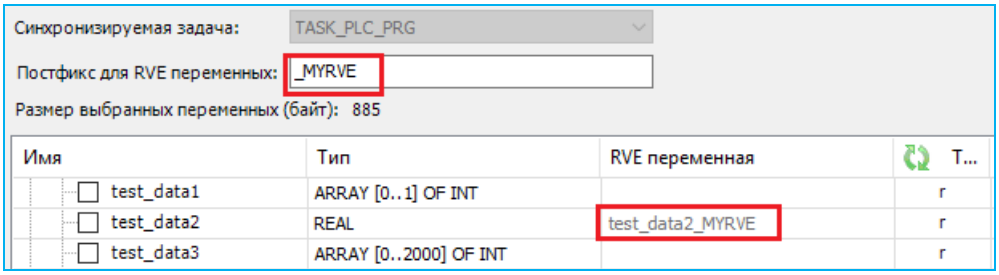

Рисунок 54

Идентификаторы доступа к переменным в резервированной задаче обозначены в таблице символами черного цвета: r - только для чтения; rw - чтение и запись; w- только для записи, отсутствие идентификатора означает, что переменная не используется в задаче. Если задача не участвует в резервировании, все идентификаторы доступа в ее столбце окрашены серым.

Обслуживание резервированного контроллера

Настройка и диагностика соединения

В окне дерева устройств дважды щелкните левой кнопкой мыши на объекте **Резервирование**. Откроется вкладка (окно) настроек этого объекта. По умолчанию открыта внутренняя вкладка **Настройка соединений** (Рисунок 55). Здесь вы можете установить путь к каждому из модулей ЦП и просмотреть текущее состояние модулей ЦП.

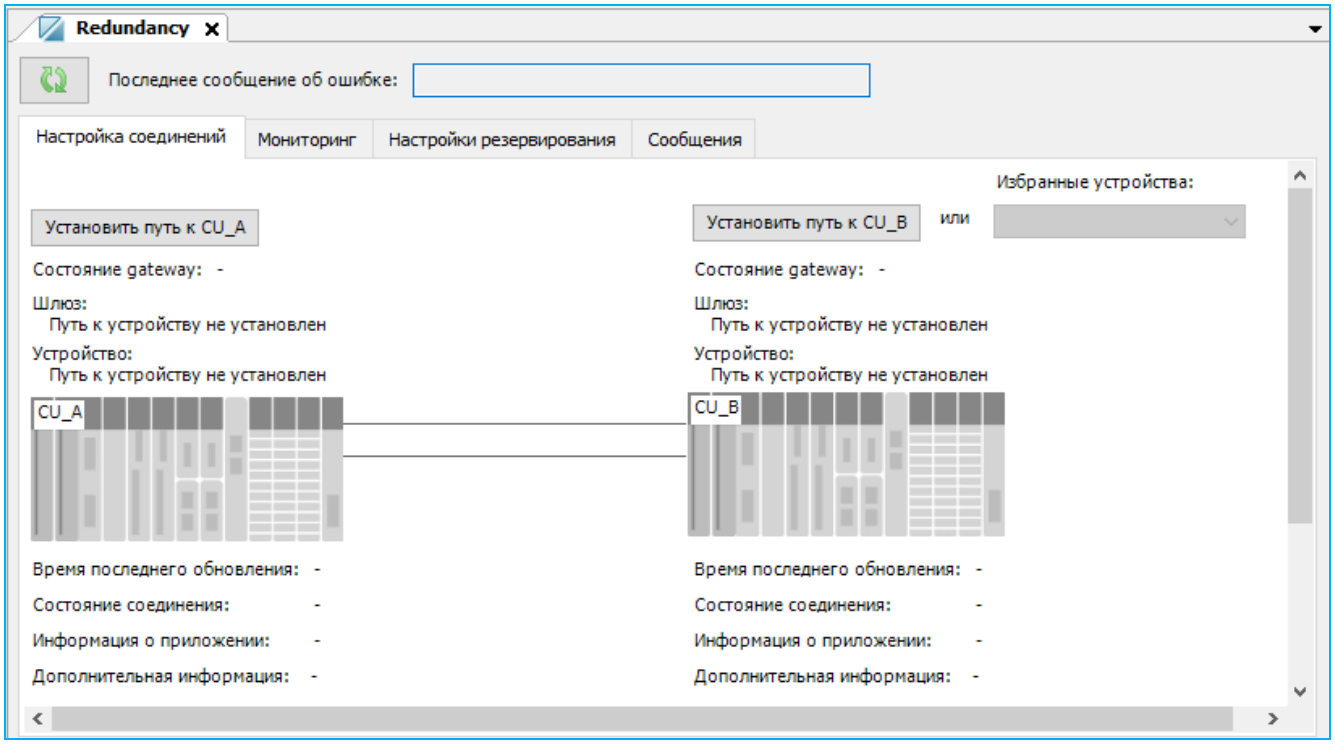

Рисунок 55 – Резервирование. Настройка соединений

Задайте модуль ЦП, участвующий в схеме резервирования, в который будет загружаться проект. Если ранее в проекте вы уже подключились к этому модулю ЦП, то он и будет указан как «CPU_A». Если ранее не подключались, то нажмите кнопку ***Установить путь к CPU_A***. Откроется вкладка (окно) настроек контроллера, внутренняя вкладка **Установки соединения**. Нажмите кнопку **Scan network…**, появится окно **Выбор устройства** (Рисунок 56). Просканируйте сеть, в левой части отобразится список найденных в этой сети контроллеров. Выберите нужный контроллер, нажмите кнопку ***OK***.

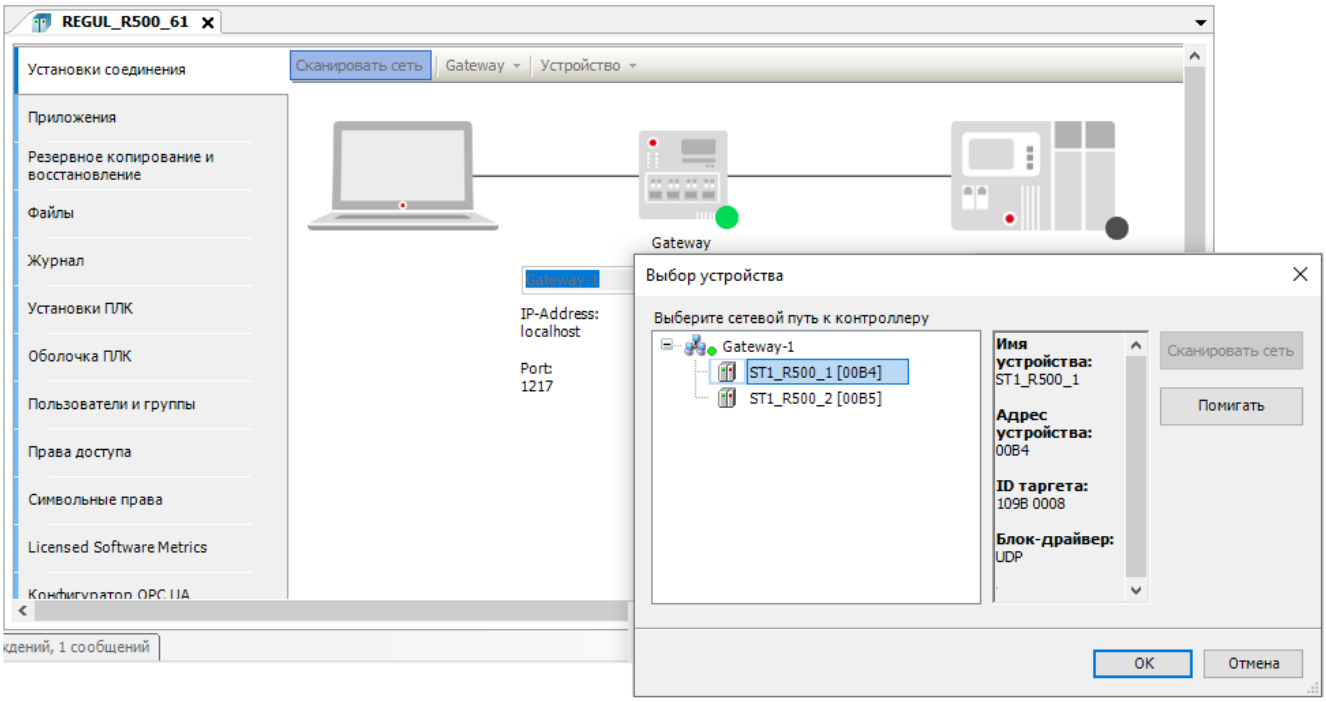

Рисунок 56 – Установка соединения с контроллером

На вкладке настроек объекта **Резервирование** нажмите кнопку ***Установить путь к CPU_B*** и установите соединение со вторым модулем ЦП, как описано выше.

После того, как определены модули ЦП-партнеры, нажмите кнопку  (***Обновить***) в левом верхнем углу вкладки. На экране отобразится информация о состоянии связи между модулями и их текущая роль в схеме резервирования. Кроме того, отобразятся данные об устройстве, текущей версии прошивки, а также сведения о прикладной программе (проекте), загруженном в данный момент на каждый из модулей ЦП. Если на обоих модулях ЦП приложения одинаковы, то информация о них выделена зеленым цветом, иначе – красным.

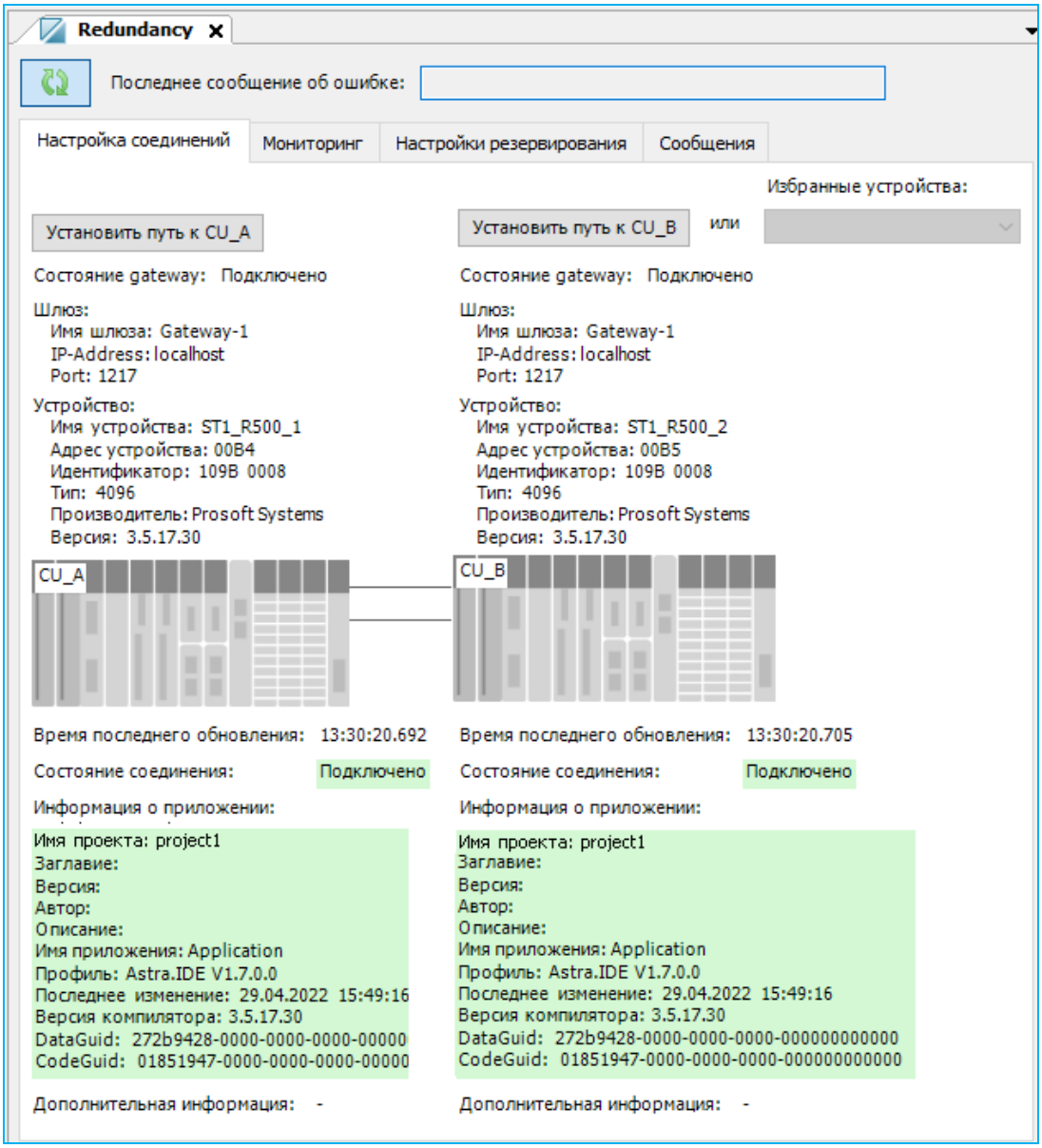

Рисунок 57 – Резервирование. Настройка соединений и текущее состояние модулей ЦП

В дереве устройств значок рядом с устройством будет указывать на его состояние:

– : устройство запущено;

– : устройство пока не запущено или не настроено, либо содержит ошибки;

– : устройство запущено, доступна диагностическая информация;

– : устройство настроено, но не запущено;

– : устройство остановлено; обмен данными с устройством не производится;

– : устройство находится в пассивном (не диагностируемом) состоянии.

В схемах частичного резервирования при подключении к одному из модулей ЦП, модуль ЦП-партнер будет находиться в не диагностируемом состоянии (Рисунок 58).

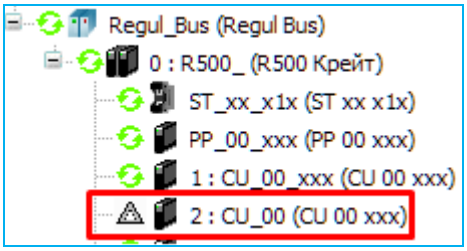

Рисунок 58 – Отображение пассивного состояния модуля ЦП-партнера в онлайн режиме

Настройка резервирования

Для настройки параметров работы модулей ЦП (CU A/B) в резервированном контролере перейдите на вкладку **Настройки резервирования** (Рисунок 59).

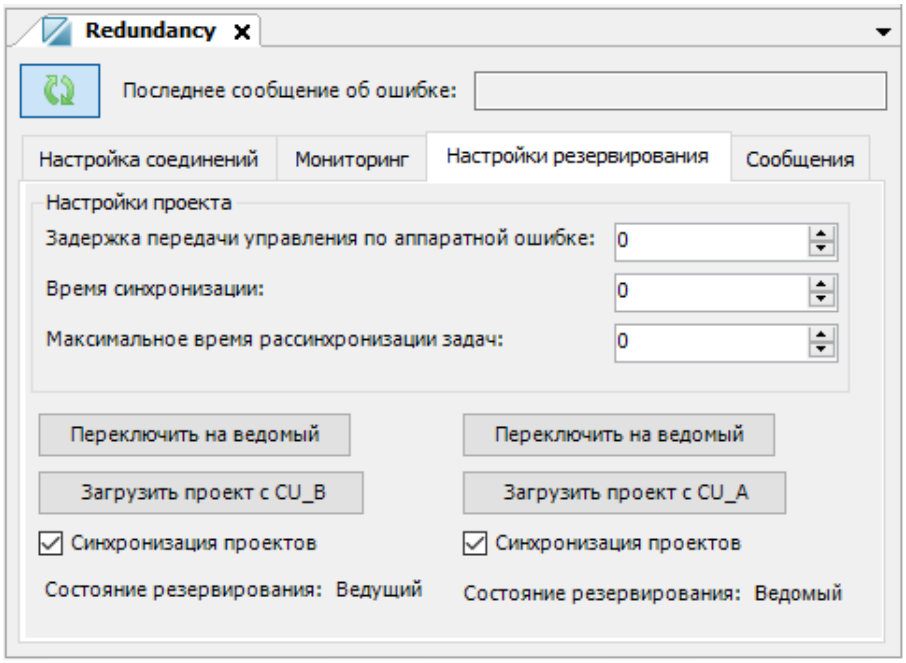

Рисунок 59 – Настройка параметров модулей ЦП (CU_A/B)

Описание настраиваемых параметров, с указанием значений по умолчанию, приведено в таблице 4.

Таблица 4 – Настраиваемые параметры

Параметр	Описание	Значение по умолчанию
Задержка передачи управления по аппаратной ошибке (HwErrorSwitchDelayMs)	Задержка передачи управления по аппаратной ошибке, мс. При значении «0» - задержка отсутствует	0
Время синхронизации (SynchronizeTimeMs)	Время синхронизации, мс. При значении «0» - время рассчитывается автоматически, в зависимости от объема данных, но не менее MaxTaskOffsetMs	0
Максимальное время рассинхронизации задач (MaxTaskOffsetMs)	Максимальное время рассинхронизации задач, мс. При значении «0» - время рассчитывается автоматически, но не менее 4 мс	0

Следует обратить внимание на параметр HwErrorSwitchDelayMs – задержка передачи управления по аппаратной ошибке. По умолчанию система настроена на моментальную передачу управления между модулями ЦП. При использовании схем частичного резервирования такое поведение может приводить к лишним переключениям **Ведущий** – **Ведомый** из-за того, что ведущий модуль ЦП иногда получает оповещение об ошибке раньше ведомого. Для исключения подобных переключений рекомендуется установить задержку передачи управления по аппаратной ошибке на время, как минимум равное временному интервалу опроса модулей (см. в разделе «Скорость реакции на аварийные события»).

Кнопки ***Переключить на ведомый (Switch to standbay)*** позволяют вручную изменить статус модуля ЦП, передав управление от ведущего модуля ЦП к ведомому.

Кнопки ***Загрузить проект с CU_B / с CU_A (Download project from CU_B /…CU_A)*** позволяют в ручном режиме загрузить проект с модуля ЦП-партнера (синхронизировать проекты).

Пользователь может выключить / включить автоматическую синхронизацию проектов путем снятия / установки флажка в поле ***Синхронизация проекта (Sync project)***.

В поле **Состояние резервирования** отображается информация о состоянии, в котором находится соответствующий ЦП:

- **Ведущий** (Active);
- **Включение** (Bootup);
- **Ошибка соединения** (Conn Error);
- **Выключено** (Disabled);
- **Автономный** (Standalone).
- **Ведомый** (Standby);
- **Синхронизация** (Sync);

Данную информацию также можно увидеть в **строке состояния** при подключении в режиме онлайн (Рисунок 60), но только для того ЦП, к которому в настоящий момент подключена среда разработки.

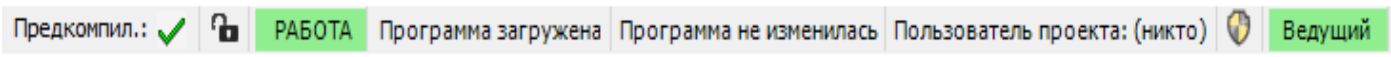

Рисунок 60 – Отображение текущего состояния резервирования в строке состояния

Синхронизация времени для компонента «Резервирование OS»

Начиная с версии СПО 1.7, в резервированном ПЛК появилась возможность настроить синхронизацию времени. Для этого перейдите на вкладку **Настройки резервирования** (Рисунок 61).

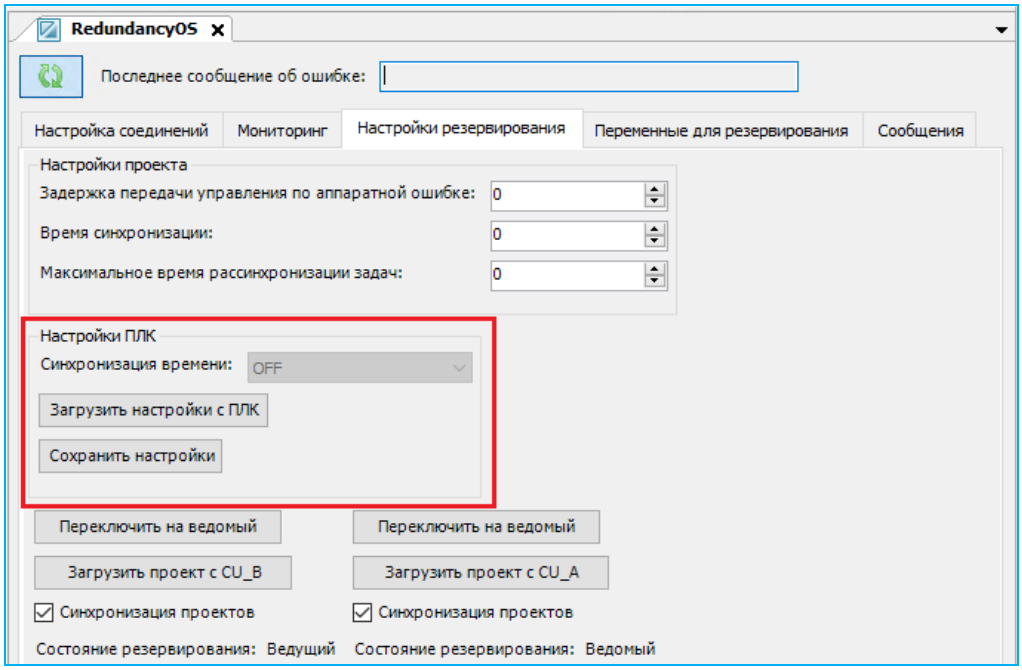

Рисунок 61

В блоке **Настройки ПЛК** выполните следующие действия:

– нажмите на кнопку **Загрузить настройки с ПЛК**. Станет активным поле **Синхронизация времени:**. Информация будет считываться с обоих ЦП-партнеров. Если настройки различаются, то появится сообщение об ошибке**;**

– выберите один из четырех режимов осуществления синхронизации времени (Рисунок 62):

- **OFF** - синхронизация отключена;
- **CPUA** - синхронизация с CU_A;
- **CPUB** - синхронизация с CU_B;
- **ACTIVE** - синхронизация с активным CU.

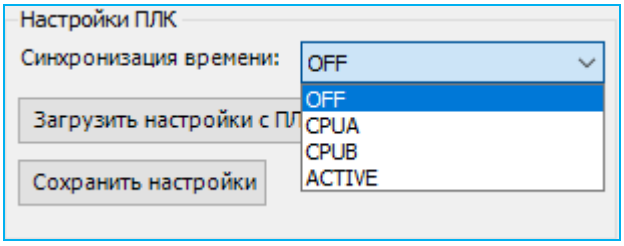

Рисунок 62

– нажмите кнопку ***Сохранить настройки***. Выбранные настройки загружаются в оба ЦП сразу.

Значение параметра сохраняется в ПЛК в конфигурационный файл /etc/runtime.cfg секции [PsRedundancy] поля TimeSynchronization. По умолчанию задано следующее значение:

[PsRedundancy]

TimeSynchronization = OFF

Если поле отсутствует, синхронизация времени отключена.

Мониторинг текущего состояния

Для просмотра статистики по текущему состоянию резервирования перейдите на вкладку **Мониторинг** (Рисунок 63). Для автоматического обновления нажмите кнопку  (***Обновить***) в левом верхнем углу вкладки .

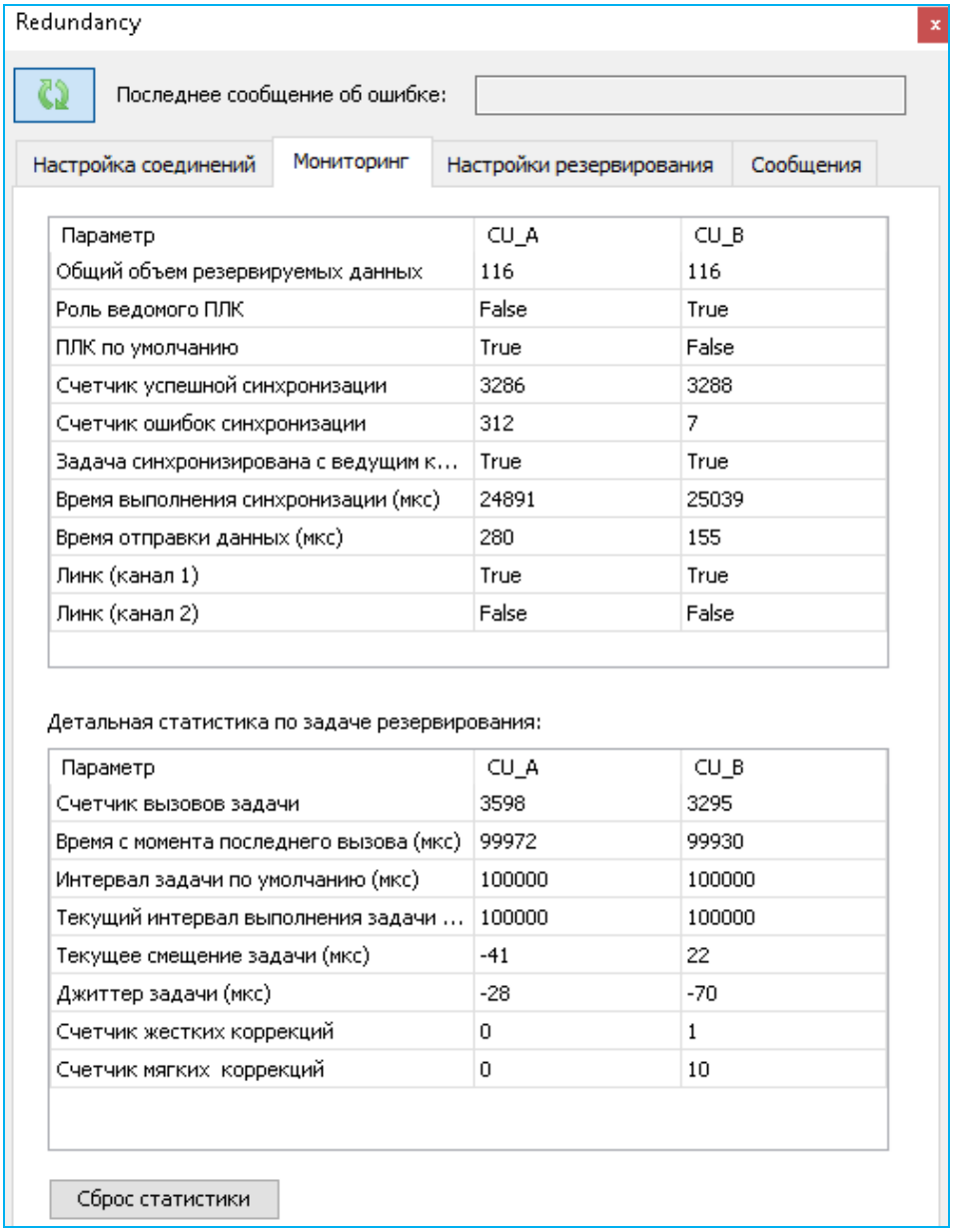

Рисунок 63 – Мониторинг текущего состояния

Верхняя таблица отображает общие параметры резервирования:

- **ПЛК по умолчанию** – отображает результат вызова функции **IsCpuA()** библиотеки **PsRedundancy**. Используется для отличия одного модуля ЦП от другого в программном коде;
- **Счетчик успешной синхронизации** должен нарастать синхронно с количеством вызовов задачи резервирования;

- **Счетчик ошибок синхронизации** на обоих модулях ЦП не должен нарастать в штатном режиме. Ненулевое количество ошибок синхронизации может быть связано с разным временем старта модулей ЦП. Для наблюдений за ним можно воспользоваться сбросом статистики;
- **Время выполнения синхронизации (мкс)** – фактическое время, затраченное на синхронизацию;
- **Линк (канал 1)** и **Линк (канал 2)** показывают наличие связи между модулями ЦП, штатное значение *True* (связь есть).

Детальная статистика по задаче резервирования содержит следующую информацию:

- **Счетчик вызовов задачи** инкрементируется при каждом вызове;
- **Интервал задачи по умолчанию (мкс)** показывает периодичность вызовов задачи, указанную в ее настройках;
- **Текущий интервал выполнения задачи** отображает скорректированное значение периодичности вызова задачи;
- **Текущее смещение задачи (мкс)** отображает разницу во времени между вызовами задачи резервирования на ведомом и ведущем центральных процессорах;
- **Джиттер задачи** показывает, насколько время фактического вызова задачи отличается от планового. Стабильно положительный джиттер говорит о том, что задача не успевает выполняться (включая синхронизацию) за отведенное время. С другой стороны, большой разброс значений джиттера может привести к принятию решения о том, что рабочие циклы модулей ЦП не синхронизированы, и, в итоге, к так называемой жесткой коррекции. Такая коррекция возникает, например, при старте модулей ЦП.
- **Счетчик жестких коррекций** показывает, сколько жестких коррекций было произведено с момента запуска ЦП. Жесткой коррекцией считается пропуск одного цикла задачи на ведомом ЦП, происходящий при превышении времени рассинхронизации tp.
- **Счетчик мягких коррекций** показывает, сколько мягких коррекций было произведено с момента запуска ЦП. Мягкая коррекция производится при рассогласовании времени начала цикла на ведомом ЦП относительно ведущего ЦП более чем на 1 мс, но менее чем время рассинхронизации tp. При этом происходит корректировка времени начала цикла задачи на ведомом ЦП на 1 мс за один цикл без пропуска цикла.

Мониторинг текущего состояния для компонента «Резервирование OS»

На вкладке **Мониторинг** можно просмотреть статистику: по текущему состоянию резервирования, отдельно по синхронной задаче и на выбор по любой из задач, участвующих в резервировании (Рисунок 64).

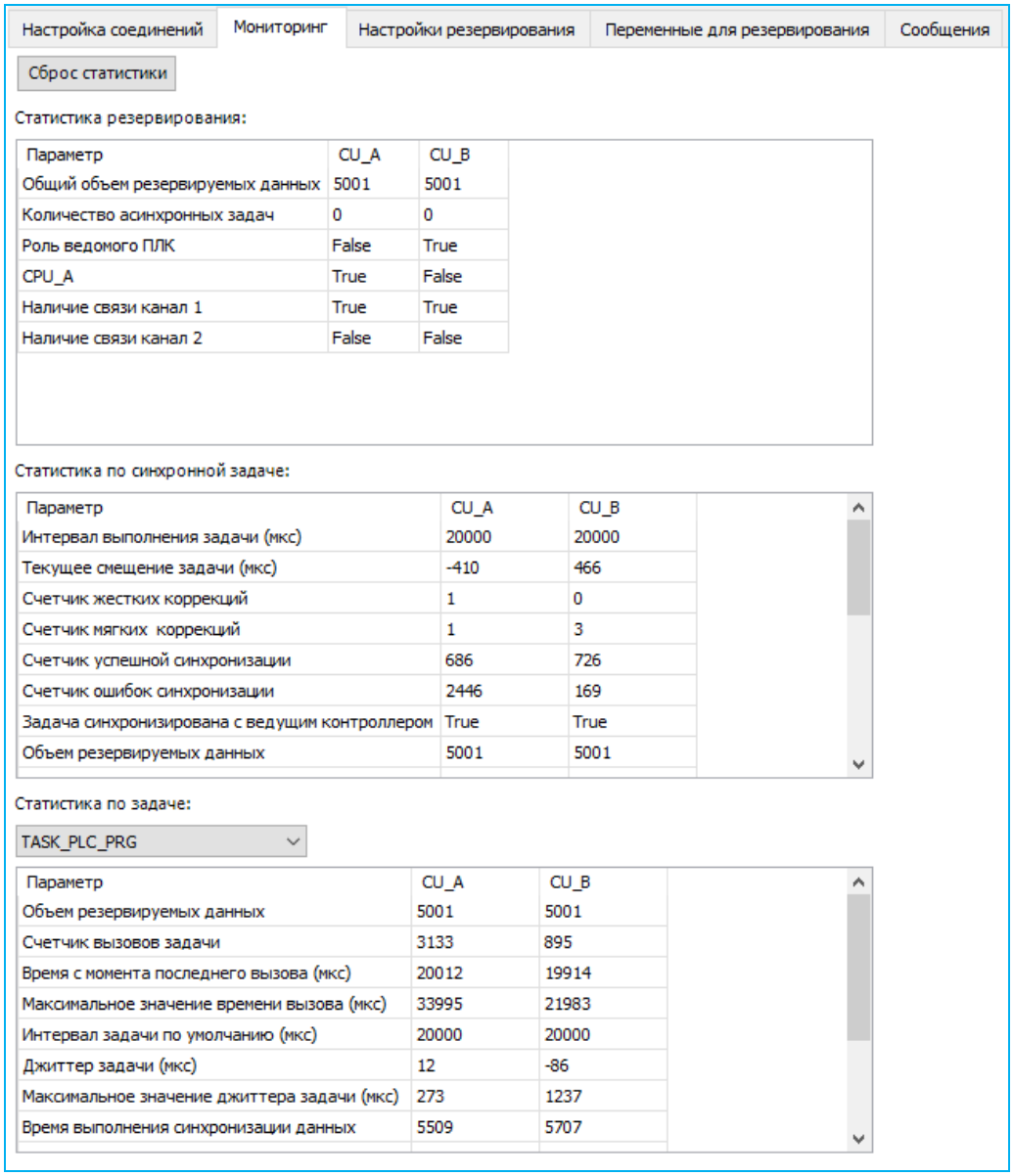

Рисунок 64 – Мониторинг текущего состояния

Журналы сообщений

Для просмотра информации, читаемой из журналов сообщений модулей ЦП, перейдите на вкладку **Сообщения**. Для автоматического обновления информации нажмите кнопку  (***Обновить***) в левом верхнем углу вкладки (Рисунок 65).

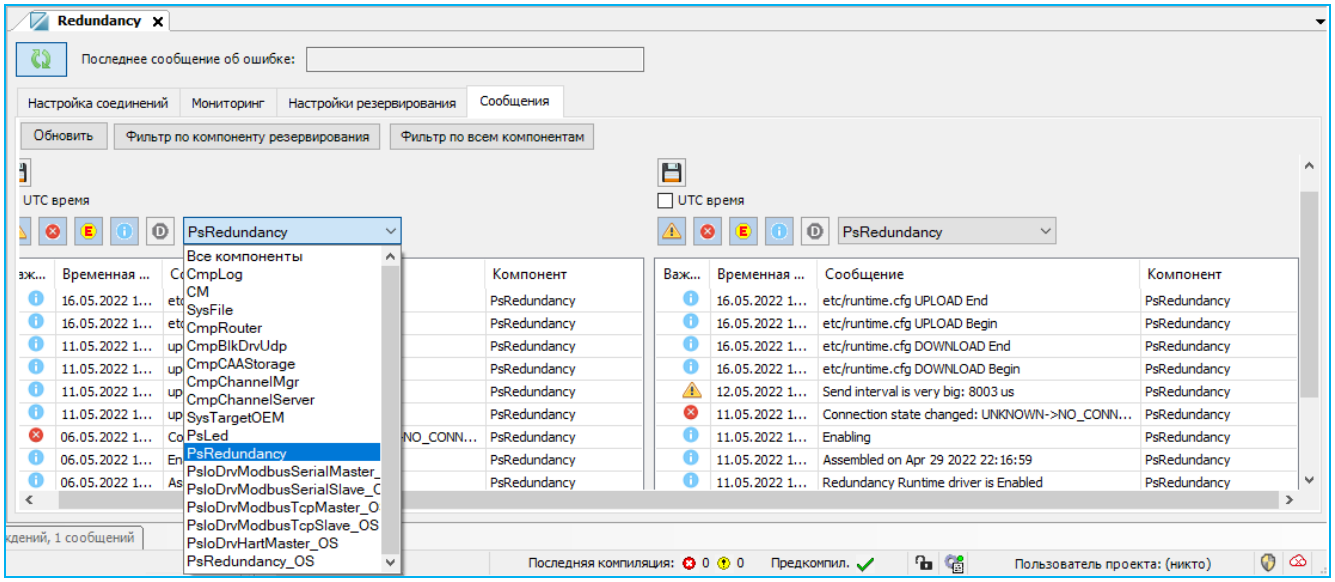

Рисунок 65 – Резервирование. Журналы сообщений

Каждая запись в журнале относится к одной из категорий:

– : предупреждение;

– : ошибка;

– : исключение;

– : сообщение;

– : отладка.

Для отображения в журнале записей, сортированных по конкретным категориям, выберите и нажмите на соответствующую категории кнопку (из выделенной красным цветом области), тем самым включив/выключив сортировку.

В журнале предусмотрена возможность быстрого фильтрования сообщений о работе библиотеки **PsRedundancy (PsRedundancyOS)**. Для этого нажмите кнопку ***Фильтр по компоненту резервирования.*** В списке **Компонент** будет автоматически выбран компонент **PsRedundancy (PsRedundancyOS)**, и в журнале будут показаны сообщения системы, относящиеся только к этому компоненту. Для быстрого выделения всех сообщений системы нажмите кнопку ***Фильтр по всем компонентам***. Для просмотра сообщений о работе любой из библиотек, достаточно выбрать необходимую библиотеку из выпадающего списка, как показано на рисунке 65.

Для очистки журнала нажмите кнопку ***Очистить***.

Для сохранения журнала в формате csv нажмите кнопку . Откроется окно для выбора места сохранения файла.

Обновление приложений резервируемого контроллера

В процессе разработки прикладной программы, а также в процессе испытаний и пусконаладочных работ, зачастую, требуется проводить большое количество процедур обновления проекта. Для облегчения выполнения обновления прикладного программного обеспечения в контроллере предусмотрена процедура автосинхронизации проектов, которая позволяет при загрузке новой версии проекта на один из модулей ЦП, автоматически загружать ее на модуль ЦП-партнера.

Функция автосинхронизации проектов по умолчанию включена, но пользователь может ее выключить на вкладке **Настройка резервирования**, либо использовать объект типа **TConfigControl**:

```
redconfig   : PsRedundancy.TConfigControl := (AutoSyncApplication := 0)
```

ВНИМАНИЕ!

Функция автосинхронизации проектов работает только при полной загрузке проекта. При процедуре онлайн-изменения автосинхронизация проекта не происходит

При включении функции автосинхронизации автоматически обновляется и файл проекта в исходном коде, в случае его наличия в загружаемом проекте (см. в документе «Программное обеспечение Astra.IDE. Руководство пользователя» в разделе «Отладка проекта. Компиляция и загрузка приложения в контроллер»).

В контроллерах серии Regul предусматривается возможность безударного (с сохранением технологических параметров и без прерывания управления технологическим процессом) обновления прикладного программного обеспечения.

В зависимости от объема внесенных в проект изменений, есть два способа безударного обновления прикладного программного обеспечения. Если изменения затрагивают только код программы, но не меняют конфигурацию системы, то возможно проведение онлайн-изменения прикладного программного обеспечения. Онлайн-изменение возможно при следующих изменениях в проекте:

- добавление нового типа и его экземпляра;
- изменение существующего типа:
  - добавление внутренних переменных;
  - удаление внутренних переменных;
  - изменение типа внутренних переменных.
- добавление / удаление экземпляров;
- изменение привязок во вкладке IOMappling.

При выполнении онлайн - изменения проекта не происходит перезагрузка приложения – в контроллер загружаются только измененные объекты проекта. Изменения будут применены путем переключения на вновь загруженный код между циклами задач. В этом случае дополнения в проект вносятся «на лету». Изменения можно проводить как на ведомом, так и на ведущем модуле ЦП, при этом, в случае проведения процедуры на ведущем модуле ЦП, не происходит передачи управления другому модулю ЦП.

При загрузке измененного проекта в любой из модулей ЦП на экране появляется сообщение: «Код изменен со времени последней загрузки. Что следует сделать?». Далее предлагаются варианты действий, означающие следующее (Рисунок 66):

- **Логин с онлайн-изменением** – только измененные части запущенного проекта загружаются в модуль ЦП;
- **Логин с загрузкой** – это загрузка полностью нового проекта взамен существующего в модуле ЦП;
- **Логин без изменений** – последние изменения проекта не будут загружены, запущенная в модуле ЦП программа сохраняется без изменений.

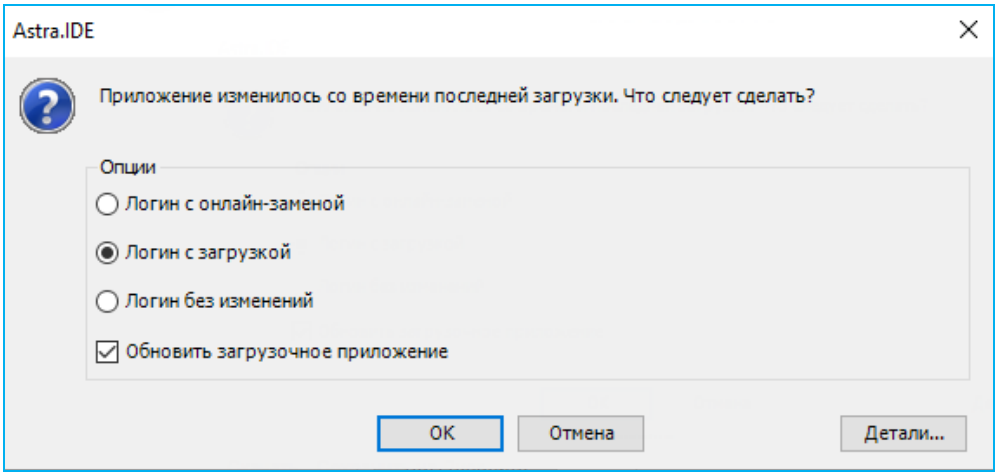

Рисунок 66 – Сообщение при загрузке проекта

	**ВНИМАНИЕ!** При обновлении проекта на ведущем контроллере не снимайте флажок с поля **Обновить загрузочное приложение,** иначе, ведомый контроллер будет уходить в бесконечную перезагрузку

При согласии пользователя на логин с онлайн-изменением происходит загрузка части проекта в один модуль ЦП.

После загрузки обновленного проекта на вкладке настроек объекта **Резервирование** в разделе **Информация о приложении** данные будут выделены красным цветом, что означает, что на модулях ЦП в настоящий момент работают разные приложения.

Нажмите кнопку ***Загрузить проект с*** CPU в блоке того модуля ЦП, на который требуется загрузить обновленный проект.

Необходимо иметь в виду, что даже в случае загрузки с помощью процедуры онлайн-изменения новой версии приложения на один модуль ЦП, загрузка этого проекта на модуль ЦП-партнера, при нажатии кнопки ***Загрузить проект с*** CPU, произойдет по схеме полной загрузки, т.е. будет выполнена перезагрузка этого модуля ЦП.

Обновите информацию на вкладке **Настройка соединений** (кнопка  в левом верхнем углу). Блок информации о приложении поменяет цвет на зеленый – на обоих модулях ЦП находятся идентичные обновленные проекты.

Если при изменении проекта была изменена его конфигурация, и, в том числе, были произведены изменения в:

- параметрах устройства;
- дереве устройств;
- объекте TaskConfiguration;

Тогда процедура онлайн-изменений не доступна и возможно выполнить только полную загрузку (логин с загрузкой) проекта с последующей перезагрузкой приложений.

Процедуру загрузки проекта можно выполнить на любом из модулей ЦП, но в случае обновления проекта на ведущем модуле ЦП произойдет штатная передача управления ведомому модулю ЦП.

Если включена функция автосинхронизации проектов, то достаточно загрузить проект (логин с загрузкой) в один модуль ЦП. Далее этот модуль ЦП осуществляет перезагрузку проекта, снова включается в систему резервирования и автоматически загружает имеющийся у него новый проект в модуль ЦП-партнера с последующей перезагрузкой его по питанию.

ВНИМАНИЕ!

При полной загрузке проекта с включённой функцией автосинхронизации не происходит безударной передачи управления между ЦП в момент загрузки нового проекта на модуль ЦП-партнера. Поэтому перед выполнением полной загрузки проекта на действующем объекте рекомендуется отключить функцию автосинхронизации

Если отключена функция автосинхронизации проектов, то после загрузки обновленного проекта в резервированном контроллере окажутся два модуля ЦП с разными версиями прикладного программного обеспечения. Если изменения в проекте не затронули объем синхронизируемой памяти, то произойдет синхронизация данных процесса, но на объекте «Redundancy» во вкладке «Настройка соединений» информация о загруженных проектах будет подсвечена красным цветом (Рисунок 67).

ВНИМАНИЕ!

Значение функции автосинхронизации проектов (включена/выключена) влияет только на загрузку полностью нового проекта при полной загрузке приложения. Однако при выключенной автосинхронизации проектов, в случае перезапуска ЦП, остается требование к идентичности проектов, поэтому при включении модуля ЦП происходит синхронизация проектов на обоих ЦП со скачиванием проекта с работающего ЦП. В итоге - любой ЦП, запускающийся вторым, скачивает приложение с первого на себя

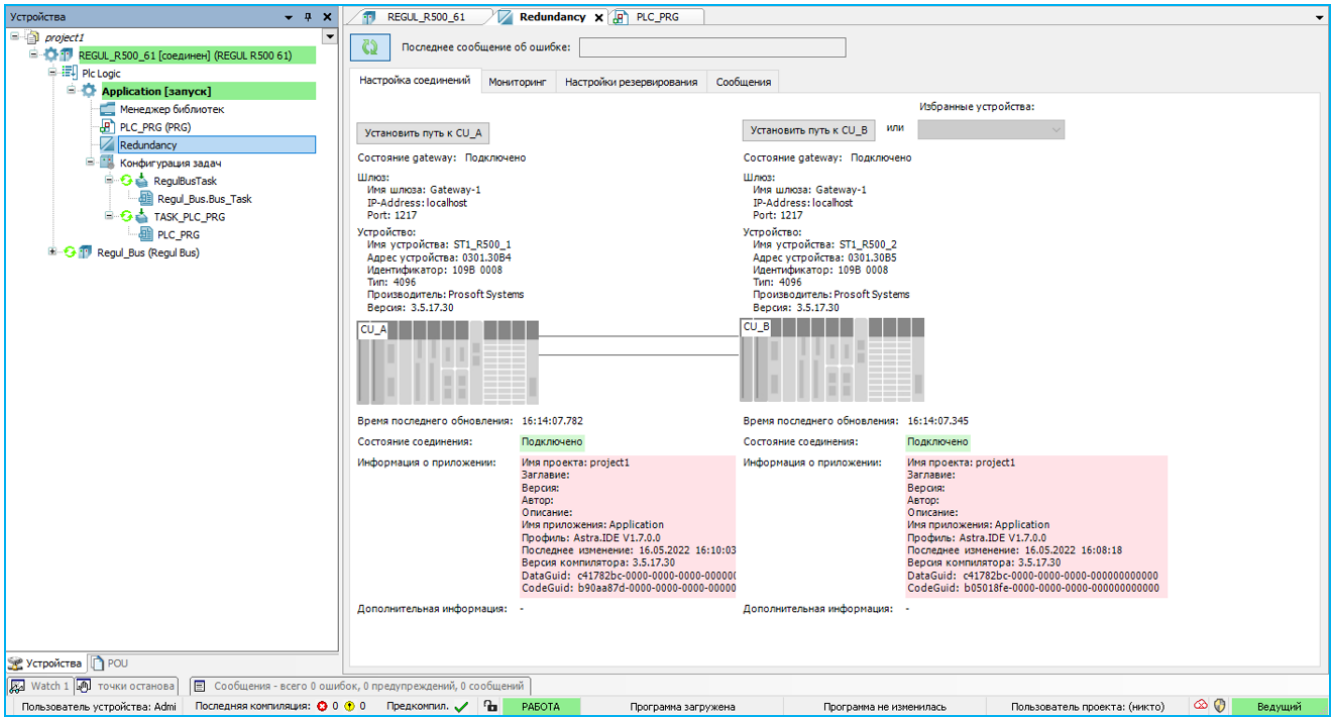

Рисунок 67 – Вкладка «Настройка соединений». Резервный контроллер находится в состоянии Standby, что сигнализирует о синхронизации данных проекта, но загружены разные проекты на модулях ЦП (информация о приложениях помещена на красном фоне)

Далее пользователь может протестировать вновь загруженный проект, передать управление процессом на модуль ЦП с новым проектом. Убедившись, что результаты выполнения новой версии проекта соответствуют ожиданиям, пользователь может загрузить новую версию приложения на модуль ЦП-партнера. Для этого нажмите кнопку ***Загрузить проект с CPU*** в блоке того модуля ЦП, на который требуется загрузить обновленный проект.

Если новая версия приложения по каким-либо причинам не устраивает пользователя, он может загрузить прежнюю версию проекта с модуля ЦП-партнера. Для этого необходимо нажать кнопку ***Загрузить проект с CPU*** у модуля ЦП, на который была осуществлена загрузка новой версии проекта.

По окончании процесса обновите информацию на объекте «Redundancy» (кнопка  в левом верхнем углу). Блок информации о приложении поменяет цвет на зеленый – оба модуля ЦП работают с одинаковыми обновленными проектами.

ВНИМАНИЕ!

Не оставляйте модули ЦП с разными версиями прикладного программного обеспечения, так как в последствии это может привести к невозможности безударного (без сохранения технологических параметров) переключения между основным и резервным модулями ЦП.

Если в процессе изменения проекта каким-либо способом была затронута синхронизируемая память, то в дальнейшем невозможно будет провести безударное обновление прикладного программного обеспечения. В этом случае обновление должно производиться через остановку ПЛК.

Если пользователь хочет заложить на будущее возможность безударного обновления приложения при значительных изменениях проекта, связанных, в том числе, с добавлением новых модулей, данных процесса, которые необходимо будет вносить в объем синхронизируемой памяти, и т.д., то следует заранее заложить необходимое количество резерва в область синхронизируемых данных путем объявления переменных соответствующего типа. В дальнейшем, при изменении проекта, пользователь может привязать новые данные к ранее объявленным переменным.

В ситуации, когда проект значительно модифицирован (внесены изменения в структуру и/или объем синхронизируемых данных), не гарантируется безударная передача управления. Обновление прикладного программного обеспечения в этом случае необходимо осуществлять таким способом, чтобы не произошла попытка синхронизации данных между модулями ЦП с разными версиями прикладного программного обеспечения. Этого можно добиться двумя способами.

Первый алгоритм

С помощью переключателей RUN / STOP на передней панели модуля ЦП или кнопок Старт / Стоп (см. раздел «Переключатели») в среде разработки произвести остановку исполнения приложения на обоих ПЛК (переход в режим «СТОП»). Далее необходимо загрузить новую версию приложения на оба модуля ЦП, после чего произойдет перезагрузка модулей и запуск их в режиме «СТОП». Далее необходимо запустить выполнение приложения одним из выше перечисленных способов.

Второй алгоритм

С помощью переключателей RUN / STOP на передней панели модуля ЦП или кнопок Старт / Стоп в среде разработки произвести остановку исполнения приложения на одном из модулей ЦП. После этого необходимо отключить автосинхронизацию данных с помощью снятия флажка в соответствующем поле во вкладке «Резервирование» объекта «Redundancy» (см. раздел «Настройка и диагностика соединения») или путем физического отключения линий синхронизации между модулями ЦП.

Далее, на остановленный модуль ЦП необходимо загрузить новую версию прикладного программного обеспечения, после чего произойдет перезагрузка приложения и запуск в режиме «СТОП». После чего можно перевести данный модуль ЦП в режим «ЗАПУСК».

Если в составе ПЛК находится хотя бы один модуль ввода / вывода, одновременно опрашиваемый обоими модулями ЦП, то модуль ЦП с обновленной версией прикладного программного запустится с ошибкой синхронизации (так как ранее она была отключена). Если такого модуля нет (ПЛК с полным резервированием), а также не обеспечивается обмен признаками активности ЦП-партнера посредством физической линии связи, то одновременно в работе будут находиться два автономных ПЛК с разными версиями прикладного программного обеспечения.

Далее следует перевести в режим «СТОП» CPU_B и провести аналогичную процедуру на нем. В момент останова приложения на CPU_B произойдет передача управления CPU_A без синхронизации данных (ударное переключение).

После загрузки обновленного прикладного программного обеспечения на CPU_B и включения его в работу можно восстановить автосинхронизацию данных.

	**ВНИМАНИЕ!** В резервированном контроллере, при заводском сбросе одного из ЦП, сбрасывается приложение и на ЦП-партнере (см. в документе «Программное обеспечение Astra.IDE. Руководство пользователя» в разделе «Отладка проекта. Сброс приложений»

Замена модуля ЦП резервированного контроллера

В процессе эксплуатации может возникнуть необходимость в замене неисправного модуля ЦП в резервированном контроллере.

Замену необходимо производить в следующей последовательности:

1. Отсоедините подключенные к разъемам на лицевой панели кабели и извлеките заменяемый модуль ЦП;
2. Возьмите новый модуль ЦП и установите переключатели на лицевой панели в следующие положения:

– «RUN/STOP» в положение «STOP»;
– «KEY» в положение «I»;
– «MBS» в положение, как на заменяемом модуле ЦП;

3. Вставьте новый модуль ЦП в корзину вместо заменяемого ЦП;
4. Подключите ПК напрямую к модулю ЦП и установите соединение (через сканер сети);
5. Если потребуется, произведите процедуру обновления СПО до необходимой версии (подробно описание в документе «Программное обеспечение Astra.IDE. Руководство пользователя» в разделе «Обновление системного программного обеспечения»);
6. Загрузите служебные и конфигурационные файлы, сетевые настройки;

7. Перезагрузите новый модуль ЦП;
8. Восстановите линии синхронизации между ЦП-партерами;
9. Переведите «KEY» в положение «II».
10. Переведите «RUN/STOP» в положение «RUN»;
11. Дождитесь, когда произойдет автосинхронизация проекта с модулем ЦП-партнером.

При наличие полного backup-файла пункты 5 и 6 упростить, произведя процедуру восстановления (подробное описание приведено в документе «Программное обеспечение Astra.IDE. Руководство пользователя» в разделе «Restore/Backup»).

ВНИМАНИЕ!

Манипуляции с новым модулем ЦП, указанные под п.п.4…7, рекомендуем производить на отдельном стенде

ВЗАИМОДЕЙСТВИЕ ГРУППЫ РЕЗЕРВИРУЕМЫХ КОНТРОЛЛЕРОВ С СИСТЕМАМИ ВЕРХНЕГО УРОВНЯ (SCADA)

Для передачи данных и управляющих сигналов между контроллерами серии Regul RX00 и SCADA-системой используются промышленные протоколы, такие как OPC DA, Modbus TCP, IEC 60870-5 104, OPC UA.

В системе резервирования каждый из двух модулей ЦП может обмениваться данными с системой управления независимо от другого модуля ЦП.

При использовании протокола IEC 60870-5 104 на каждом модуле ЦП могут быть настроены один или два канала передачи данных, при этом оба модуля ЦП отдают данные в SCADA-систему вне зависимости от состояния каждого из них (ведущий или ведомый). Система верхнего уровня выбирает актуальный канал обмена данными и управления. Предусмотрена настройка управления устройством **Slave 104** при остановке программы – прекращение всех соединений (*No activity* - по умолчанию) или продолжение работы в обычном режиме (*Normal work*). STOP-режим для IEC-устройств может быть активирован программно (свойство ActivateStopBehavior) – например, при переходе модуля ЦП в состояние *Ведомый*. Одним из сигналов, передаваемых с каждого модуля ЦП, должен быть идентификатор состояния (флаг **IsStateActive** либо значение переменной **redmode** из примера программного кода). Подробное описание настройки протокола IEC 60870-5 104 приведено в документе «Настройка обмена данными по протоколу IEC-104 на контроллерах серии Regul RX00. Руководство пользователя».

При использовании протокола Modbus TCP оба модуля ЦП также независимо передают данные в систему верхнего уровня, но при этом есть возможность изменить поведение ведомого модуля ЦП. Программные устройства **Modbus TCP Master** и **Modbus TCP Slave**, реализующие обмен, имеют специальную настройку **StopModeBehavior** (поведение в режиме Stop) – аппаратная остановка программы модуля ЦП переключателем RUN/STOP. Поведение в данном режиме варьируется от «не изменять» до «закрыть соединение». STOP-режим для Modbus-устройств может быть активирован программно (свойство ActivateStopBehavior) – например, при переходе модуля ЦП в состояние *Ведомый*. Подробное описание настройки протокола Modbus приведено в документе «Настройка обмена данными по протоколу Modbus на контроллерах серии Regul RX00. Руководство пользователя».

Обмен по протоколу OPC DA подразумевает использование дополнительного программного обеспечения (ПО). На сервер сбора данных либо отдельный компьютер, имеющий прямой доступ к резервируемым контроллерам, устанавливается Regul OPC DA Server. Его можно скачать с сайта www.reglab.ru. При настройке требуется внести в конфигурацию OPC-сервера оба модуля ЦП, а затем объединить их в «группу резервирования» (Redundancy

Group). После настройки OPC-сервер будет отображать результирующий набор тегов. Данные будут браться с модуля ЦП, который доступен и находится в данный момент в режиме *Ведущий*. У программного обеспечения Regul OPC DA Server есть дополнительные возможности: различные алгоритмы переключения, ручной выбор модуля ЦП в качестве источника данных и т.п. Кроме того, для системы верхнего уровня предоставляется уже подготовленный канал для обмена данными с группой резервирования, а не с отдельными контроллерами. Подробное описание Regul OPC DA Server приведено в документе «Настройка и работа Regul OPC DA Server. Руководство пользователя».

Обмен с системой верхнего уровня, посредством протокола OPC UA, можно осуществлять, воспользовавшись программным компонентом PsOpcUaServer. Подробное описание PsOpcUaServer приведено в документе «Настройка и работа Regul OPC UA Server. Руководство пользователя».

ОБРАЩЕНИЕ В СЛУЖБУ ТЕХНИЧЕСКОЙ ПОДДЕРЖКИ

Для обращения в техническую поддержку Пользователю необходимо сформировать запрос на сайте технической поддержки: https://support.prosoftsystems.ru, либо отправить письмо по электронной почте: support@prosoftsystems.ru. В первом случае требуется предварительная регистрация.

Обращение обязательно должно содержать следующие сведения:

- подробное описание сложившейся ситуации;
- наименование объекта и его месторасположение;
- наименование системы автоматизации;
- модель ПЛК;
- серийный номер ПЛК;
- версия среды разработки Astra.IDE;
- версия СПО контроллера;
- файл экспорта сетевых настроек контроллера;
- архив с лог-файлами, включающими в себя период времени, когда произошел отказ;
- дата и время возникновения отказа. А также периодичность и устойчивость повторения подобных отказов в случае, если такая информация имеется.

Желательно прислать проект для Astra.IDE, так как это может значительно упростить и ускорить процесс поиска причины отказа.

Лог-файлы, скопированные на компьютер, желательно поместить в архив. Объем заархивированных текстовых файлов сокращается примерно в 10 раз.

Для того, чтобы узнать версию Astra.IDE, в главном меню выберите Справка ⇨ **О программе…** и в открывшемся окне нажмите кнопку ***Информация о версии***.

Для выяснения номера версии СПО контроллера выберите инструмент **Сканер сети**. Сканируйте сеть и найдите нужный контроллер. В области **Информация о ПЛК:** будут отображены имя контроллера в сети, сведения о типе контроллера и версии СПО.

Для сохранения сетевых настроек контроллера в файл:

- выберите инструмент **Сканер сети**. Сканируйте сеть и найдите нужный контроллер;
- нажмите кнопку ***Экспорт***, откроется окно **Save net interfaces settings**. Определите папку, в которой будет храниться этот файл, задайте ему имя, нажмите кнопку ***Сохранить***.

ПРИЛОЖЕНИЕ А

Типы модулей центрального процессора с поддержкой резервирования

Таблица А.1

Тип	Наименование модуля ЦП		
	Модель R200	**Модель R500**	**Модель R600**
I	—	CU 00 051(-W) CU 00 061(-W) CU 00 071(-W)	CU 00 061(-W) CU 00 071(-W)
II	CU 00 041(-W) CU 00 061(-W)	CU 00 021(-W) CU 00 031(-W)	—
III		CU 00 151(-W) CU 00 161(-W) CU 00 171(-W) CU 00 181(-W)	

ПРИЛОЖЕНИЕ Б

Библиотеки

PS_Redundancy

Компонент резервирования ЦП.

Перечисления

RedMode

Перечисление RedMode типа SINT содержит режимы работы резервирования, указанные в таблице Б.1.

Таблица Б.1 – Режимы работы резервирования

Режим	Значение	Описание режима
CONN_ERROR	-3	Ошибка соединения
CR_ERROR	-2	Критическая ошибка
ERROR	-1	Ошибка
DISABLED	0	Выключено
BOOTUP	1	Инициализация
SYNC	2	Синхронизация
STANDBY	3	Ведомый
ACTIVE_STANDALONE	4	Автономный
ACTIVE	5	Активный

Структуры

TAppInfo

Структура с информацией о приложении.

Содержит переменные, указанные в таблице Б.2.

Таблица Б.2 – Переменные TAppInfo

Переменная	Тип	Описание
ProjInfo	PROJECT_INFO	Информация о проекте
sApplicationName	STRING	Имя приложения
sProfile	STRING	Профиль
dtLastChanges	DWORD	Последнее изменение
sCompilerVersion	STRING	Версия компилятора
dataGuid	TGuid	Уникальный идентификатор IEC данных

Переменная	Тип	Описание
codeGuid	TGuid	Уникальный идентификатор IEC кода

➢TGuid

Структура для уникального идентификатора.

Содержит переменные, указанные в таблице Б.3

Таблица Б.3 – Переменные TGuid

Переменная	Тип
data1	DWORD
data2	WORD
data3	WORD
data4	ARRAY [0..7] OF BYTE

➢TStat2

Структура с диагностической информацией.

Содержит переменные, указанные в таблице Б.4.

Таблица Б.4 – Переменные TStat2

Переменная	Тип	Описание
TotalDataSize	UDINT	Общий объем резервируемых данных
xPassiveCpu	BOOL	Роль ведомого ПЛК
xCpuA	BOOL	Признак идентификации ЦП как CPU_A
SyncOkCount	ULINT	Счетчик успешной синхронизации
SyncErrCount	ULINT	Счетчик ошибок синхронизации
SyncOk	BOOL	Задача синхронизирована с ведущим контроллером
TaskStat	TTaskStat	Статистика по задаче
DoSyncTimeUs	UDINT	Время выполнения синхронизации
SendVarTimeUs	UDINT	Время отправки данных
xConnection1	BOOL	Наличие связи по каналу 1
xConnection2	BOOL	Наличие связи по каналу 2

➢TTaskStat

Структура со статистикой по задаче резервирования.

Содержит переменные, указанные в таблице Б.5.

Таблица Б.5 – Переменные TTaskStat

Переменная	Тип	Описание
CallCount	ULINT	Счетчик вызовов задачи
CallIntervalUs	UDINT	Время с момента последнего вызова (мкс)
ConfigIntervalDefaultUs	UDINT	Интервал задачи по умолчанию (мкс)
ConfigIntervalCurrentUs	UDINT	Текущий интервал выполнения задачи (мкс)
OffsetUs	DINT	Текущее смещение задачи (мкс)
JitterUs	DINT	Джиттер задачи (мкс)
StepCorrectionCnt	ULINT	Счетчик жестких коррекций
SoftCorrectionCnt	ULINT	Счетчик мягких коррекций

Функции

➢GetAppInfo

Запрос информации о приложении.

Входной аргумент:

– информация о приложении *appInfo* типа REFERENCE TO TAppInfo.

Возвращаемое значение:

– результат запроса GetAppInfo типа RTS_IEC_RESULT.

Пример:

```
info : PsRedundancy.TAppInfo;
res : RTS_IEC_RESULT;
// ...
res := PsRedundancy.GetAppInfo(info);
```

➢GetConnection

Запрос состояния каналов связи.

Входные аргументы:

– состояние первого канала *xConnection1* типа REFERENCE TO BOOL;

– состояние второго канала *xConnection2* типа REFERENCE TO BOOL.

Возвращаемое значение:

– результат запроса GetConnection типа RTS_IEC_RESULT.

Пример:

```
ch1 : BOOL;
ch2 : BOOL;
res : RTS_IEC_RESULT;
```

```
// ...
res := PsRedundancy.GetConnection(ch1, ch2);
```

➢**GetMode**

Запрос режима работы резервирования.

Входной аргумент:

– отсутствует.

Возвращаемое значение:

– режим работы резервирования GetMode типа RedMode.

Пример:

```
mode : PsRedundancy.RedMode;
// ...
mode := PsRedundancy.GetMode();
```

➢**GetStats2**

Запрос на получение диагностической информации.

Входной аргумент:

– диагностическая информация *stats* типа REFERENCE TO TStat2.

Возвращаемое значение:

– результат запроса GetStats2 типа RTS_IEC_RESULT.

Пример:

```
stats : PsRedundancy.TStat2;
res : RTS_IEC_RESULT;
// ...
res := PsRedundancy.GetStats2(stats);
```

➢**IsCpuA**

Запрос признака CPU_A.

Входной аргумент:

– отсутствует.

Возвращаемое значение:

– TRUE, если данный CPU является CPU_A;

– FALSE, если данный CPU не является CPU_A.

Пример:

```
IF PsRedundancy.IsCpuA() THEN
// Данный ПЛК является CPU A.
END_IF
```

➢SwitchToStandby

Запрос на передачу управления ПЛК-партнеру.

Входной аргумент:

– отсутствует.

Возвращаемое значение:

– результат запроса SwitchToStandby типа RTS_IEC_RESULT.

Пример:

```
IF PsRedundancy.SwitchToStandby() <> CmpErrors.Errors.ERR_OK THEN
// Запрос отклонен
END_IF
```

➢Synchronize

Функция для организации синхронизации задач и данных на двух ПЛК.

Функция должна вызываться перед алгоритмической частью задачи.

Возвращаемое значение:

– TRUE, если синхронизация выполнена в данном цикле;

– FALSE, если синхронизация не выполнена в данном цикле.

Пример:

```
IF PsRedundancy.Synchronize(FALSE) THEN
// Синхронизация выполнена в данном цикле
...
ELSE
// Синхронизация не выполнена в данном цикле
...
END_IF
```

➢Synchronize4

Функция для организации синхронизации задач и данных на двух ПЛК с получением диагностической информации.

Функция должна вызываться перед алгоритмической частью задачи.

Входные аргументы:

– диагностическая информация stats типа REFERENCE TO TStat2.

Возвращаемое значение:

– TRUE, если синхронизация выполнена в данном цикле;

– FALSE, если синхронизация не выполнена в данном цикле.

Пример:

```
stats : PsRedundancy.TStat2;
```

```
// ...
IF PsRedundancy.Synchronize4(FALSE, stats) THEN
// Синхронизация выполнена в данном цикле
...
ELSE
// Синхронизация не выполнена в данном цикле
...
END_IF
```

Функциональные блоки

➢CrossMemory

Функциональный блок для обмена данными между модулями ЦП.

Пример:

```
LocalVariable : BOOL;
RemoteVariable : BOOL;
CrossData : PsRedundancy.CrossMemory(SIZEOF(LocalVariable), ADR(LocalVariable),
ADR(RemoteVariable));
...
PsRedundancy.Synchronize(FALSE);
// После вызова Synchronize данные в RemoteVariable синхронизованы
IF LocalVariable <> RemoteVariable
THEN
    ...
END_IF
```

Метод fb_init

Инициализация функционального блока CrossMemory.

Входной аргумент:

- размер переменной *udiSize* типа UDINT;
- указатель на переменную *pVariable* типа POINTER TO BYTE;
- указатель на переменную - обратная связь *pFeedbackVariable* типа POINTER TO BYTE.

Пример:

```
LocalVariable : DINT;
RemoteVariable : DINT;
CrossData : PsRedundancy.CrossMemory(SIZEOF(LocalVariable), ADR(LocalVariable),
ADR(RemoteVariable));
```

➢SharedMemory

Функциональный блок разделяемой памяти.

При синхронизации данные копируются от ведущего модуля ЦП к ведомому, заменяя предыдущие данные в ведомом ЦП.

Пример:

```
haredVariable : BOOL;
SharedData : PsRedundancy.SharedMemory(SIZEOF(SharedVariable),
ADR(SharedVariable));
...
```

```
PsRedundancy.Synchronize(FALSE);
// После вызова Synchronize данные в SharedVariable синхронизованы
IF SharedVariable
THEN
    ...
END_IF
```

Метод fb_init

Инициализация функционального блока SharedMemory.

Входной аргумент:

- размер переменной *udiSize* типа UDINT;
- указатель на переменную *pVariable* типа POINTER TO BYTE.

Пример:

```
SharedVariable : BOOL;
SharedData : PsRedundancy.SharedMemory(SIZEOF(SharedVariable),
ADR(SharedVariable));
```

PS_Redundancy_OS

Компонент резервирования ЦП. Описание **RedMode, TAppInfo, TGuid** смотри выше.

▸ Структуры

➢TTaskIds

Структура с описанием идентификаторов задач, задействованных в резервировании.

Содержит переменные, указанные в таблице Б.6.

Таблица Б.6 – Переменные TTaskIds

Переменная	Тип	Описание
ids	ARRAY [0..(255 - 1)] OF INT	Массив идентификаторов задач (-1 - invalid id)

➢TRedundancyStat

Структура с диагностической информацией.

Содержит переменные, указанные в таблице Б.7.

Таблица Б.7 – Переменные TRedundancyStat

Переменная	Тип	Описание
TotalDataSize	UDINT	Общий объем резервируемых данных
AsyncTasksCount	UDINT	Количество асинхронных задач
xPassiveCpu	BOOL	Роль ведомого ПЛК

Переменная	Тип	Описание
xCpuA	BOOL	Признак CPU_A. Для CPU A всегда TRUE
xConnection1	BOOL	Наличие связи канал 1
xConnection2	BOOL	Наличие связи канал 2

➢**TSyncTaskStat**

Структура со статистикой по синхронной задаче.

Содержит переменные, указанные в таблице Б.8.

Таблица Б.8 – Переменные TSyncTaskStat

Переменная	Тип	Описание
TaskStat	TTaskStat	Статистика по задаче
SyncStat	TSyncStat	Дополнительная статистика по синхронной задаче

➢**TTaskStat**

Структура со статистикой по задаче.

Содержит переменные, указанные в таблице Б.9.

Таблица Б.9 – Переменные TTaskStat

Переменная	Тип	Описание
TaskName	STRING	Имя задачи
TaskDataSize	UDINT	Объем резервируемых данных в задаче
CallCount	ULINT	Счетчик вызовов задачи
CallIntervalUs	UDINT	Время с момента последнего вызова (мкс)
MaxCallIntervalUs	UDINT	Максимальное значение CallIntervalUs
ConfigIntervalDefaultUs	UDINT	Интервал задачи по умолчанию (мкс)
JitterUs	DINT	Джиттер задачи (мкс)
MaxJitterUs	DINT	Максимальное значение JitterUs
DoSyncTimeUs	UDINT	Время выполнения синхронизации данных
MaxDoSyncTimeUs	UDINT	Максимальное значение DoSyncTimeUs
SendVarTimeUs	UDINT	Время отправки данных
MaxSendVarTimeUs	UDINT	Максимальное значение SendVarTimeUs
IsSyncTask	BOOL	true - синхронная задача, false - асинхронная

➢**TSyncStat**

Структура с дополнительной статистикой по синхронной задаче.

Содержит переменные, указанные в таблице Б.10.

Таблица Б.10 – Переменные TSyncStat

Переменная	Тип	Описание
ConfigIntervalCurrentUs	UDINT	Текущий интервал выполнения задачи (мкс)
OffsetUs	DINT	Текущий смещение задачи (мкс)
StepCorrectionCnt	ULINT	Счетчик жестких коррекций
SoftCorrectionCnt	ULINT	Счетчик мягких коррекций
SyncOkCount	ULINT	Счетчик успешной синхронизации
SyncErrCount	ULINT	Счетчик ошибок синхронизации
SyncOk	BOOL	Задача синхронизирована с ведущим контроллером

▸ Функции

➢GetRedundancyStat

Запрос на получение общей диагностической информации. Перед вызовом обновите статистику, выполнив UpdateStats().

Входной аргумент:

– диагностическая информация по синхронной задаче *stats* типа REFERENCE TO TRedundancyStat.

Возвращаемое значение:

– результат запроса *GetRedundancyStat* типа RTS_IEC_RESULT.

Пример:

```
stats : PsRedundancy_OS.TRedundancyStat;
res : RTS_IEC_RESULT;
// ...
res := PsRedundancy_OS.GetRedundancyStat(stats);
```

➢GetSyncTaskStat

Запрос на получение диагностической информации по синхронной задаче. Перед вызовом обновите статистику, выполнив UpdateStats().

Входной аргумент:

– диагностическая информация по синхронной задаче *stats* типа REFERENCE TO TSyncTaskStat.

Возвращаемое значение:

– результат запроса *GetSyncTaskStat* типа RTS_IEC_RESULT.

Пример:

```
stats : PsRedundancy_OS.TSyncTaskStat;
res : RTS_IEC_RESULT;
// ...
res := PsRedundancy_OS.GetSyncTaskStat(stats);
```

➢**GetTaskStat**

Запрос на получение диагностической информации по задаче. Перед вызовом обновите статистику, выполнив UpdateStats().

Входные аргументы:

- идентификатор задачи *task_id* типа BYTE;
- диагностическая информация по задаче *stats* типа REFERENCE TO TTaskStat.

Возвращаемое значение:

- результат запроса *GetTaskStat* типа RTS_IEC_RESULT.

Пример:

```
stats : PsRedundancy_OS.TTaskStat;
task_id: BYTE := 0;
res : RTS_IEC_RESULT;
// ...
res := PsRedundancy_OS.GetTaskStat(task_id, stats);
```

➢**UpdateStats**

Запрос на обновление статистики по всем задачам.

Для получения актуальной информации, при использовании функций GetRedundancyStat(), GetSyncTaskStat(), GetTaskStat(), GetConnection() нееобходимо, перед их вызовом, обновить статистику, выполнив UpdateStats().

Входной аргумент:

- отсутсвует.

Возвращаемое значение:

- результат запроса *UpdateStats* типа RTS_IEC_RESULT.

Пример:

```
res : RTS_IEC_RESULT;
// ...
res := PsRedundancy_OS.UpdateStats();
```

➢**GetAppInfo**

Запрос информации о приложении.

Входной аргумент:

- информация о приложении *appInfo* типа REFERENCE TO TAppInfo.

Возвращаемое значение:

- результат запроса *GetAppInfo* типа RTS_IEC_RESULT.

Пример:

```
info : PsRedundancy_OS.TAppInfo;
res : RTS_IEC_RESULT;
// ...
res := PsRedundancy_OS.GetAppInfo(info);
```

➢GetConnection

Запрос состояния каналов связи. Перед вызовом, обновите статистику, выполнив UpdateStats().

Входные аргументы:

- состояние первого канала *xConnection1* типа REFERENCE TO BOOL;
- состояние второго канала *xConnection2* типа REFERENCE TO BOOL.

Возвращаемое значение:

- результат запроса *GetConnection* типа RTS_IEC_RESULT.

Пример:

```
ch1 : BOOL;
ch2 : BOOL;
res : RTS_IEC_RESULT;
// ...
res := PsRedundancy_OS.GetConnection(ch1, ch2);
```

➢GetMode

Запрос режима работы резервирования.

Входной аргумент:

- отсутсвует.

Возвращаемое значение:

- режим работы резервирования *GetMode* типа RedMode.

Пример:

```
mode : PsRedundancy_OS.RedMode;
// ...
mode := PsRedundancy_OS.GetMode();
```

➢GetTaskList

Запрос идентификаторов задач, задействованных в резервировании.

Входной аргумент:

- идентификаторы задач *task_ids* типа REFERENCE TO TTaskIds.

Возвращаемое значение:

- результат запроса *GetTaskList* типа RTS_IEC_RESULT.

Пример:

```
task_ids : PsRedundancy_OS.TTaskIds;
res : RTS_IEC_RESULT;
// ...
res := PsRedundancy_OS.GetTaskList(task_ids);
```

➢**IsCpuA**

Запрос признака CPU_A.

Входной аргумент:

- отсутсвует.

Возвращаемое значение:

- TRUE, если данный CPU является CPU_A;
- FALSE, если данный CPU не является CPU_A.

Пример:

```
IF PsRedundancy_OS.IsCpuA() THEN
    // Данный ПЛК является CPU A.
END_IF
```

➢**SwitchToStandby**

Запрос на передачу управления ПЛК-партнеру.

Входной аргумент:

- отсутсвует.

Возвращаемое значение:

- результат запроса *SwitchToStandby* типа RTS_IEC_RESULT.

Пример:

```
IF PsRedundancy_OS.SwitchToStandby() <> CmpErrors.Errors.ERR_OK THEN
    // Запрос отклонен
END_IF
```

➢**setDenyBeActive (устаревшая)**

Запрос на установку/снятия флага на запрет быть ведущим.

Входной аргумент:

- флаг запрета быть ведущим *DenyBeActive* типа BOOL.

Возвращаемое значение:

– результат запроса *setDenyBeActive* типа RTS_IEC_RESULT.

Пример:

```
xDenyBeActive : BOOL := TRUE;
res : RTS_IEC_RESULT;
// ...
res := PsRedundancy_OS.setDenyBeActive(xDenyBeActive);
```