Методы реализации резервирования процессорных модулей для Багет-ПЛК1

Я. А. Зотов[1], Д. В. Яриков[2]

[1]НИЦ «Курчатовский институт» – НИИСИ, Москва, Россия, zotov@niisi.ras.ru

[2]НИЦ «Курчатовский институт» – НИИСИ, Москва, Россия, yarikov@niisi.ras.ru

Аннотация. В статье рассматриваются методы реализации резервирования процессорных модулей для отечественного программируемого логического контроллера Багет-ПЛК1. Предложен алгоритм резервирования с синхронизацией данных между основным и резервным процессорными модулями по сети Ethernet, обеспечивающий бесшовное переключение в случае отказа основного модуля. Описаны особенности программной реализации на основе трансляции прикладных программ с языков МЭК 61131-3 в код на языке Си, а также механизм автоматического определения ролей модулей и восстановления работоспособности системы. Решение предназначено для применения в системах управления объектами критической инфраструктуры.

Ключевые слова: резервирование, процессорные модули, Багет-ПЛК1, синхронизация данных, бесшовное переключение

1. Введение

В НИЦ «Курчатовский институт» НИИСИ разработаны программируемые логические контроллеры семейства Багет-ПЛК1 (далее ПЛК) на основе отечественного микропроцессора 1890ВМ108 [1]. ПЛК предназначены для сбора, преобразования, обработки, хранения информации и выработки команд управления в режиме реального времени в автоматизированных системах управления (АСУ) объектов критической инфраструктуры.

ПЛК содержит один или несколько процессорных модулей и модулей ввода-вывода, обеспечивающие взаимодействие ПЛК с периферийными устройствами по дискретным и аналоговым сигналам, а также по последовательным интерфейсам. Процессорные модули и модули ввода-вывода взаимодействуют между собой посредством аппаратной шины RS-485 с программным интерфейсом Modbus RTU.

ПЛК функционирует под управлением отечественной операционной системы реального времени ОС РВ Багет 2.7 (далее ОСРВ). ОСРВ предназначена для создания программного обеспечения вычислительных систем, комплексов и средств автоматизированного управления, работающих в режиме реального масштаба времени [2].

Описанная в данной работе реализация резервирования процессорных модулей предназначена для выполнения в прикладной программе, созданной с помощью инструментальной программы «Среда разработки и отладки» (СРиО). Программа СРиО предназначена для создания и отладки прикладных программ на языках, специализированных в стандарте МЭК 61131-3-2016 (далее МЭК) для ПЛК [3]. Помимо других функций программа СРиО осуществляет трансляцию программ с языков МЭК в программу на языке Си для ОСРВ, который после компилируется в бинарный исполняемый файл для ПЛК.

2. Резервирование

Резервирование — это метод повышения характеристик надёжности технических устройств или поддержания их на требуемом уровне посредством использования аппаратной избыточности за счет включения запасных (резервных) элементов и связей, дополнительных по сравнению с минимально необходимым для выполнения заданных функций в данных условиях функционирования [4].

Элементы устройства, обеспечивающие его работоспособность, называются основными элементами; резервными элементами называются элементы, предназначенные для обеспечения работоспособности устройства в случае отказа основных элементов.

Резервирование процессорных модулей ПЛК может осуществляться двумя методами: активным и пассивным. При активном резервировании два или более процессорных модуля работают одновременно, обеспечивая постоянный контроль процесса управления. Один модуль является основным, а остальные – резервными, активирующимися в случае отказа

основного. Пассивное резервирование предполагает устройство системы таким образом, что один процессорный модуль выполняет прикладную программу, а резервные находятся в состоянии ожидания. При возникновении неисправности основного процессорного модуля, переключение на резервный процессорный модуль осуществляется вручную или автоматически.

Процедура резервирования включает определение неисправности элемента оборудования, автоматическую замену неисправного оборудования резервным элементом, автоматическую передачу резервному элементу оборудования функций основного элемента [5].

Процедура резервирования должна занимать настолько малое количество времени, чтобы замена основного элемента на резервный не вызывала изменений в функционировании объекта критической инфраструктуры. Такое резервирование называется «бесшовным».

Резервирование процессорных модулей включает в себя выполнение в ПЛК процедуры анализа корректности функционирования процессорного модуля.

Если происходит сбой процессорного модуля, который не допускает дальнейшей эксплуатации, то управление передаётся резервному модулю, который должен выполнять в полном объёме функции основного модуля. Передача управления от основного процессорного модуля к резервному модулю должна происходить так, чтобы алгоритм управления объектом критической инфраструктуры не нарушался. В частности, требуется, чтобы на момент переключения данные, необходимые для функционирования управляющей программы на резервном модуле, совпадали с такими данными на основном модуле. На основном и на резервном процессорном модуле должна выполняться одна и та же прикладная программа

3. Аппаратная конфигурация ПЛК с резервированием

Для обеспечения резервирования в рамках рассматриваемой реализации процессорные модули должны быть соединены напрямую друг с другом посредством локальной сети Ethernet.

Основной и резервный процессорные модули подключены к одному набору модулей ввода-вывода. Для связи с модулями ввода-вывода на аппаратном уровне используется шина RS-485, а на программном уровне протокол Modbus RTU. Этот протокол предполагает только одно активное устройство [6], поэтому резервный процессорный модуль не выполняет опросы модулей ввода-вывода.

Помимо модулей ввода-вывода ПЛК может общаться с периферийными устройствами по сети Ethernet посредством промышленных протоколов связи через первый порт Ethernet.

Аппаратная конфигурация ПЛК с резервированием представлена на Рис. 1.

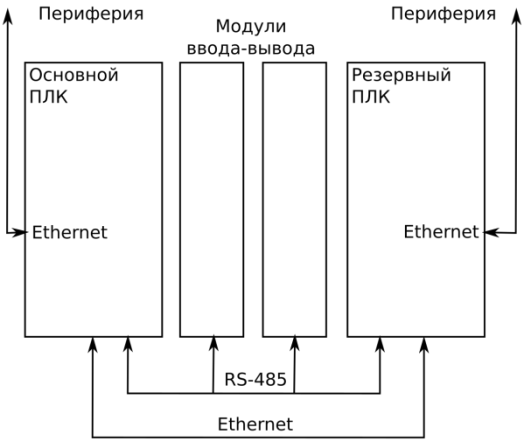

Рис. 1. Аппаратная конфигурация ПЛК

4. Алгоритм функционирования ПЛК с резервированием

Программа, написанная на языке МЭК, представляет собой набор задач, объединённых в один ресурс (или в несколько ресурсов). Задачи циклически выполняются, каждая со своим периодом. Задача имеет входные переменные, выходные переменные и промежуточные переменные, которые можно использовать и как входные и как выходные. Кроме того, ресурс может иметь глобальные переменные, доступные образующим ресурс задачам. Значения глобальных переменных сохраняются между циклами выполнения этих задач [7].

Результат выполнения каждой задачи на очередном цикле полностью определяется набором значений входных переменных задачи и набором значений глобальных переменных ресурса.

Поэтому для дублирования вычислительного процесса на резервном (запасном) процессорном модуле достаточно обеспечить совпадение значений входных и глобальных переменных на основном и на резервном модулях.

Для реализации алгоритма резервирования на резервный модуль передаются актуальные значения всех переменных из исходных текстов, написанных на языках МЭК. Сразу после переключения управления с текущего основного процессорного модуля на резервный, на резервном модуле начинает выполняться

управляющая программа с актуальными текущими значениями всех переменных.

В силу изложенного результат выполнения управляющей программы будет такой же, как и на основном модуле (при условии его штатной работы).

Таким образом, на текущем основном модуле программа выполняет основной цикл управления (опрос данных, выполнение прикладного алгоритма, вывод данных) и последующую передачу значений входных и глобальных переменных на резервный модуль.

Программа, выполняемая на резервном модуле в цикле, определяет работоспособность основного модуля и в случае обнаружения некорректной работы основного модуля, запускает прикладную программу, которая берет на себя функции управления.

Ещё одна функция резервного модуля состоит в том, чтобы перевести бывший основной модуль в состояние резервного, для того чтобы этот модуль гарантированно перестал направлять управляющие воздействия модулям ввода-вывода.

Алгоритм функционирования ПЛК с резервированием основан на постоянной синхронизации данных между основным и резервным процессорными модулями и контроле работоспособности основного процессорного модуля.

Для передачи данных между основным и резервным процессорными модулями используется протокол связи, использующий спецификацию TCP. Установка соединения между процессорными модулями осуществляется по их индивидуальным IP-адресам. На основном и на резервном процессорном модуле выполняется одна и та же прикладная программа, количество и типы переменных прикладной программы определены статично и не меняются во время выполнения программы. Это позволяет заранее определить набор прикладных переменных и последовательно передавать их в определенном порядке. При этом целостность и порядок следования данных обеспечивается на уровне протокола TCP [8].

Алгоритм функционирования прикладной программы ПЛК состоит из трех циклически повторяемых этапов:

- опрос значений модулей ввода
- выполнение алгоритма прикладной программы
- отправка значений в модули вывода.

Для резервирования после трех основных этапов добавляется четвертый этап: синхронизация текущего состояния переменных между основным и резервный процессорными модулями. При этом на основном процессорном модуле происходит отправка значений переменных на резервный процессорный модуль, а при выполнении на резервном процессорном модуле получение значений переменных от основного процессорного модуля.

В ходе выполнения прикладной программы, основной процессорный модуль выполняет все четыре этапа алгоритма, а резервный процессорный модуль выполняет только четвертый этап и находится в состоянии ожидания, для того чтобы приступить к выполнению трех основных этапов при смене ролей.

При старте процессорных модулей происходит автоматическое определение основного и резервного процессорного модуля. В связи с тем, что время инициализации программы ПЛК не является константным (в масштабе микросекунд), это значение используется для инициализации генератора псевдослучайных чисел.

В прикладной программе указаны IP адреса портов Ethernet основного и резервного процессорного модуля. При старте процессорного модуля определение роли (основной/резервный) выполняется следующим образом:

- Процессорный модуль назначает себе случайный IP адрес из заданной в конфигурации подсети (адресом является адрес основного процессорного модуля, у которого младший октет заменен случайным числом, генерация случайного числа происходит на основе времени загрузки)
- происходит поиск остальных устройств в локальной сети второго порта Ethernet
- если обнаружено устройство с адресом основного процессорного модуля, текущий процессорный модуль устанавливает себе сетевые адреса резервного процессорного модуля (на первый и второй порт Ethernet)
- если обнаружено устройство с адресом резервного процессорного модуля, текущий процессорный модуль устанавливает себе сетевые адреса основного устройства
- если устройств с адресами из конфигурации не обнаружено, то процессорный модуль, у которого последний октет адреса второго порта Ethernet больше, становится основным и устанавливает себе сетевые адреса основного процессорного модуля
- если устройств с адресами из конфигурации не обнаружено, то процессорный модуль, у которого последний октет адреса второго порта Ethernet меньше, становится резервным и устанавливает себе сетевые адреса резервного

процессорного модуля

- Процессорный модуль устанавливает подключение по адресу другого процессорного модуля (основного/резервного), запрашивает его роль (основной/резервный) и принимает себе роль отличную от второго процессорного модуля (резервный/основной).

При таком алгоритме резервирования сетевые адреса процессорного модуля не связаны однозначно с его ролью, то есть при смене роли не должна происходить смена сетевых адресов.

Смена роли происходит при обрыве соединения или отсутствии данных на резервном процессорном модуле более трех циклов программы, при этом резервный процессорный модуль сначала пытается повторно установить соединение с основным и в случае неудачи меняет свои роль на основную.

При выходе из строя одного из процессорных модулей аппаратура позволяет произвести его горячую замену. Условием возможности замены является то, что прикладная программа на вновь устанавливаемом процессорном модуле, а также на основном и резервном процессорных модулях одна и та же.

При этом на вновь устанавливаемом процессорном модуле будет автоматически выбрана роль резервного процессорного модуля, а сетевые адреса будут отличаться от активного процессорного модуля.

5. Особенности программной реализации

Специфика программной реализации состоит в том, что управляющая программа, написанная на одной из языков МЭК, подвергается промежуточной трансляции, которая переводит исходную программу на язык программирования Си.

При трансляции программы, написанной на языках МЭК в Си код, происходит преобразование объектов языков МЭК в конструкции языка Си. При этом простые типы преобразуются в простые типы языка Си. В то же время программы и функциональные блоки, написанные на языках МЭК, преобразуются в функции и структуры Си, содержащие все переменные программы или функционального блока [9].

Транслятор не осуществляет неявное создание переменных, сохраняющих свои значения при выполнении прикладной программы, помимо переменных, явно определенных в программе на языках МЭК.

Для осуществления бесшовного резервирования при смене роли резервного процессорного модуля на основную, необходимо синхронизировать значения всех переменных, сохраняющих свое значение между итерациями цикла функционирования прикладной программы ПЛК. Таким образом, необходимо синхронизировать значения всех глобальных в терминологии МЭК переменных, переменных программ и функциональных блоков.

Для синхронизации стандартных типов достаточно скопировать содержимое участков памяти, в которых хранятся значения переменных прикладной программы. Несколько сложнее обстоит дело с пользовательскими программами и функциональными блоками.

Согласно стандарту МЭК 61131-3-2016 переменные пользовательского функционального блока могут иметь тип "внешний", а переменные программы могут как иметь тип внешний, так и соответствовать физическим входам или выходам. В этих случаях такие переменные в полученном Си коде преобразуются в указатели на глобальные переменные или переменные, содержащие значение физического входа или выхода, соответственно. Аналогично преобразуются глобальные переменные, соответствующие физическим входам или выходам. Значения таких переменных-указателей передавать на резервный блок не требуется, и более того во время выполнения программы на разных процессорных модулях данные адреса не совпадают, поэтому копирование значений указателей может привести к ошибке сегментирования в программе при активации резервного процессорного модуля.

Для решения задачи синхронизации переменных пользовательских программ и функциональных блоков при трансляции программы из языков МЭК в Си код дополнительно добавлено формирование вспомогательных элементов:

- массив структур содержащих указатель на переменную, требующую синхронизацию, и размер переменной в памяти;
- массив указателей на переменные-указатели.

Данные массивы позволяют осуществить передачу в резервный процессорный модуль содержимого участков памяти, в которых хранятся значения переменных. Для передачи достаточно в цикле обойти все указатели и отправить фрагмент памяти указанного размера для каждой переменной. При приёме необходимо выполнить следующее:

- скопировать значения локальных указателей переменных во временный массив;

- принять весь набор переменных, полученных от основного модуля, во временный буфер;
- после успешного приема резервным модулем всех переменных, скопировать полученные значения из временного буфера в участки памяти переменных прикладной программы;
- восстановить значения ранее скопированных локальных указателей из временного массива.

Количество и типы переменных не меняются во время выполнения программы, поэтому размер буфера также не изменяется, и этот размер возможно вычислить при старте программы. В случае если не удается получить буфер целиком, заполнение значений переменных не выполняется.

При смене роли резервный процессорный модуль начинает выполнение алгоритма прикладной программы, при этом выполнение программы продолжается из состояния, отличающегося не более чем на один цикл прикладной программы. Под состоянием в данном случае предполагается набор значений всех переменных программы. В связи с тем, что синхронизация значений переменных осуществляется после выполнения основных этапов цикла прикладной программы, не представляется возможным гарантировать синхронизацию значений всех необходимых переменных между двумя процессорными модулями с разницей менее чем в один цикл.

Передача значений между процессорными модулями не является атомарной операцией. В случае отказа процессорного модуля во время выполнения отправки значений переменных, резервный модуль не получит полный набор переменных, размер полученного буфера будет меньше ожидаемого. Поскольку размер буфера фиксирован и не меняется во время выполнения программы, получение меньшего количества данных будет критерием выхода из строя основного процессорного модуля. В таком случае резервному процессорному модулю придется продолжать выполнение программы с набором переменных предыдущего цикла прикладной программы.

6. Заключение

В ходе работы над реализацией процедуры резервирования процессорных модулей для Багет-ПЛК1 был создан алгоритм и разработана соответствующая программная реализация определения состояния основного процессорного модуля и автоматического изменения роли резервного модуля на основную.

Основной процессорный модуль отправляет данные на резервный процессорный модуль каждую итерацию цикла прикладной программы, благодаря чему достигается идентичность состояния процессорных модулей обеспечивается бесшовное резервирование.

Данная реализация процедуры резервирования предназначена для прикладных программ, написанных на языках МЭК, разработанных в среде разработки СРиО.

Публикация выполнена в рамках государственного задания НИЦ «Курчатовский институт» - НИИСИ по теме FNEF-2024-0001.

An approach to implementing a redundancy algorithm

Y. A. Zotov, D. V. Yarikov

Abstract. The article discusses methods for implementing redundancy of processor modules for the programmable logic controller Baget-PLC1. A redundancy algorithm with data synchronization between the main and backup modules via Ethernet is proposed, ensuring seamless switching in case of main module failure. The specifics of software implementation based on the translation of application programs from IEC 61131-3 languages into C code are described, as well as the mechanism for automatic role determination of modules and system recovery. The solution is intended for use in control systems of critical infrastructure facilities.

Keywords: redundancy, processor modules, Baget-PLC1, data synchronization, seamless switching

Литература

1. Сердин, О. В. Многоцелевой программируемый логический контроллер «Багет-ПЛК1»: патент на полезную модель № 211983 Рос. Федерация: G06F 9/00 / О. В. Сердин, М. А. Голяков, А. В. Бакалдин, С. Е. Серяков, М. А. Чушев; патентообладатель Федеральное государственное учреждение «Федеральный научный центр Научно-исследовательский институт системных исследований Российской академии наук». — № 2021129783 ; заявл. 12.10.2021 ; опубл. 30.06.2022.

2. Годунов А.Н., Солдатов В.А. Операционные системы семейства Багет (сходство, отличия и перспективы) // Программирование, 2014, № 5, с. 68-76

3. ГОСТ ГОСТ Р МЭК 61131-3-2016 «Контроллеры программируемые. Часть 3. Языки программирования»

4. Черкесов. Г.Н. Надежность аппаратно-программных комплексов/ Учебное пособие. – СПб.: Питер, 2005. –479 с.

5. Энциклопедия АСУ ТП. 8. Аппаратное резервирование. 8.1. Основные понятия и определения. URL: https://www.reallab.ru/bookasutp/8-apparatnoe-rezervirovanie/8-1-osnovnie-ponyatiya-i-opredeleniya // Энциклопедия АСУ ТП (дата обращения: 08.12.2025).

6. Просто о Modbus RTU с подробным описанием и примерами. URL: https://ipc2u.ru/articles/prostye-resheniya/modbus-rtu // IPC2U — Промышленные компьютеры, ПЛК, системы связи (дата обращения: 10.12.2025).

7. Энциклопедия АСУ ТП. 9.3. Системы программирования на языках МЭК 61131-3. URL: https://www.reallab.ru/bookasutp/9-programmnoe-obespechenie/9-3-sistemi-programmirovaniya-mek-61131-3/ // Энциклопедия АСУ ТП (дата обращения: 09.12.2025).

8. Рубашенков Антон Михайлович, Бобров Андрей Виорелович Протокол tcp // Наука, техника и образование. 2018. №11 (52). URL: https://cyberleninka.ru/article/n/protokol-tcp (дата обращения: 16.12.2025).

9. Documentation | beremiz.org. URL: https://beremiz.org/doc // beremiz.org (дата обращения: 12.12.2025).