GE Fanuc Automation

Программируемые управляющие устройства

ПЛК VersaMax®

Руководство пользователя

GFK-1503C-RU *Март 2001*

Как используются предупреждения, предостережения и примечания в этой публикации.

Предупреждение

Предупреждения используются в этой публикации чтобы подчеркнуть, что в этом оборудовании используются или могут быть связаны с ним опасные напряжения, токи, температуры или другие условия, которые могут причинить ущерб персоналу.

Предупреждение используется в ситуациях, когда невнимание может вызвать несчастный случай или повреждение оборудования.

Предостережение

Предостережение используется в случаях, когда оборудование может быть повреждено по неосторожности.

Примечание

Примечание привлекает внимание к информации, которая особенно важна для понимания и для работы с оборудованием.

Этот документ основан на информации, доступной на время его публикации. Несмотря на принятые меры по повышению точности документа, информация, приведенная в нем, не содержит всех деталей или изменений в программном или аппаратном обеспечении, и не включает в себя все возможные случаи, связанные с установкой, работой или обслуживанием. Возможности, описанные здесь, могут иметься не во всех аппаратных и программных системах. GE Fanuc Automation не берет на себя обязательств по сообщению пользователям этого документа об изменениях, сделанных впоследствии.

GE Fanuc Automation не заявляет и не гарантирует ничего из высказанного, подразумевающегося или установленного и не несет никакой ответственности за точность, полноту или полезность информации, приведенной в этом документе. Коммерческое применение или соответствие цели не гарантируется.

Содержание

Содержание

Содержание

Содержание

Глава 1 Введение

Руководство к набору документов VersaMax®

Это руководство содержит основную информацию по работе ЦПУ и содержанию программы. Оно также содержит подробное описание особенностей программирования.

Глава 1 содержит общее описание продукции семейства VersaMax.

Модули ЦПУ подробно описаны в главах 2 и 3.

Установка описана в главе 4

Конфигурирование ПЛК описано в главе 5. Конфигурация определяет некоторые параметры работы модуля, а также устанавливает адреса, используемые каждым модулем системы.

Конфигурирование порта Ethernet для ЦПУ модели IC200CPUE05 описано в главе 6.

Работа ЦПУ описана в главе 7.

Связь по последовательному порту описана в главе 12.

Связь по порту Ethernet для ЦПУ модели IC200CPUE05 описана в главе 13.

Остальная часть руководства описывает программирование:

- Организация прикладной программы: глава 8
- Структура данных: глава 9
- Система команд: глава 10
- Функция системного запроса Service Request: глава 11
- Функция ПИД-регулирования: глава 14
- Временные параметры команд: приложение A

Другие руководства по VersaMax

Модули, источники питания и шасси VersaMax. Руководство пользователя (VersaMax Modules, Power Supplies, and Carriers User's Manual) (номер по каталогу GFK-1504)	Описывает большое количество модулей ввода-вывода и вспомогательных модулей VersaMax, источников питания и шасси. Это руководство содержит также детальные инструкции по монтажу системы.
Монитор Ethernet станции ПЛК VersaMax. Руководство пользователя (VersaMax PLC Ethernet Station Manager's Manual) (номер по каталогу GFK-1876)	Описывает интерфейс функций диагностики Ethernet модуля ЦПУ IC200CPUE05.
Модуль сетевого интерфейса Ethernet VersaMax. Руководство пользователя (VersaMax Ethernet Network Interface Unit User's Manual) (номер по каталогу GFK-1860)	Описывает установку и работу модуля сетевого интерфейса Ethernet.
Модуль сетевого интерфейса Genius VersaMax. Руководство пользователя (VersaMax Genius NIU User's Manual) (номер по каталогу GFK-1535)	Описывает установку и работу модуля сетевого интерфейса Genius.
Коммуникационные модули DeviceNet VersaMax Руководство пользователя (VersaMax DeviceNet Communications Modules User's Manual) (номер по каталогу GFK-1533)	Описывает установку и работу модуля сетевого интерфейса DeviceNet и сетевого модуля DeviceNet Slave.
Коммуникационные модули Profibus VersaMax Руководство пользователя (VersaMax Profibus Communications Modules User's Manual) (номер по каталогу GFK-1534)	Описывает установку и работу модуля сетевого интерфейса Profibus и сетевого коммуникационного модуля Profibus.

Продукция семейства VersaMax®

Продукция семейства VersaMax обеспечивает построение универсальной распределенной системы ввода-вывода для систем управления на основе PC и ПЛК. Разработанная для промышленной автоматизации, система ввода-вывода VersaMax обеспечивает общую гибкую структуру ввода-вывода для задач местного и удаленного управления. ПЛК VersaMax сочетает мощность большого ПЛК и полный диапазон модулей ввода-вывода и вспомогательных модулей. Станции ввода-вывода VersaMax с модулями сетевого интерфейса позволяют использовать гибкую систему ввода-вывода VersaMax в сетях различных типов. VersaMax соответствует требованиям UL, CUL, CE, Demko, Class 1 Zone 2 и Class I Division 2.

Будучи масштабируемым решением, система ввода-вывода VersaMax сочетает компактность и модульность для облегчения использования. Глубина 70 мм и небольшие посадочные размеры обеспечивают легкий и удобный монтаж, а также экономию места. Модули могут содержать до 32 каналов ввода-вывода каждый.

Компактные модульные изделия VersaMax устанавливаются на DIN-рейку – до 8 модулей ввода-вывода и вспомогательных модулей на крейт, и до 8 крейтов на ПЛК VersaMax или станцию ввода-вывода VersaMax. Крейты расширения могут быть расположены на расстоянии до 750 м от главного ПЛК VersaMax или главного крейта станции ввода-вывода VersaMax. Крейты расширения могут включать любые модули ввода-вывода, вспомогательные и коммуникационные модули VersaMax.

VersaMax обеспечивает автоматическую адресацию, что позволяет исключить традиционное конфигурирование и потребность в дополнительных инструментальных средствах. Различные способы внешних подключений обеспечивают поддержку двух-, трех- и четырех-проводных схем подключения.

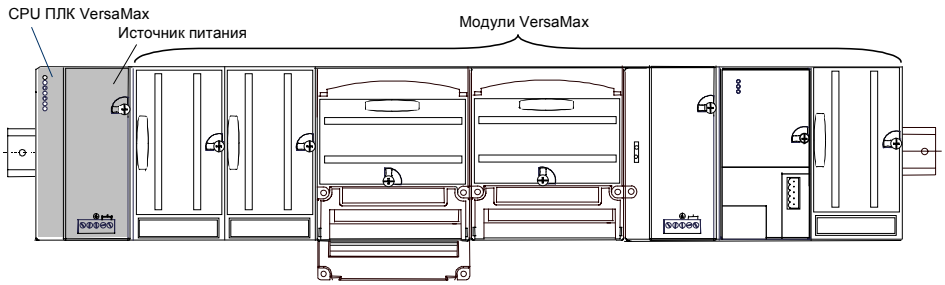
Функция горячей замены позволяет добавлять и заменять модули при работающем оборудовании или выполняющемся процессе без нарушения монтажа внешних проводов, что обеспечивает максимально быстрый ремонт оборудования и уменьшение времени, требующегося на ремонт.

Ввод-вывод VersaMax может быть подключен к сетевой шине. Имеются интерфейсы полевой шины для Genius, DeviceNet, Profibus и Ethernet.

1

Модули ЦПУ для ПЛК VersaMax.

ПЛК VersaMax состоит из группы модулей VersaMax с ЦПУ VersaMax и присоединенным к нему блоком питания в первой позиции.



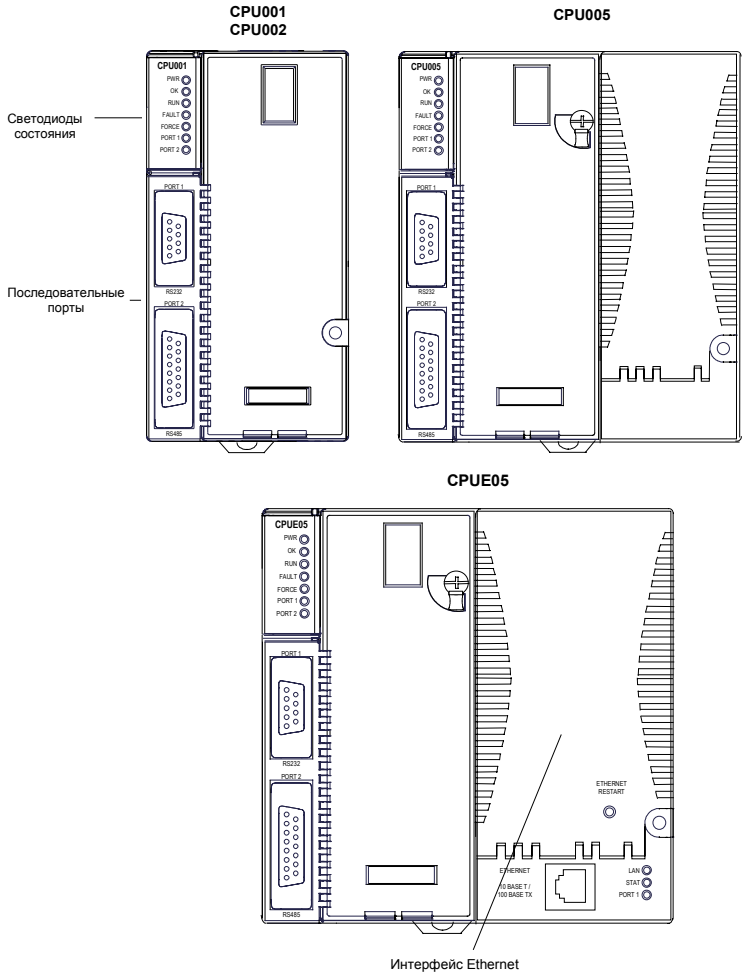
Все ЦПУ VersaMax обеспечивают эффективное функционирование ПЛК. Они разработаны для построения систем управления, включающих в себя до 64 модулей, с количеством каналов ввода-вывода до 2048. Два последовательных порта обеспечивают интерфейсы RS-232 и RS-485 для связи по протоколам SNP слэйв и RTU слэйв. Модель ЦПУ IC200CPUE05 имеет встроенный порт Ethernet.

Основные возможности ЦПУ.

- Программирование на языке релейно-контактной логики, языке функциональных схем и на языке текстовых команд.
- Обработка данных с плавающей точкой.
- Энергонезависимая флэш-память для хранения программ.
- Батарейная поддержка программы, даты и времени суток.
- Переключатель режима Run/Stop.
- Встроенные порты RS-232 и RS-485.
- Совместимость с устройством хранения программ EZ.

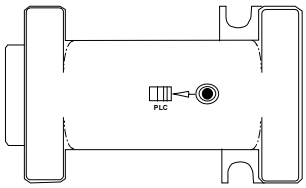
Существующие ЦПУ VersaMax

ЦПУ с двумя последовательными портами, 34кБ конфигурируемой памяти	IC200CPU001
ЦПУ с двумя последовательными портами, 42кБ конфигурируемой памяти	IC200CPU002
ЦПУ с двумя последовательными портами, 64кБ конфигурируемой памяти	IC200CPU005
ЦПУ с двумя последовательными портами и встроенным интерфейсом Ethernet, 64кБ конфигурируемой памяти	IC200CPUE05



Устройство хранения программ *EZ.*

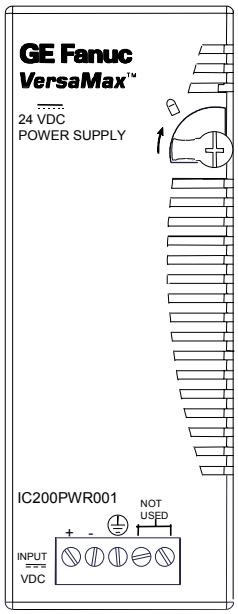
Устройство хранения программ EZ (IC200ACC003) может быть использовано для хранения и обновления конфигурации, прикладной программы и данных ПЛК VersaMax. Для первоначальной записи данных в устройство используются программатор и ЦПУ ПЛК.



Блоки питания

Блоки питания переменного и постоянного тока обеспечивают модулям в крейте питание напряжением +5В и +3.3В. При необходимости на специальном дополнительном шасси может быть установлен вспомогательный блок питания. Для питания обычных модулей ввода-вывода дополнительный блок питания не требуется.

ЦПУ моделей IC200CPU005 и IC200CPUE05 требуют использования источника питания с увеличенной мощностью выхода 3.3В. См. таблицу ниже.



Поставляемые источники питания и шасси.

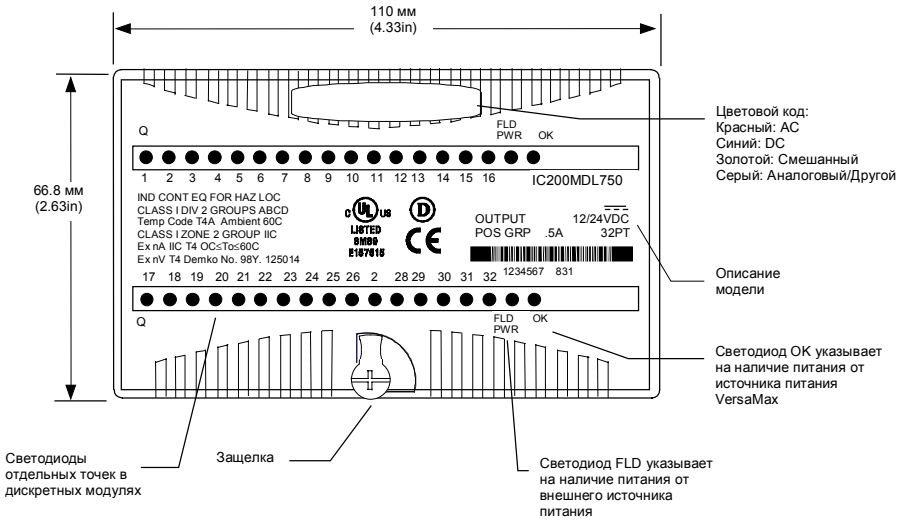
Поставляются следующие источники питания и шасси VersaMax:

24VDC Блок питания	IC200PWR001
24VDC Блок питания, расширенный выход 3.3В	IC200PWR002
120/240VAC Блок питания	IC200PWR101
120/240VAC Блок питания, расширенный выход 3.3В	IC200PWR102
12VDC Блок питания	IC200PWR201
12VDC Блок питания, расширенный выход 3.3В	IC200PWR202
Дополнительное шасси блока питания	IC200PWB001

Источники питания описаны в документе *Модули, источники питания и шасси VersaMax. Руководство пользователя (VersaMax Modules, Power Suppllies, and Carriers User's Manual)* (GFK-1504).

Модули ввода-вывода

Размеры модулей ввода-вывода и дополнительных модулей VersaMax составляют примерно 110 мм (4.33 дюйма) на 66.8 мм (2.63 дюйма). Модули могут быть установлены на шасси ввода-вывода нескольких типов. Глубина модулей – 50 мм (1.956 дюйма) без учета высоты шасси и подключенных разъемов.



Модули ввода-вывода VersaMax описаны в *Модули, источники питания и шасси VersaMax. Руководство пользователя (VersaMax Modules, Power Suppllies, and Carriers User's Manual)* (GFK-1504).

Поставляемые модули ввода-вывода

Поставляются следующие типы модулей ввода-вывода VersaMax:

Дискретные модули ввода	
Модуль ввода 120VAC 8 каналов (1 группа)	IC200MDL140
Модуль ввода 240VAC 8 каналов (1 группа)	IC200MDL141
Модуль ввода 120VAC 8 изолированных каналов	IC200MDL143
Модуль ввода 240VAC 4 изолированных каналов	IC200MDL144
Модуль ввода 120VAC 16 каналов (2 группы по 8)	IC200MDL240
Модуль ввода 240VAC 16 каналов (2 группы по 8)	IC200MDL241
Модуль ввода 120VAC 16 изолированных каналов	IC200MDL243
Модуль ввода 240VAC 8 изолированных каналов	IC200MDL244
Модуль ввода 125VDC 8 каналов (1 группа), положительная/отрицательная логика	IC200MDL631
Модуль ввода 125VDC 16 каналов (1 группа), положительная/отрицательная логика	IC200MDL632
Модуль ввода 48VDC 16 каналов (1 группа), положительная/отрицательная логика	IC200MDL635
Модуль ввода 48VDC 32 канала (1 группа), положительная/отрицательная логика	IC200MDL636
Модуль ввода 24VDC 16 каналов (2 группы по 8), положительная/отрицательная логика	IC200MDL640
Модуль ввода 5/12VDC (ТТЛ) 16 каналов, положительная/отрицательная логика	IC200MDL643
Модуль ввода 5/12VDC (ТТЛ) 32 канала (1 группа), положительная/отрицательная логика	IC200MDL644
Модуль ввода 24VDC 32 канала (4 группы по 8), положительная/отрицательная логика	IC200MDL650
Дискретные модули вывода	
Модуль вывода 120VAC, 8 изолированных каналов, 0.5A на канал	IC200MDL329
Модуль вывода 120VAC, 16 изолированных каналов, 0.5A на канал	IC200MDL330
Модуль вывода 120VAC, 8 изолированных каналов, 2A на канал	IC200MDL331
Модуль вывода 24VDC, 8 каналов (1 группа), 2A на канал, положительная логика, с электронной защитой от короткого замыкания	IC200MDL730
Модуль вывода 12/24VDC, 16 каналов (1 группа), 0.5A на канал, положительная логика	IC200MDL740
Модуль вывода 24VDC, 16 каналов (1 группа), 0.5A на канал, положительная логика, с электронной защитой от короткого замыкания	IC200MDL741
Модуль вывода 24VDC, 32 канала (2 группы по 16), 0.5A на канал, положительная логика, с электронной защитой от короткого замыкания	IC200MDL742
Модуль вывода 5/12/24VDC, 16 каналов (1 группа), 0.5A на канал, отрицательная логика	IC200MDL743
Модуль вывода 5/12/24VDC, 32 канала (2 группы по 16), 0.5A на канал, отриц. логика	IC200MDL744
Модуль вывода 12/24VDC, 32 канала (2 группы по 16), 0.5A на канал, полож. логика	IC200MDL750
Релейный модуль вывода, 8 изолированных каналов, 2A на канал, НО	IC200MDL930
Релейный модуль вывода, 16 изолированных каналов, 2A на канал, НО	IC200MDL940

Комбинированные дискретные модули ввода-вывода	
Комбинированный модуль 24VDC, 20 входов (1 группа), положительная логика / 12 релейных выходов (1 группа), 2.0A на канал	IC200MDD840
Комбинированный модуль 24VDC, 20 входов / 12 выходов/ (4) высокоскоростных счетчика, широтно-импульсных модулятора или импульсных выхода	IC200MDD841
Комбинированный модуль 24VDC, 16 входов (1 группа), положительная/отрицательная логика / 16 выходов (1 группа), положительная логика, 0.5A на канал, с электронной защитой от КЗ	IC200MDD842
Комбинированный модуль 24VDC, 10 входов (1 группа), положительная логика / 6 релейных выходов, 2.0A на канал	IC200MDD843
Комбинированный модуль 24VDC, 16 входов (1 группа), положительная/отрицательная логика / 16 выходов 12/24VDC, положительная логика, 0.5A на канал	IC200MDD844
Комбинированный модуль 24VDC, 16 входов (1 группа), положительная/отрицательная логика / 8 изолированных релейных выходов, 2.0A на канал, НО	IC200MDD845
Комбинированный модуль 120VAC 8 входов / 8 релейных выходов, 2.0A на канал	IC200MDD846
Комбинированный модуль 240VAC 8 входов / 8 релейных выходов, 2.0A на канал	IC200MDD847
Комбинированный модуль 120VAC 8 входов / 8 выходов 120VAC 0.5A на канал	IC200MDD848
Комбинированный модуль 120VAC 8 изолированных входов / 8 изолированных релейных выходов, 2.0A на канал	IC200MDD849
Комбинированный модуль 240VAC 4 изолированных входа / 8 изолированных релейных выходов, 2.0A на канал	IC200MDD850
Аналоговые модули ввода	
Аналоговый модуль ввода, 12 бит, Напряжение/Ток, 4 канала	IC200ALG230
Аналоговый модуль ввода, 16 бит, Напряжение/Ток, изоляция 1500VAC, 4 канала	IC200ALG240
Аналоговый модуль ввода, 12 бит, Напряжение/Ток, 8 каналов	IC200ALG260
Аналоговый модуль ввода, 15 бит, Дифференциальное напряжение, 8 каналов	IC200ALG261
Аналоговый модуль ввода, 16 бит, Дифференциальный ток, 8 каналов	IC200ALG262
Аналоговый модуль ввода, 15 бит, Напряжение, 15 каналов	IC200ALG263
Аналоговый модуль ввода, 15 бит, Ток, 15 каналов	IC200ALG264
Аналоговый модуль ввода, 16 бит, Термосопротивление, 4 канала	IC200ALG620
Аналоговый модуль ввода, 16 бит, Термопара, 7 каналов	IC200ALG630
Аналоговые модули вывода	
Аналоговый модуль вывода, 12 бит, Ток, 4 канала	IC200ALG320
Аналоговый модуль вывода, 12 бит, Напряжение, 4 канала, диапазон от 0 до +10VDC	IC200ALG321
Аналоговый модуль вывода, 12 бит, Напряжение, 4 канала, диапазон от -10 до +10VDC	IC200ALG322
Аналоговый модуль вывода, 13 бит, Напряжение, 8 каналов	IC200ALG325
Аналоговый модуль вывода, 12 бит, Ток, 8 каналов	IC200ALG326
Аналоговый модуль вывода, 13 бит, Напряжение, 12 каналов	IC200ALG327
Аналоговый модуль вывода, 12 бит, Ток, 12 каналов	IC200ALG328
Аналоговый модуль вывода, 16 бит, Напряжение/Ток, изоляция 1500VAC, 4 канала	IC200ALG331
Комбинированные аналоговые модули ввода-вывода	
Аналоговый комбинированный модуль, вход – Ток, 4 канала, выход – Ток, 2 канала	IC200ALG430
Аналоговый комбинированный модуль, вход 0 +10VDC, 4 канала, выход 0 +10VDC 2 канала	IC200ALG431
Аналоговый комбинированный модуль, 12 бит, вход -10 +10VDC, 4 канала / выход -10 +10VDC 2 канала	IC200ALG432

Шасси

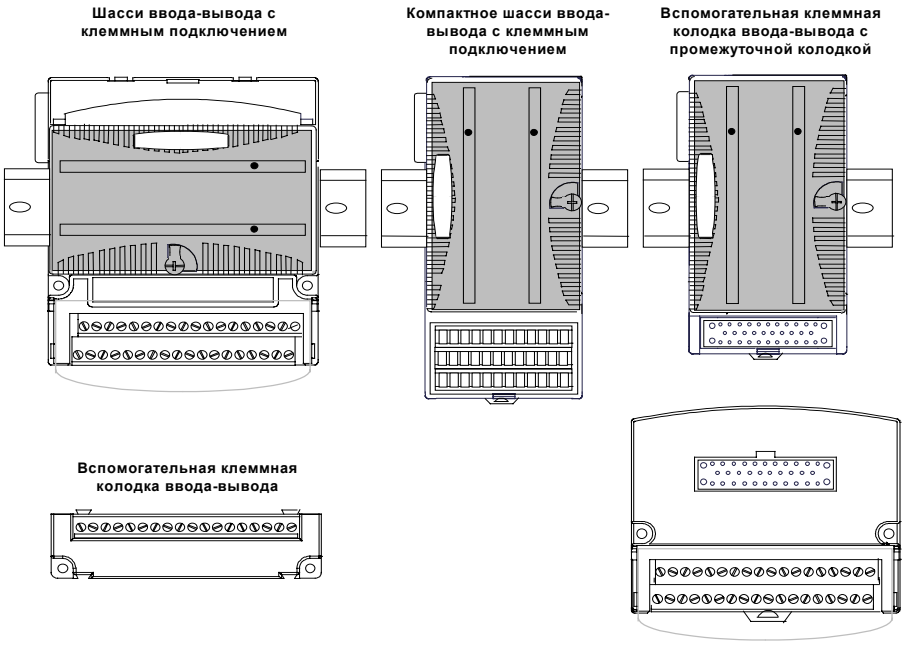
Шасси обеспечивает установку, связь с системной шиной и внешнее подключение для модулей VersaMax всех типов. Модули ввода-вывода могут быть установлены на шасси или сняты без нарушения внешних подключений.

Существуют три основных типа шасси ввода-вывода:

- Шасси ввода-вывода с клеммным подключением. Модули устанавливаются параллельно DIN-рейке.
- Компактное шасси ввода-вывода с клеммным подключением. Модули устанавливаются перпендикулярно DIN-рейке.
- Шасси ввода-вывода с разъемом. Модули устанавливаются перпендикулярно DIN-рейке. Эти шасси обычно используются с промежуточным клеммной колодкой ввода-вывода, как показано ниже.

Информация о шасси ввода-вывода VersaMax приведена в документе *Модули, источники питания и шасси VersaMax. Руководство пользователя (VersaMax Modules, Power Suppllies, and Carriers User's Manual)* (GFK-1504).

Шасси ввода-вывода с клеммным подключением имеют 36 отдельных клемм для прямого подключения внешних проводов. В случаях, когда требуются дополнительные клеммы, может использоваться вспомогательная клеммная колодка ввода-вывода.



Поставляемые шасси и клеммные колодки

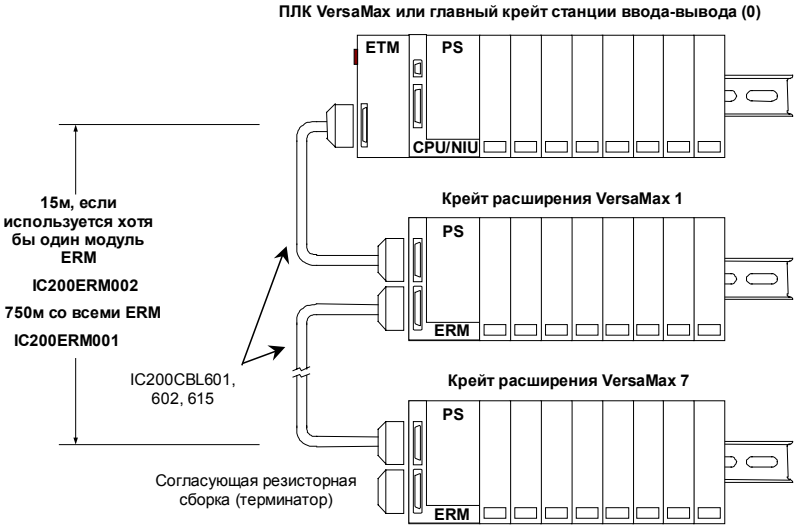
Поставляются следующие шасси, клеммные колодки и кабели:

Шасси ввода-вывода с клеммным подключением	
Шасси ввода-вывода с клеммами под винт	IC200CHS001
Шасси ввода-вывода с клеммами по стандарту IEC	IC200CHS002
Шасси ввода-вывода типа с пружинными клеммами	IC200CHS005
Компактное шасси ввода-вывода с клеммным подключением	
Компактное шасси ввода-вывода с клеммами по стандарту IEC	IC200CHS022
Компактное шасси ввода-вывода типа с пружинными клеммами	IC200CHS025
Шасси ввода-вывода с разъемом	
Шасси ввода-вывода с разъемом	IC200CHS003
Промежуточная клеммная колодка для использования с шасси ввода-вывода с разъемом	
Промежуточная клеммная колодка ввода-вывода с клеммами под винт	IC200CHS011
Промежуточная клеммная колодка ввода-вывода с клеммами по стандарту IEC	IC200CHS012
Промежуточная клеммная колодка ввода-вывода для термопар	IC200CHS014
Промежуточная клеммная колодка ввода-вывода с пружинными клеммами	IC200CHS015
Кабели для использования с шасси ввода-вывода с разъемом	
2 разъема, 0.5м, с экраном	IC200CBL305
2 разъема, 1.0м, с экраном	IC200CBL310
2 разъема, 2.0м, с экраном	IC200CBL320
1 разъем, 3.0м, с экраном	IC200CBL430
2 разъема, 0.5м, без экрана	IC200CBL105
2 разъема, 1.0м, без экрана	IC200CBL110
2 разъема, 2.0м, без экрана	IC200CBL120
1 разъем, 3.0м, без экрана	IC200CBL230
Вспомогательные клеммные колодки ввода-вывода для использования с шасси ввода-вывода с клеммным подключением и промежуточными клеммными колодками	
Вспомогательные колодки ввода-вывода с клеммами под винт	IC200TBM001
Вспомогательные колодки ввода-вывода с клеммами по стандарту IEC	IC200TBM002
Вспомогательные колодки ввода-вывода с пружинными клеммами	IC200TBM005
Другие шасси	
Коммуникационное шасси	IC200CHS006
Дополнительное шасси блока питания	IC200PWB001

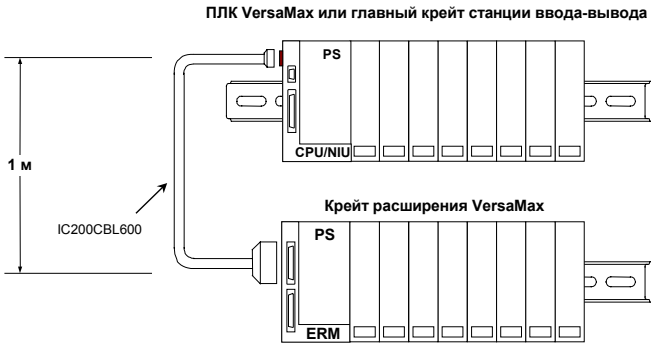
Модули расширения

Существуют два основных типа систем расширения ввода-вывода VersaMax, с мульти-крейтовым подключением и с единичным подключением.

- **Мульти-крейтовое подключение:** ПЛК VersaMax или станция ввода-вывода на базе устройства сетевого интерфейса с передающим модулем расширения (IC200ETM001) и от одного до семи крейтов расширения, каждый с принимающим модулем расширения (IC200ERM001 или IC200ERM002). Если все принимающие модули расширения изолированного типа (IC200ERM001), максимальная общая длина кабеля составляет 750 метров. Если шина расширения включает неизолированный принимающий модуль расширения (IC200ERM002), максимальная общая длина кабеля составляет 15 метров.



- **Единичное подключение:** ПЛК VersaMax или станция ввода-вывода на базе устройства сетевого интерфейса подключаются напрямую к одному крейту расширения с неизолированным передающим модулем расширения (IC200ERM002). Максимальная длина кабеля – 1 метр.



Модули VersaMax для крейтов расширения

Все типы модулей ввода-вывода и коммуникационных модулей VersaMax могут быть использованы в крейтах расширения. Некоторые аналоговые модули VersaMax должны быть определенной версии, как указано ниже.

Модуль	Версия модуля
IC200ALG320	B или выше
IC200ALG321	B или выше
IC200ALG322	B или выше
IC200ALG430	C или выше
IC200ALG431	C или выше
IC200ALG432	B или выше

Поставляемые модули расширения

Поставляются следующие модули расширения и аксессуары к ним:

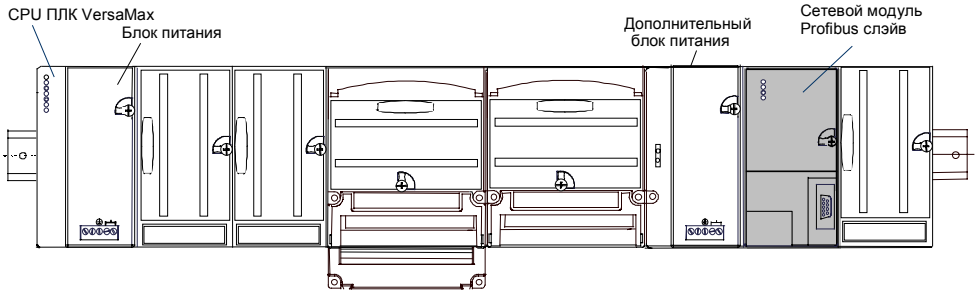
Модули расширения	
Передающий модуль расширения	IC200ETM001
Принимающий модуль расширения, изолированный	IC200ERM001
Принимающий модуль расширения, неизолированный	IC200ERM002
Кабели	
Кабель расширения, 1 метр	IC200CBL601
Кабель расширения, 2 метра	IC200CBL602
Кабель расширения, 15 метров	IC200CBL615
Кабель обновления фирменного программного обеспечения	IC200CBL002
Согласующая резисторная сборка (терминатор) (входит в комплект ETM)	IC200ACC201
Комплект разъемов	IC200ACC302

Информация о модулях расширения VersaMax приведена в документе *Модули, источники питания и шасси VersaMax. Руководство пользователя (VersaMax Modules, Power Suppllies, and Carriers User's Manual)* (GFK-1504).

Коммуникационные модули

Коммуникационные модули придают системам VersaMax дополнительную гибкость.

Коммуникационные модули устанавливаются на коммуникационные шасси VersaMax. Питание к коммуникационным модулям подается от основного блока питания системы или от вспомогательного блока питания, как показано ниже.



Поставляемые коммуникационные модули ПЛК VersaMax

Поставляются следующие коммуникационные модули ПЛК VersaMax:

Коммуникационные модули	
Сетевой модуль Profibus-DP слэйв	IC200BEM002
Сетевой модуль DeviceNet	IC200BEM103
Коммуникационное шасси	IC200CHS006

Информация о коммуникационных шасси приведена в документе *Модули, источники питания и шасси VersaMax. Руководство пользователя (VersaMax Modules, Power Suppllies, and Carriers User's Manual)* (GFK-1504).

Сетевой модуль Profibus-DP слэйв

Сетевой модуль Profibus DP слэйв (IC200BEM002) – коммуникационный модуль, который передает данные ПЛК по сети Profibus. ЦПУ ПЛК VersaMax может читать и записывать эти данные, как если бы они были обычными битами и словами ввода-вывода.

Несколько сетевых модулей Profibus DP слэйв могут использоваться в одном ПЛК VersaMax. Каждый может читать до 244 байт данных из сети, и передавать до 244 байт выходных данных. Общее количество входных и выходных данных – 384 байт.

Информация о сетевом модуле Profibus DP слэйв приведена в документе *Коммуникационные модули Profibus VersaMax Руководство пользователя (VersaMax Profibus Communications Modules User's Manual)* (GFK-1534, версия A или выше).

Сетевой модуль DeviceNet

Сетевой модуль DeviceNet (IC200BEM103) – коммуникационный модуль, который может быть сконфигурирован для работы как мастер, так и слэйв, или в обоих режимах одновременно. Он может обмениваться данными (до 512 байт входных данных и 512 байт выходных данных) с другими устройствами по сети DeviceNet. ПЛК VersaMax может читать и записывать эти данные, как если бы они были обычными битами и словами ввода-вывода.

Сетевой модуль DeviceNet работает как клиент только Group 2 (мастер) и может связываться только с устройствами Group 2 слэйв. Он также может работать как сервер Group 2 или UCMM (слэйв), или как мастер и слэйв одновременно.

Информация о сетевом модуле DeviceNet приведена в документе *Коммуникационные модули DeviceNet VersaMax Руководство пользователя (VersaMax DeviceNet Communications Modules User's Manual)* (GFK-1533).

Глава
2

Модули ЦПУ: CPU001, CPU002, CPU005

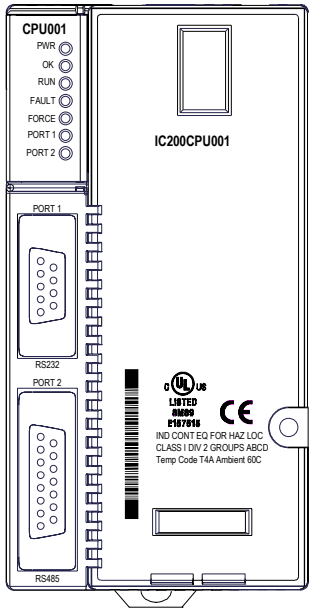
В этой главе описаны особенности и функциональные возможности следующих модулей ЦПУ ПЛК VersaMax:

- IC200CPU001 ЦПУ с 34kБ конфигурируемой памяти
- IC200CPU002 ЦПУ с 42kБ конфигурируемой памяти
- IC200CPU005 ЦПУ с 64kБ конфигурируемой памяти

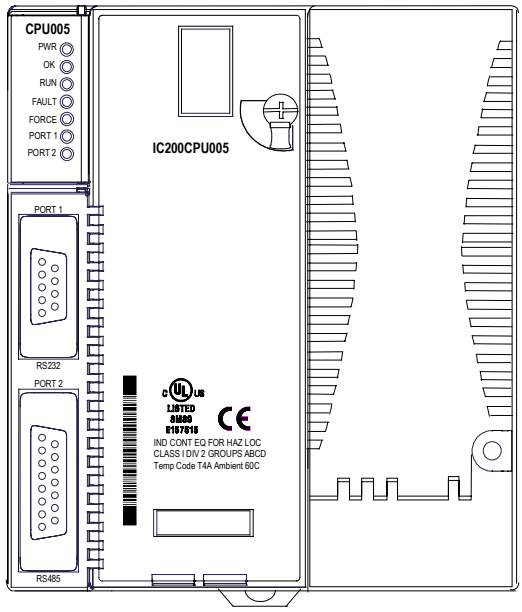
IC200CPU001: ЦПУ с 34кБ конфигурируемой памяти
IC200CPU002: ЦПУ с 42кБ конфигурируемой памяти
IC200CPU005: ЦПУ с 64кБ конфигурируемой памяти

ЦПУ IC200CPU001, CPU002 и CPU005 ПЛК VersaMax® обеспечивают широкие функциональные возможности ПЛК в небольших универсальных системах. Они разработаны для построения систем управления, включающих до 64 модулей и до 2048 каналов ввода-вывода. Два последовательных порта обеспечивают интерфейсы RS-232 and RS-485 для связи по протоколам SNP слэйв и RTU слэйв.

CPU001, CPU002



CPU005



Основные особенности

- Энергонезависимая флэш-память для хранения программ
- Программирование на языках релейно-контактной логики, функциональных схем и текстовых команд
- Батарейная поддержка программ, данных и времени суток
- Переключатель режима Run/Stop
- Обработка данных с плавающей точкой
- Встроенные порты RS-232 и RS-485
- Высота 70мм при монтаже на DIN-рейке с блоком питания
- Совместимость с устройством хранения программ EZ

ЦПУ с 34kБ конфигурируемой памяти: IC200CPU001
ЦПУ с 42kБ конфигурируемой памяти: IC200CPU002
ЦПУ с 64kБ конфигурируемой памяти: IC200CPU005

Спецификации модулей

Размер	CPU001/002: 2.63" (66.8мм) x 5.04" (128мм) CPU005: 4.20" (106.7мм) x 5.04" (128мм)		
Хранение программ	Системная флэш-память и ОЗУ с батарейной поддержкой		
Потребление электроэнергии: IC200CPU001, IC200CPU002	Без конвертера последовательного порта или устройства хранения программ EZ	5В 40мА	3.3В, 100мА
	С конвертером последовательного порта или устройства хранения программ EZ	5В 140мА	
Потребление электроэнергии: IC200CPU005	Без конвертера последовательного порта или устройства хранения программ EZ	5В 80мА	3.3В 290мА*
	С конвертером последовательного порта или устройства хранения программ EZ	5В 180мА	
Плавающая точка	Да		
Встроенные порты	RS-232, RS-485		
Скорость выполнения булевых операций	CPU001, CPU002: 1.8mc/K (типовая) CPU005: 0.5мс/K (типовая)		
Точность часов реального времени	100ppm (0.01%) или +/- 9с/день		
Точность часов времени суток	23ppm (0.0023%) или +/- 2с/день @ 30С. 100 ppm (0.01%) или +/- 9с/день @ весь температурный диапазон		

* Для CPU005 требуется источник питания с увеличенной мощностью выхода 3.3В.

ЦПУ с 34кБ конфигурируемой памяти: IC200CPU001
ЦПУ с 42кБ конфигурируемой памяти: IC200CPU002
ЦПУ с 64кБ конфигурируемой памяти: IC200CPU005

Общие спецификации изделий VersaMax

Изделия VersaMax должны устанавливаться и использоваться в соответствии со своими описания, а также в соответствии со следующими спецификациями:

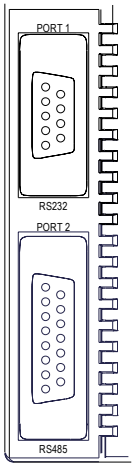
Условия окружающей среды		
Вибрация	IEC68-2-6	1G @57-150Гц, 0.012" p--p @10-57Гц
Удар	IEC68-2-27	15G, 11мс
Рабочая температура		0^{O} C to $+60^{O}$ C
Температура хранения		-40^{O} C to $+85^{O}$ C
Влажность		5% - 95%, без конденсата
Корпусная защита	IEC529	Стальной шкаф IP54: защита от пыли и брызг воды
Электромагнитное излучение		
Излучаемое, проводимое	CISPR 11/EN 55011	Промышленное Научное и Медицинское Оборудование (Group 1, Class A)
	CISPR 22/EN 55022	Оборудование информационных технологий (Class A)
	FCC 47 CFR 15	Соответствует FCC part 15, Радиоустройств (Class A)
Электромагнитная устойчивость		
Электростатический разряд	EN 61000-4-2	8кВ воздух, 4кВ контакт
РЧ чувствительность	EN 61000-4-3	$10В_{rms}$ /м, 80МГц - 1000МГц, 80% AM
	ENV 50140/ENV 50204	$10В_{rms}$/м, 900 МГц +/-5 МГц 100%AM при меандре 200Гц
Импульс быстрого переходного режима	EN 61000-4-4	2кВ: источники питания, 1кВ: ввод-вывод, связь
Устойчивость к импульсам	ANSI/IEEE C37.90a	Затухающие колебания: 2.5кВ источники питания, ввод-вывод [12В-240В]; 1кВ связь
	IEC255-4	Затухающие колебания: Class II, источники питания, ввод-вывод [12В-240В]
	EN 61000-4-5	2 кВ см(источники питания); 1 кВ см (ввод-вывод и связь)
РЧ проводимость	EN 61000-4-6	$10В_{rms}$, 0.15 - 80Mhz, 80%AM
Изоляция		

Диэлектрическая прочность	UL508, UL840, IEC664	1.5кВ
Источник питания		
Провалы входного напряжения, отклонения	EN 61000-4-11	При работе: провалы до 30% и 100%, Отклонение для переменного тока +/-10%, отклонение для постоянного тока +/-20%

IC200CPU001: ЦПУ с 34kБ конфигурируемой памяти
IC200CPU002: ЦПУ с 42kБ конфигурируемой памяти
IC200CPU005: ЦПУ с 64kБ конфигурируемой памяти

Последовательные порты.

Два последовательных порта программно конфигурируются для работы в качестве SNP слэйв или RTU слэйв. Поддерживаются 4-проводной и 2-проводной интерфейсы RTU. Если порт используется как RTU-интерфейс, он автоматически переключается в режим SNP слэйв, если требуется. Оба порта по умолчанию находятся в режиме SNP слэйв и оба автоматически переходят в режим SNP слэйв, когда ЦПУ находится в режиме Stop, если они сконфигурированы как Serial I/O. Оба порта могут быть программно сконфигурированы для обеспечения связи между ЦПУ и различными последовательными устройствами. Внешнее устройство может получать питание от Порта 2, если это требуется: 100мА, 5В.



Порт 1: RS-232 порт с 9-контактной розеткой типа D-sub. Выводы Порта 1 позволяют использовать обычный кабель для соединения со стандартным портом RS-232 типа AT.

Порт 2: RS-485 порт с 15-контактной розеткой типа D-sub. Он может быть подключен к адаптеру RS-485/RS-232 (IC690ACC901) напрямую.

В таблице сравниваются возможности Порта 1 и Порта 2.

	Порт 1	Порт 2
Протоколы ЦПУ (SNP слэйв, RTU слэйв, Serial I/O)	По умолчанию SNP слэйв	По умолчанию SNP слэйв
Обновление фирменного программного обеспечения	ПЛК в режиме Stop/No I/O.	Нет
Обновление фирменного программного обеспечения интеллектуального модуля	ПЛК в режиме Stop/No I/O.	ПЛК в режиме Stop/No I/O.

Длина кабеля

Максимальная длина кабеля – общее расстояние от ЦПУ до последнего устройства, подключенного к кабелю:

Портt 1 (RS-232) = 15 метров (50 футов)
Порт 2 (RS-485) = 1200 метров (4000 футов)

ЦПУ с 34кБ конфигурируемой памяти: IC200CPU001
ЦПУ с 42кБ конфигурируемой памяти: IC200CPU002
ЦПУ с 64кБ конфигурируемой памяти: IC200CPU005

Скорость обмена по последовательным портам

	CPU001, CPU002	CPU005
Протокол RTU	1200, 2400, 4800, 9600, 19.2K	1200, 2400, 4800, 9600, 19.2K, 38.4K, 57.6K**
Протокол Serial I/O	4800, 9600, 19.2K	4800, 9600, 19.2K, 38.4K, 57.6K**
Протокол SNP	4800, 9600, 19.2K, 38.4K*	4800, 9600, 19.2K, 38.4K*
Обновление фирменного программного обеспечения через WInloader	2400, 4800, 9600, 19.2K, 38.4K	Не используется

* Одновременно возможно только на одном порту.

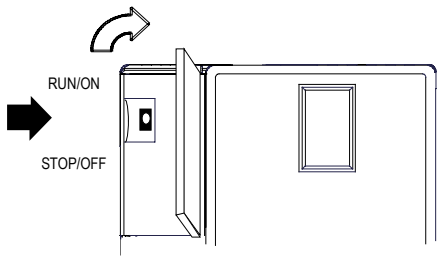
** Программное обеспечение VersaPro позволяет сконфигурировать протоколы RTU и Serial I/O на скорость 115.2Кбод. Однако, ЦПУ не поддерживает эту скорость. Если конфигурация, использующая эту скорость, сохранена в ПЛК, то:

1. Для протокола RTU, регистрируется ошибка “Unsupported Feature in Configuration” и ПЛК переходит в режим Stop Faulted.
2. Для протокола Serial I/O, регистрируется та же самая ошибка, при переходе в режим Run. ПЛК немедленно перейдет в режим Stop Faulted.

Переключатель режима

Модуль ЦПУ имеет удобный переключатель, который может использоваться для установки ПЛК в режим Stop или Run. Этот же выключатель может быть также использован для защиты от случайной записи в память ЦПУ и принудительной установки или сброса дискретных данных. Использование этого переключателя конфигурируется.

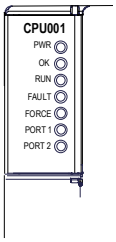
Конфигурация по умолчанию позволяет переключать режимы Run/Stop и отключает защиту памяти.



IC200CPU001: ЦПУ с 34кБ конфигурируемой памяти
IC200CPU002: ЦПУ с 42кБ конфигурируемой памяти
IC200CPU005: ЦПУ с 64кБ конфигурируемой памяти

Светодиоды ЦПУ

Семь светодиодов ЦПУ, видимые через дверцу модуля, показывают наличие питания, режим работы и состояние ЦПУ. Они также указывают на наличие ошибок, принудительно установленных данных, и связи через два порта ЦПУ.



PWR Светится, когда ЦПУ получает питание 5В от источника питания. Не отражает состояние выхода 3.3В источника питания.

OK Свечение означает, что ЦПУ прошло самодиагностику при включении питания и работает нормально. Отсутствие свечения указывает на проблемы с ЦПУ. Быстрое мигание означает, что ЦПУ выполняет самодиагностику после включения питания. Медленное мигание означает, что ЦПУ конфигурирует модули ввода-вывода. Одновременное мигание этого светодиода и зеленого светодиода Run означает, что ЦПУ находится в режиме загрузки и ожидает обновления фирменного программного обеспечения через Порт 1.

RUN Зеленый, если ЦПУ в режиме Run. Желтый, если ЦПУ в режиме Stop/IO Scan. Если этот светодиод не светится, а светодиод OK светится, ЦПУ находится в режиме Stop/No IO Scan.

Если этот светодиод мигает зеленым, а светодиод Fault светится, переключатель модуля был переведен из положения Stop в положение Run, хотя имелась фатальная ошибка. ЦПУ находится в режиме RUN.

FAULT Светится, если ЦПУ находится в режиме Stop/Faulted вследствие фатальной ошибки. Чтобы светодиод Fault погас, нужно очистить и таблицу ошибок ввода-вывода (I/O Fault Table) и таблицу ошибок ПЛК (PLC Fault Table). Если этот светодиод мигает и светодиод OK не горит, фатальная ошибка была обнаружена во время выполнения диагностики ПЛК при включении питания. Свяжитесь с отделом обслуживания ПЛК.

FORCE Светится, если битовая ячейка принудительно установлена.

PORT 1
PORT 2 Мигание указывает на активность на этом порту.

ЦПУ с 34kБ конфигурируемой памяти: IC200CPU001
ЦПУ с 42kБ конфигурируемой памяти: IC200CPU002
ЦПУ с 64kБ конфигурируемой памяти: IC200CPU005

Конфигурируемая память

CPU001 и CPU002 (версия 2.0 или выше) и CPU005 имеют конфигурируемую пользовательскую память. Конфигурируемая память – объем памяти, требуемый для прикладной программы, конфигурации оборудования, регистров (%R), аналоговых входов (%AI), и аналоговых выходов (%AQ).

Объем памяти, отведенный прикладной программе и конфигурации оборудования, автоматически определяется в соответствии с программой и конфигурацией, введенной через программатор. Остальная часть конфигурируемой памяти может быть легко распределена так, чтобы соответствовать приложению.

Конфигурируемая память	CPU001: 34K байт максимум CPU002: 42K байт максимум CPU005: 64K байт максимум
Размер прикладной программы (не конфигурируемый) CPU001, для совместимости с вер. 1.50 CPU002, для совместимости с вер. 1.50	128 байт минимум 12K байт 20K байт
Размер конфигурации оборудования (не конфигурируемый)	400 байт минимум
Регистры (%R) CPU001/002, для совместимости с вер. 1.50	256 байт минимум 4,096 байт
Аналоговые входы (%AI)	256 байт минимум
Аналоговые выходы (%AQ)	256 байт минимум

Глава 3

Модуль ЦПУ: CPUE05

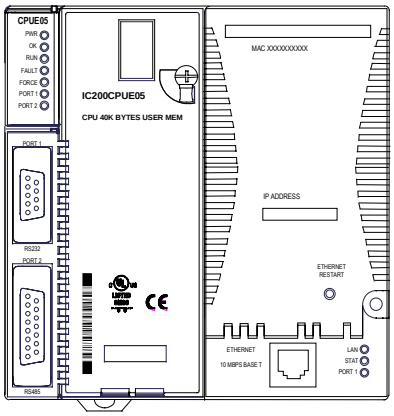
В этой главе описаны особенности и функциональные возможности следующего модуля ЦПУ ПЛК VersaMax:

- IC200CPUE05: ЦПУ с двумя последовательными портами, встроенным интерфейсом Ethernet и 64кБ конфигурируемой памяти

IC200CPUE05: ЦПУ с двумя последовательными портами, встроенным интерфейсом Ethernet и 64K конфигурируемой памяти

ЦПУ IC200CPUE05 ПЛК VersaMax® обладает основными чертами остальных ЦПУ ПЛК VersaMax. Он обеспечивает широкие функциональные возможности ПЛК в небольших универсальных системах. CPUE05 может использоваться в контроллерах, включающих до 64 модулей и до 2048 каналов ввода-вывода. Два последовательных порта обеспечивают интерфейсы RS-232 и RS-485. CPUE05 также включает в себя встроенный интерфейс Ethernet. Последовательный порт RS-232 может быть сконфигурирован для работы с монитором станции, что обеспечивает доступ к диагностической информации интерфейса Ethernet. CPUE05 имеет 64кБ конфигурируемой памяти.

Также CPUE05 совместимо с устройством хранения программ EZ, которое может использоваться для записи, чтения, обновления и проверки программ, конфигурации и данных без использования программатора или инструментального программного обеспечения.



Основные особенности

- 64Кб конфигурируемой памяти
- Программирование на языках релейно-контактной логики, функциональных схем и текстовых команд
- Совместимость с устройством хранения программ EZ
- Энергонезависимая флэш-память для хранения программ
- Батарейная поддержка программ, данных и времени суток
- Переключатель режима Run/Stop
- Обработка данных с плавающей точкой
- Встроенные порты RS-232 и RS-485
- Встроенный интерфейс Ethernet
- Высота 70мм при монтаже на DIN-рейке с источником питания

ЦПУ с двумя последовательными портами, встроенным интерфейсом Ethernet и 64K конфигурируемой памяти: IC200CPUE05

Спецификации модуля

Размер	4.95" (126мм) x 5.04" (128мм)		
Хранение программ	Флэш-память, ОЗУ с батарейной поддержкой		
Потребление электроэнергии: IC200CPUE05	Без конвертера последовательного порта или устройства хранения программ EZ	5В 160мА	3.3В 650мА
	С конвертером последовательного порта или устройства хранения программ EZ	5В 260мА	
Плавающая точка	Да		
Скорость выполнения булевых операций	0.5мс/К (типовая)		
Точность часов реального времени	100ppm (0.01%) или +/- 9с/день		
Точность часов времени суток	23ppm (0.0023%) или +/- 2с/день @ 30С. 100 ppm (0.01%) или +/- 9с/день @ весь температурный диапазон		
Встроенные порты	RS-232, RS-485, интерфейс Ethernet		
Конфигурируемая память	64Кбайт максимум		
Спецификация интерфейса Ethernet			
Кол-во подключений сервера SRTP	8		
Скорость передачи данных	10Mbps		
Физический интерфейс	10BaseT RJ45		
Поддержка WinLoader	Через порт ЦПУ		
Кол-во конфигурируемых каналов обмена Ethernet Global Data	32		
Ограничения на обмен EGD	100 диапазонов данных и 1400 байт данных за обмен; 1200 диапазонов данных всего.		
Временная синхронизация	Только NTP - клиент		
Выборочный прием EGD	Да		
Загрузка конфигурации EGD из ПЛК в программатор	Да		
Удаленный монитор станции через UDP	Да		
Локальный монитор станции (RS-232)	Через порт ЦПУ		
Конфигурируемые дополнительные пользовательские параметры	Да		

* Для CPUE05 требуется блок питания с повышенной нагрузочной способностью по выходу 3.3В.

IC200CPUE05: ЦПУ с двумя последовательными портами, встроенным интерфейсом Ethernet и 64K конфигурируемой памяти

Общие спецификации изделий VersaMax

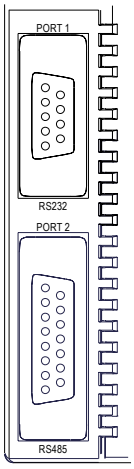
Изделия VersaMax должны устанавливаться и использоваться в соответствии с требованиями на изделия, а также в соответствии со следующими спецификациями:

Условия окружающей среды		
Вибрация	IEC68-2-6	1G @57-150Гц, 0.012" p--p @10-57Гц
Удар	IEC68-2-27	15G, 11мс
Рабочая температура		0^{O} C to $+60^{O}$ C
Температура хранения		-40^{O} C to $+85^{O}$ C
Влажность		5% - 95%, без конденсата
Корпусная защита	IEC529	Стальной шкаф для защиты IP54: от пыли и брызг
Электромагнитное излучение		
Излучаемое, проводимое	CISPR 11/EN 55011	Промышленное Научное и Медицинское Оборудование (Group 1, Class A)
	CISPR 22/EN 55022	Оборудование информационных технологий (Class A)
	FCC 47 CFR 15	Соответствует FCC part 15, Радиоустройств (Class A)
Электромагнитная устойчивость		
Электростатический разряд	EN 61000-4-2	8кВ воздух, 4кВ контакт
РЧ чувствительность	EN 61000-4-3	$10B_{rms}$ /м, 80МГц - 1000МГц, 80% AM
	ENV 50140/ENV 50204	$10B_{rms}$/м, 900 МГц +/-5 МГц 100%AM при меандре 200Гц
Импульс быстрого переходного режима	EN 61000-4-4	2кВ: источники питания, 1кВ: ввод-вывод, связь
Устойчивость к импульсам	ANSI/IEEE C37.90a	Затухающие колебания: 2.5кВ источники питания, ввод-вывод [12В-240В]; 1кВ связь
	IEC255-4	Затухающие колебания: Class II, источники питания, ввод-вывод [12В-240В]
	EN 61000-4-5	2 кВ см(источники питания); 1 кВ см (ввод-вывод,связь)
РЧ проводимость	EN 61000-4-6	$10B_{rms}$, 0.15 - 80Mhz, 80%AM
Изоляция		
Диэлектрическая прочность	UL508, UL840, IEC664	1.5кВ
Источник питания		
Провалы входного напряжения, отклонения	EN 61000-4-11	При работе: провалы до 30% и 100%, Отклонение для переменного тока +/-10%, отклонение для постоянного тока +/-20%

ЦПУ с двумя последовательными портами, встроенным интерфейсом Ethernet и 64K конфигурируемой памяти: IC200CPUE05

Последовательные порты

Оба последовательных порта программно конфигурируются для работы в качестве SNP-слэйв или RTU-слэйв. Поддерживаются 4-проводной и 2-проводной интерфейсы RTU. Если порт используется в качестве RTU-слэйв, при необходимости он автоматически переключается в режим SNP-слэйв. Порт 1 может быть также сконфигурирован для работы с локальным монитором станции, который обеспечивает доступ к диагностической информации интерфейса Ethernet. Оба порта по умолчанию находятся в режиме SNP слэйв и оба автоматически переходят в режим SNP слэйв, когда ЦПУ находится в режиме Stop, если они сконфигурированы как Serial I/O. Оба порта могут быть программно сконфигурированы для обеспечения связи между ЦПУ и различными последовательными устройствами. Внешнее устройство может получать питание от Порта 2, если это требуется: 100мА, 5В.



Порт 1: RS-232 порт с 9-контактной розеткой типа D-sub. Выводы Порта 1 позволяют использовать обычный кабель для соединения со стандартным портом RS-232 типа AT.

Порт 1 может быть сконфигурирован либо для обеспечения ЦПУ последовательной связью (SNP, RTU, Serial I/O), либо для использования с локальным монитором станции. Если Порт 1 сконфигурирован для работы с ЦПУ, он может быть переведен в режим работы с локальным монитором станции с помощью кнопки Restart. Будучи переведенным, Порт 1 остается доступным для монитора станции пока не будет выключено питание ПЛК, или не будет нажата кнопка Restart.

Если Порт 1 сконфигурирован для работы с локальным монитором станции, он не может быть использован для последовательной связи ЦПУ или для обновления фирменного программного обеспечения с помощью Winloader. Кнопка Restart не будет переключать его в режим последовательных протоколов ЦПУ.

Порт 2: RS-485 порт с 15-контактной розеткой типа D-sub. Он может быть подключен к адаптеру RS-485/RS-232 (IC690ACC901) напрямую. Порт 2 может быть использован для обновления программы, конфигурации и таблиц ссылок с помощью устройства хранения программ EZ.

IC200CPUE05: ЦПУ с двумя последовательными портами, встроенным интерфейсом Ethernet и 64K конфигурируемой памяти

В следующей таблице сравниваются возможности Порта 1 и Порта 2.

	Порт 1	**Порт 2**
Протоколы ЦПУ (SNP слэйв, RTU слэйв, Serial I/O)	По умолчанию SNP слэйв	По умолчанию SNP слэйв
Локальный монитор станции	Да (см. выше)	Нет
Обновление фирменного программного обеспечения	ПЛК в режиме Stop/No I/O, Порт 1 не используется или в режиме локального монитора станции.	Нет
Обновление фирменного программного обеспечения интеллектуального модуля	ПЛК в режиме Stop/No I/O, Порт 1 сконфигурирован для протокола ЦПУ	ПЛК в режиме Stop/No I/O.
Устройство хранения программ EZ	Нет	Чтение, запись, сравнение и обновление. ПЛК в режиме Stop/No I/O.

ЦПУ с двумя последовательными портами, встроенным интерфейсом Ethernet и 64K конфигурируемой памяти: IC200CPUE05

Длина кабеля

Максимальная длина кабеля – общее расстояние от ЦПУ до последнего устройства, подключенного к кабелю:

Портт 1 (RS-232) = 15 метров (50 футов)
Порт 2 (RS-485) = 1200 метров (4000 футов)

Скорость обмена по последовательным портам

	Порт 1	Порт 2
Протокол RTU	1200, 2400, 4800, 9600, 19.2K, 38.4*K, 57.6*K	1200, 2400, 4800, 9600, 19.2K, 38.4*K, 57.6*K
Протокол Serial I/O	4800, 9600, 19.2K, 38.4K*, 57.6K*	4800, 9600, 19.2K, 38.4K*, 57.6K*
Протокол SNP	4800, 9600, 19.2K, 38.4K*	4800, 9600, 19.2K, 38.4K*
Local Station Manager (не зависимо от скорости последовательного порта)	1200, 2400, 4800, 9600, 19.2K, 38.4K, 57.6K, 115.2K	Не используется
Обновление фирменного программного обеспечения через WInloader	2400, 4800, 9600, 19.2K, 38.4K, 57.6K, 115.2K	Не используется

* Одновременно возможно только на одном порту.

Программное обеспечение VersaPro позволяет сконфигурировать протоколы RTU и Serial I/O на скорость 115.2Кбод. Однако, ЦПУ не поддерживает эту скорость. Если конфигурация, использующая эту скорость, сохранена в ПЛК, то:

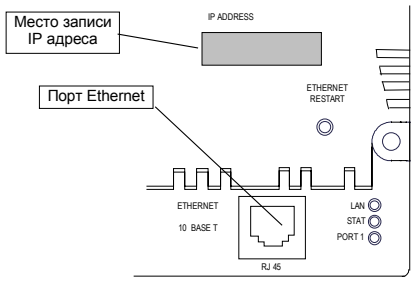
1. Для протокола RTU, регистрируется ошибка “Unsupported Feature in Configuration” и ПЛК переходит в режим Stop Faulted.
2. Для протокола Serial I/O, регистрируется та же самая ошибка, при переходе в режим Run. ПЛК немедленно перейдет в режим Stop Faulted.

IC200CPUE05: ЦПУ с двумя последовательными портами, встроенным интерфейсом Ethernet и 64K конфигурируемой памяти

Порт сети Ethernet

Порт сети Ethernet поддерживает сервер SRTP и обмен данными Ethernet Global Data. Этот порт подключается напрямую к сети 10BaseT (витая пара) без внешнего трансивера. Кабели витой пары 10BaseT должны соответствовать стандартам IEEE 802. CPUE05 автоматически выбирает полнодуплексный или полудуплексный режим в зависимости от сети.

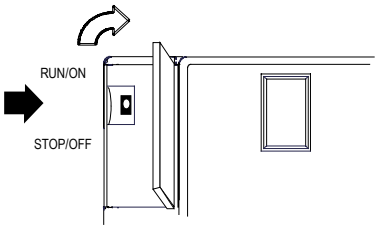
На передней панели модуля CPUE05 выделено место, где может быть записан сконфигурированный IP адрес.



ЦПУ с двумя последовательными портами, встроенным интерфейсом Ethernet и 64K конфигурируемой памяти: IC200CPUE05

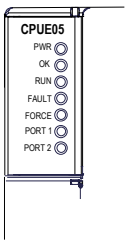
Переключатель режима

Переключатель режима расположен под дверцей модуля. Он может использоваться для установки ПЛК в режим Stop или Run. Этот же выключатель может быть также использован для защиты от случайной записи в память ЦПУ и принудительной установки или сброса дискретных данных. Использование этого переключателя конфигурируется. Конфигурация по умолчанию позволяет переключать режимы Run/Stop и отключает защиту памяти.



Светодиоды ЦПУ

Семь светодиодов ЦПУ, видимые через дверцу модуля, показывают наличие питания, режим работы и состояние ЦПУ. Они также указывают на наличие ошибок, принудительно установленных данных, и связи через два порта ЦПУ.



POWER Светится, когда ЦПУ получает питание 5В от источника питания. Не отражает состояние выхода 3.3В источника питания.

OK Свечение означает, что ЦПУ прошло самодиагностику при включении питания и работает нормально. Отсутствие свечения указывает на проблемы с ЦПУ. Быстрое мигание означает, что ЦПУ выполняет самодиагностику после включения питания. Медленное мигание означает, что ЦПУ конфигурирует модули ввода-вывода. Одновременное мигание этого светодиода и зеленого светодиода Run указывает, что ЦПУ находится в режиме загрузки и ожидает обновления фирменного программного обеспечения через Порт 1.

RUN Зеленый, если ЦПУ в режиме Run. Желтый, если ЦПУ в режиме Stop/IO Scan. Если этот светодиод не светится, а светодиод OK светится, ЦПУ находится в режиме Stop/No IO Scan.

Если этот светодиод мигает зеленым, а светодиод Fault светится, переключатель модуля был переведен из положения Stop в положение Run, хотя имелась фатальная ошибка. ЦПУ находится в режиме RUN.

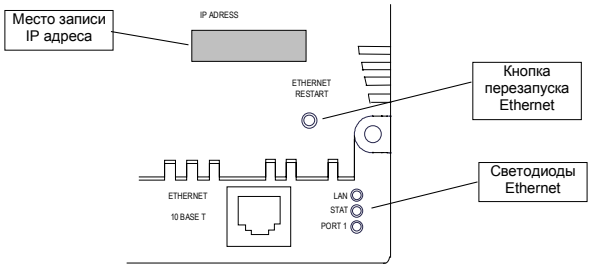
IC200CPUE05: ЦПУ с двумя последовательными портами, встроенным интерфейсом Ethernet и 64K конфигурируемой памяти

FAULT	Светится, если ЦПУ находится в режиме Stop/Faulted вследствие фатальной ошибки. Чтобы светодиод Fault погас, нужно очистить и таблицу ошибок ввода-вывода (I/O Fault Table) и таблицу ошибок ПЛК (PLC Fault Table). Если этот светодиод мигает и светодиод OK не горит, фатальная ошибка была обнаружена во время выполнения диагностики ПЛК при включении питания. Свяжитесь с отделом обслуживания ПЛК.
FORCE	Светится, если битовая ячейка принудительно установлена.
PORT 1	Мигание указывает на активность на этом порту.
PORT 2	

ЦПУ с двумя последовательными портами, встроенным интерфейсом Ethernet и 64K конфигурируемой памяти: IC200CPUE05

Кнопка перезапуска (Restart) Ethernet

Кнопка перезапуска Ethernet находится с правой стороны модуля.



Кнопка перезапуска Ethernet выполняет две функции:

- При нажатии менее чем на 5 секунд, она сбрасывает оборудование Ethernet, тестирует светодиоды Ethernet, и перезапускает программное обеспечение Ethernet. Это прерывает все соединения Ethernet, установленные в настоящий момент.
- При нажатии на 5 секунд и более, она переключает режим работы порта 1 между сконфигурированным режимом и режимом работы с локальным монитором станции. Заметьте, что если Порт 1 используется с локальным монитором станции, Winloader не может быть использован для обновления фирменного программного обеспечения.

Светодиоды Ethernet

Три светодиода Ethernet указывают состояние и активность интерфейса Ethernet.

LAN Указывает состояние и активность соединения по сети Ethernet. Светящийся/мерцающий зеленый означает, что интерфейс Ethernet в режиме online. Светящийся желтый означает, что интерфейс Ethernet в режиме offline

STAT Указывает состояние интерфейса Ethernet. Светящийся зеленый означает, что коллизий не обнаружено. Светящийся желтый указывает на коллизию. Мигающий желтый указывает на ошибку кода. Мигающий зеленый указывает на ожидание конфигурации или IP адреса.

PORT1 Означает, что интерфейс Ethernet управляет последовательным портом RS-232. Также означает, что кнопка перезапуска Ethernet была использована для переключения порта RS-232 в режим работы с локальным монитором станции. Светящийся желтый означает, что Порт 1 может использоваться с локальным монитором станции. Не светится, если Портом 1 управляет ЦПУ ПЛК. (не мигает во время обмена).

IC200CPUE05: ЦПУ с двумя последовательными портами, встроенным интерфейсом Ethernet и 64K конфигурируемой памяти

Светодиоды Ethernet кратковременно загораются, сначала желтым светом, затем зеленым, всякий раз, когда перезапуск выполняется в рабочем состоянии путем нажатия и отпускания кнопки Restart. Это позволяет проверить работу светодиодов Ethernet. Все три светодиода мигают зеленым светом одновременно во время загрузки программного обеспечения.

Конфигурируемая память

CPUE05 имеет 64Кбайт конфигурируемой пользовательской памяти. Эта память используется для прикладной программы, конфигурации оборудования, регистров (%R), аналоговых входов (%AI), и аналоговых выходов (%AQ). Количество памяти, отведенное прикладной программе и конфигурации оборудования, автоматически определяется действующей программой и конфигурацией, введенной из программатора. Оставшаяся часть 64Кбайт может быть легко сконфигурирована так, чтобы удовлетворять приложению.

Конфигурируемая память	64Кбайт максимум
Размер прикладной программы (не конфигурируется)	128 байт минимум
Размер конфигурации оборудования (не конфигурируется)	528 байт минимум
Регистры (%R)	256 байт минимум
Аналоговые входы (%AI)	256 байт минимум
Аналоговые выходы (%AQ)	256 байт минимум

ЦПУ с двумя последовательными портами, встроенным интерфейсом Ethernet и 64K конфигурируемой памяти: IC200CPUE05

Обзор интерфейса Ethernet

CPUE05 имеет встроенный интерфейс Ethernet, позволяющий связываться по сети 10BaseT. Поддерживаются полнодуплексный и полудуплексный режимы. Использование коммутаторов 10/100 позволяет подключать CPUE05 к сети, содержащей 100Мб устройства.

Сервер SRTP

CPUE05 поддерживает до восьми одновременных подключений к серверу SRTP других устройств сети Ethernet, таких как программатор ПЛК, CIMPLICITY HMI, каналы SRTP для ПЛК Series 90, и приложения коммуникации с хостом. Для работы сервера не требуется программирование ПЛК.

Ethernet Global Data

CPUE05 поддерживает до 32 одновременных обменов Ethernet Global Data. Обмены Global Data конфигурируются с помощью инструментального программного обеспечения ПЛК, затем они сохраняются в ПЛК. Могут быть сконфигурированы как входящие, так и исходящие обмены. CPUE05 поддерживает до 1200 переменных для всех обменов Ethernet Global Data, а также поддерживает избирательный прием Ethernet Global Data. Информация о Ethernet Global Data приведена в главе 13.

Функциональные возможности монитора станции

CPUE05 имеет встроенный монитор станции. Это позволяет проводить диагностику on-line и обеспечивает супервизорный доступ либо через порт монитора станции, либо через сеть Ethernet. Службы монитора станции включают в себя:

- Интерактивный набор команд для опроса станции и управления ей.
- Неограниченный доступ к внутренней статистике, журналу регистрации и параметрам конфигурации.
- Защита паролем для команд, которые изменяют параметры станции или ее работу.

Для использования функций монитора станции требуется отдельный компьютерный терминал или эмулятор терминала.

Информация о работе монитора станции приведена в документе GFK-1876.

Глава 4 Установка

Эта глава описывает:

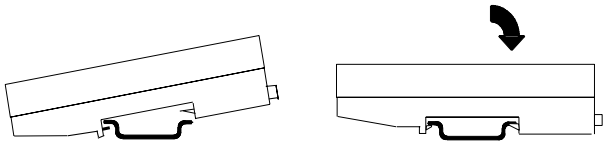
- Установку ЦПУ
- Установку блока питания
- Установку дополнительных модулей
- Активизацию или замену батареи
- Подключение последовательного порта
- Установку модулей расширения
- Подключение CPUE05 к сети Ethernet
- Требования к установке CE Mark

Инструкции по установке системы, описывающие основные принципы установки шасси, источника питания и модулей, а также информация о подключении и заземлении приведены в документе *Модули, источники питания и шасси VersaMax. Руководство пользователя (VersaMax Modules, Power Supplies, and Carriers Manual)*, GFK-1504.

Инструкции по установке

Все модули и шасси VersaMax® в одном крейте ПЛК должны быть установлены на одной секции DIN-рейки 7.5мм x 35мм, толщиной 1мм. Рекомендуется использовать стальную DIN-рейку. DIN-рейка должна быть заземлена для обеспечения электромагнитной защиты. Рейка должна иметь проводящую (неокрашенную) поверхность, защищенную от коррозии. Предпочтительны DIN-рейки, соответствующие стандарту DIN EN50022. Для обеспечения виброустойчивости DIN-рейка должна быть закреплена на панели с помощью винтов, находящихся на расстоянии приблизительно 15.24см (6 дюймов) друг от друга.

Шасси легко фиксируется на DIN-рейке. Для установки или заземления на рейку инструменты не требуются.

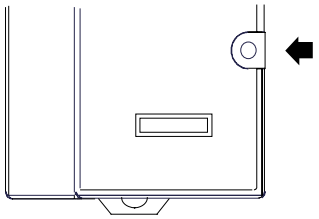


Демонтаж ЦПУ с DIN-рейки

1. Отключите питание от блока питания.
2. (Если ЦПУ прикреплено к панели винтом) снимите модуль источника питания. Вывинтите крепежный винт.
3. Сдвигайте ЦПУ по DIN-рейке от других модулей пока разъемы не выйдут из зацепления.
4. С помощью маленькой плоской отвертки оттяните защелку на дне модуля вниз от модуля и снимите модуль с DIN-рейки.

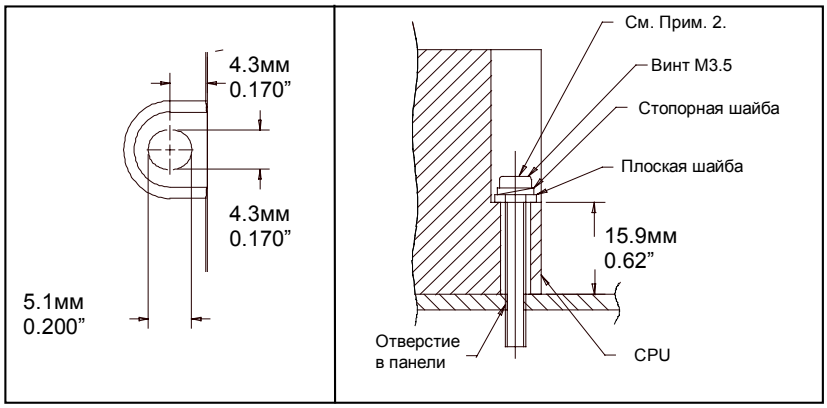
Установка на панель

Для максимальной устойчивости к механической вибрации и ударам оборудование должно быть закреплено на панели. Используя модуль как шаблон, отметьте на панели место для крепежного отверстия. Просверлите отверстие в панели. Установите модуль, закрепив его винтом M3.5 (#6).



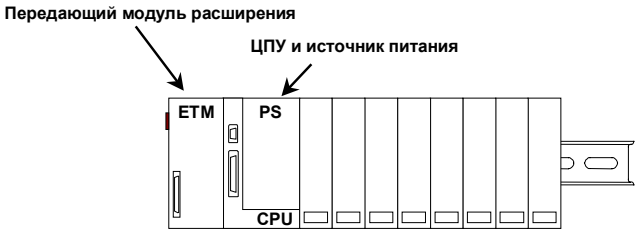
Примечание 1. Допуски на все размеры составляют +/- 0.13мм +/- 0.005дюйма.

Примечание 2. Для крепления используются стальные винты M3.5 (усилие затяжки от 1.1 до 1.4Нм) и панель толщиной не менее 2.4 мм с резьбовыми отверстиями.



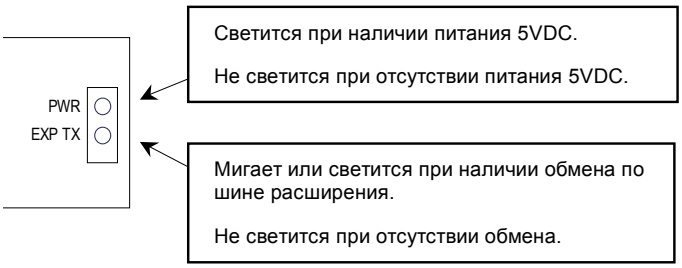
Установка передающего модуля расширения (ETM)

Если ПЛК VersaMax имеет более одного крейта расширения или один крейт расширения, использующий изолированный принимающий модуль расширения (IC200ERM001) как интерфейс шины расширения, передающий модуль расширения должен быть установлен слева от ЦПУ. Передающий модуль расширения должен быть установлен на той же секции DIN-рейки, что и остальные модули главного крейта (крейт 0).



Главный крейт (0) ПЛК VersaMax

1. Убедитесь, что питание крейта выключено.
2. Установите передающий модуль расширения на DIN-рейку слева от ЦПУ.
3. Установите ЦПУ. Соедините модули и сожмите их так, чтобы разъемы соединились.
4. После завершения любых дополнительных действий по установке системы подайте питание и проверьте состояние светодиодов модуля.



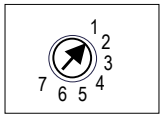
Демонтаж передающего модуля расширения

1. Убедитесь, что питание крейта выключено.
2. Сдвиньте модуль по DIN-рейке от ЦПУ в главном крейте.
3. С помощью маленькой отвертки оттяните защелку на дне модуля вниз от модуля и снимите модуль с DIN-рейки.

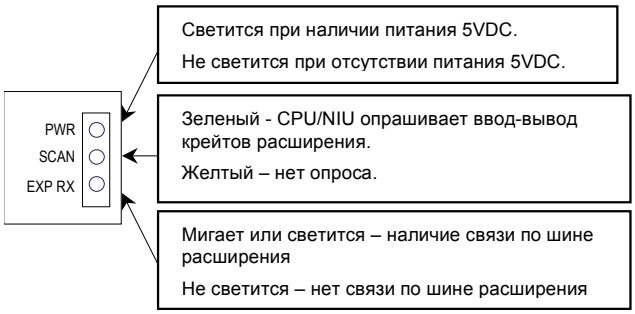
Установка принимающего модуля расширения (ERM)

Принимающий модуль расширения (IC200ERM001 или 002) должен быть установлен в крайний левый слот каждого крейта расширения VersaMax.

1. Установите этикетку под дверку в правом верхнем углу модуля.
2. Установите модуль на DIN-рейку с левой стороны крейта расширения.
3. Выберите ID (от 1 до 7) крейта расширения с помощью поворотного переключателя под дверкой в левом верхнем углу модуля. Для каждого крейта должен быть установлен свой ID. При единичном подключении (только один крейт расширения), установите ID в 1.



4. Установите модуль источника питания VersaMax сверху принимающего модуля расширения. Подробности приведены в этой главе в разделе “Установка источника питания”.
5. Присоедините кабели. Если система включает в себя передающий модуль расширения, подключите блок согласующих резисторов (терминатор) к порту EXP2 последнего принимающего модуля расширения.
6. После завершения любых дополнительных действий по установке системы подайте питание и проверьте состояние светодиодов модуля.



Снятие принимающего модуля расширения

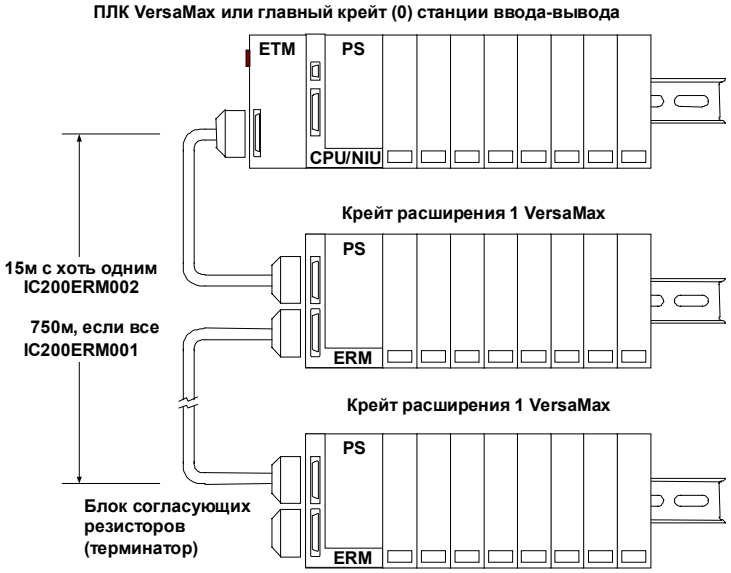
1. Убедитесь, что питание крейта выключено.
2. Снимите модуль блока питания с принимающего модуля расширения.
3. Сдвиньте принимающий модуль расширения по DIN-рейке от других модулей.
4. С помощью маленькой отвертки оттяните защелку на дне модуля вниз от модуля и снимите модуль с DIN-рейки.

Блоки питания крейта расширения

Питание для работы модулей поступает от блока питания, установленного на принимающем модуле расширения. Если крейт расширения включает дополнительное шасси блока питания и дополнительный блок питания крейта, он должен быть подключен к тому же источнику напряжения, что и блок питания на принимающем модуле расширения.

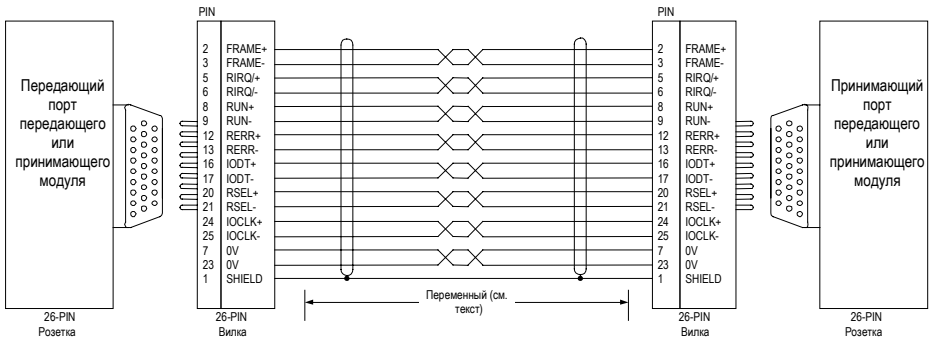
Подключение кабеля расширения: дифференциальный RS-485

Для систем расширения с несколькими крейтами, подключите кабель к порту расширения передающего модуля и к принимающему модулю, как показано ниже. Если все используемые принимающие модули изолированного типа (IC200ERM001), максимальная длина кабеля составляет 750 метров. Если шина расширения включает хоть один не изолированный принимающий модуль (IC200ERM002), максимальная длина кабеля составляет 15 метров.



Установите блок согласующих резисторов (поставляется с передающим модулем расширения) в нижний порт последнего принимающего модуля. Блок согласующих резисторов может быть приобретен отдельно (номер по каталогу IC200ACC201, кол-во – 2 шт.).

Дифференциальные RS-485 межкрейтовые соединительные кабели (IC200CBL601, 602, 615)

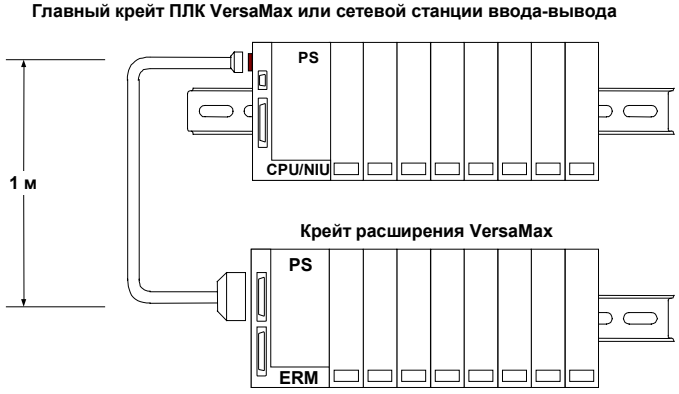


Изготовление пользовательского кабеля расширения

Пользовательские кабели расширения могут быть изготовлены с использованием комплекта разъемов IC200ACC202, обжимного инструмента AMP 90800-1 и кабеля Belden 8138, Manhattan/CDT M2483, Alpha 3498C, или эквивалентного кабеля AWG #28 (0.089mm^2).

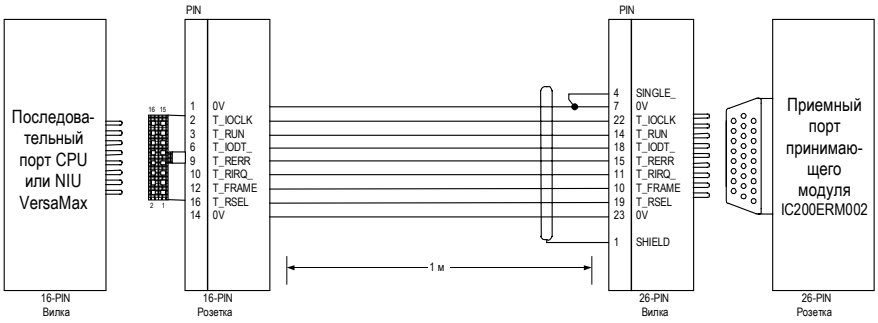
Подключение кабеля расширения: Единичное подключение

Для систем, не включающих в себя передающего модуля расширения, и состоящих из одного не изолированного крейта расширения (IC200ERM002), соедините кабелем расширения последовательный порт ЦПУ VersaMax и принимающий модуль, как показано ниже. Максимальная длина кабеля – 1 метр. Для соединения данного типа кабель не может быть изготовлен самостоятельно, следует отдельно заказать кабель IC200CBL600.



При единичном подключении блок согласующих резисторов (терминатор) не требуется; однако, в случае его установки, он не помешает работе системы.

Межкрейтовое соединение при единичном подключении (IC200CBL600)



Блоки питания для систем расширения с единичным подключением

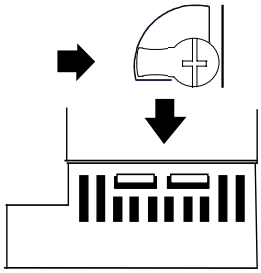
Для работы системы в режиме единичного подключения, блоки питания главного крейта и крейта расширения должны быть подключены к одному источнику напряжения. Главный крейт и крейт расширения не могут быть включены или выключены по отдельности; для надлежащей работы они должны быть или оба включены или оба выключены.

Питание для работы модулей в крейте расширения поступает от источника питания, установленного на принимающем модуле расширения. Если крейт расширения включает дополнительное шасси блока питания и дополнительный блок питания крейта, он должен быть подключен к тому же источнику напряжения, что и блок питания на принимающем модуле расширения.

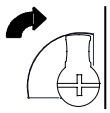
Установка модулей блоков питания

Модули блоков питания устанавливаются прямо на модуль ЦПУ, принимающие модули расширения, и дополнительные шасси блока питания.

Блок питания, установленный на ЦПУ или принимающем модуле расширения, обеспечивает модулям питание +5В и +3.3В. Допустимое количество модулей зависит от потребления питания модулями. Возможно использование дополнительных блоков питания для обеспечения питанием всех модулей. Если крейт включает в себя шасси дополнительного блока питания и дополнительный блок питания, он должен быть подключен к тому же источнику напряжения, что и блок питания, установленный на ЦПУ. Конфигурационное программное обеспечение позволяет подсчитать требуемое питание в зависимости от конфигурации оборудования. Инструкции по установке блока питания приведены ниже.



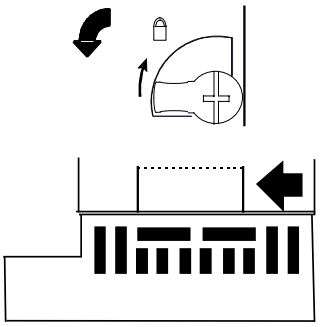
1. Фиксатор на блоке питания должен быть открыт.
2. Выровняйте разъемы и фиксатор и нажмите на модуль блока питания так, чтобы защелкнулись два выступа на нижней части источника питания. Убедитесь, что выступы полностью вставлены в отверстия нижней кромки ЦПУ, принимающего модуля расширения (ERM), или шасси.



3. Поверните фиксатор, чтобы зафиксировать блок питания.

Снятие блока питания

Будьте осторожны при работе с включенным оборудованием. Устройства могут сильно нагреться и причинить ущерб здоровью.



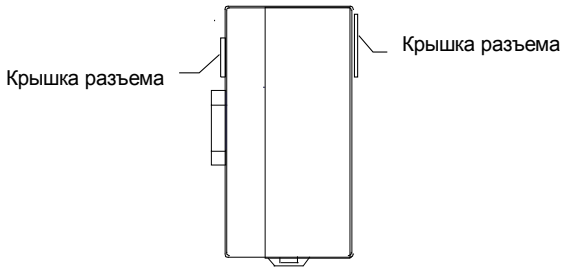
1. Выключите питание.
2. Поверните фиксатор, как показано на рисунке.
3. Нажмите на гибкую панель нижней части источника питания, чтобы выступы источника питания вышли из отверстий шасси.
4. Снимите источник питания.

Установка дополнительных модулей

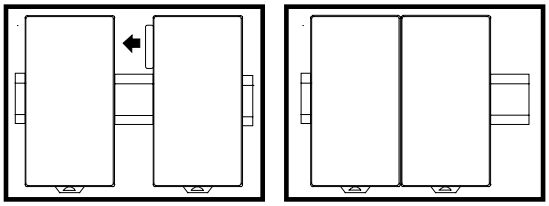
ЦПУ или принимающий модуль расширения может обслуживать до 8 модулей ввода-вывода и вспомогательных модулей, установленных на той же секции DIN-рейки. Перед добавлением шасси в крейт питание должно быть выключено.

Перед подключением шасси к ЦПУ или ERM, снимите с правой стороны ЦПУ/ERM крышку, закрывающую разъем. Не выбрасывайте эту крышку, она понадобится для установки на последнее шасси. Она защищает контакты разъема от повреждения и электростатических разрядов при установке и работе.

Не снимайте крышку разъема с левой стороны.



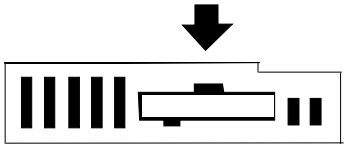
Установите шасси рядом с предыдущим шасси, затем прижмите шасси друг к другу так, чтобы разъемы соединились. Для предупреждения повреждения контактов разъема избегайте слишком сильного нажима и ударов.



С обеих сторон станции на DIN-рейку должны быть установлены зажимы (номер по каталогу IC200ACC313) для фиксации положения модулей.

Активация или замена батареи

Модуль ЦПУ поставляется с уже установленной батареей. Батарейный отсек установлен на верхней стороне модуля ЦПУ. Перед началом использования активируйте батарею, потянув и удалив изолятор.



Замена литиевой батареи

Чтобы заменить батарею, аккуратно откройте батарейный отсек с помощью маленькой отвертки.

Допускается использование только следующих батарей:

GE Fanuc	IC200ACC001
Panasonic	BR2032

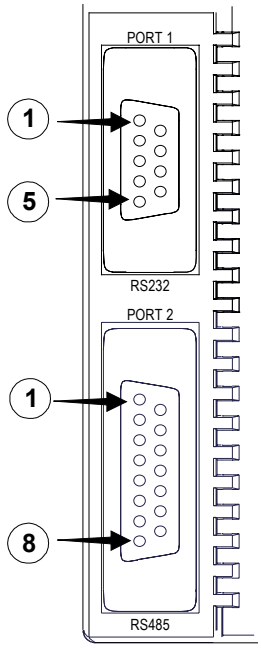
Использование другой батареи может вызвать пожар или взрыв.

Предостережение

Батарея может взорваться при неправильном обращении.

Запрещается заряжать, разбирать, нагревать выше 100°C (212°F) и сжигать.

Подключение последовательного порта



Подача питания на внешнее устройство через Порт 2

Если порт предназначен для последовательной связи с устройством, которому требуется питание 100мА или меньше при напряжении 5В постоянного тока, то это устройство может получать питание от Порт 2.

Длина кабеля и скорость

Максимальная длина кабеля (от ЦПУ до последнего устройства, подключенного к кабелю) составляет:

Порт 1 (RS-232) = 15 метров (50 футов)
Порт 2 (RS-485) = 1200 метров (4000 футов)

Оба порта конфигурируются на различные скорости, как указано в описаниях ЦПУ в этом руководстве.

Поставляются следующие готовые кабели:

IC200CBL001	RS232 кабель для программирования ЦПУ
IC200CBL002	Кабель для обновления системного ПО в модулях расширения

Порт 1: RS-232

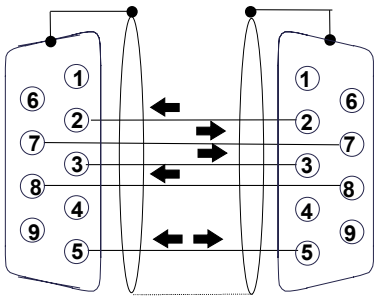
Назначение контактов разъема Порт 1

Порт 1 - RS-232 порт с 9-контактной розеткой типа D-sub. Он используется как загрузочный порт для обновления системного программного обеспечения ЦПУ. Его цоколевка позволяет использовать простой прямой кабель для соединения со стандартным портом RS-232 типа AT. Экран кабеля присоединяется к корпусу.

Контакт	Сигнал	Направление	Функция
1	-		
2	TXD	Выход	Выход передачи данных
3	RXD	Вход	Вход приема данных
4	-		
5	GND	--	0V/общий провод
6	-		
7	CTS	Вход	Вход сброса передатчика
8	RTS	Выход	Выход запроса передатчика
9	-		
Корпус	SHLD	--	Подключение экрана кабеля / 100% экранирование кабеля

Соединение RS-232 точка-точка

В конфигурации точка-точка, два устройства подключаются к одной коммуникационной линии. Для RS-232 максимальная длина - 15 метров (50 футов).



PC 9-Pin Serial Port	ЦПУ Port 1
9-pin розетка	9-pin вилка
(2) RXD	(2) TXD
(3) TXD	(3) RXD
(5) GND	(5) GND
(7) RTS	(7) CTS
(8) CTS	(8) RTS

Экран должен быть подключен к корпусам разъемов с обеих сторон кабеля.

Спецификации разъема и кабеля для Порт 1

Изготовитель комплектующих указан только для примера. Могут быть использованы любые комплектующие, удовлетворяющие спецификации.

Кабель: Belden 9610	Компьютерный кабель 5 жил с общим экраном † 30 Вольт / 80°C (176°F) 24 AWG медные проводники, многожильный 7x32			
Разъем (вилка на 9 контактов):	Тип: Crimp (под обжим)	Изготовитель: ITT/Cannon AMP	Штепсель: DEA9PK87F0 205204-1	Контакт: 030-2487-017 66506-9
	Solder (под пайку)	ITT/Cannon AMP	ZDE9P 747904-2	-- --
Корпус разъема:	Комплект *– ITT Cannon DE121073-54: Металлизированный пластик (Пластик с покрытием из меди и никеля) † Зажим заземления кабеля (включен) Выход кабеля под углом 40° обеспечивает низкопрофильную установку Плюс – ITT Cannon 250-8501-010 [Дополнительные винты]: Предназначены для крепления разъема к порту CPU001 † Заказывайте 2 шт. для каждого заказанного корпуса			

† Критичная информация – другие выбранные комплектующие должны соответствовать этим критериям или превышать их.

* Использование этого комплекта обеспечивает глубину в сборе 70 мм.

Порт 2: RS-485

Назначение контактов разъема Порт 2

Порт 2 - RS-485 порт с 15-контактной розеткой типа D-sub. Он может быть подключен напрямую к адаптеру RS-485/RS-232.

Контакт	Сигнал	Направление	Функция
1	SHLD	--	Подключение экрана кабеля
2, 3, 4	Не исп.		
5	P5V	Выход	+5.1VDC для питания внешних устройств (100мА макс.)
6	RTSA	Выход	Выход запроса передатчика (A)
7	GND	--	0V/общий провод
8	CTSB’	Вход	Вход сброса передатчика (B)
9	RT	--	Согласующий резистор (120 ом) для RDA
10	RDA’	Вход	Вход приема данных (A)
11	RDB’	Вход	Вход приема данных (B)
12	SDA	Выход	Выход передачи данных (A)
13	SDB	Выход	Выход передачи данных (B)
14	RTSB	Выход	Выход запроса передатчика (B)
15	CTSA’	Вход	Вход сброса передатчика (A)
Корпус	SHLD	--	Подключение экрана кабеля / 100% экранирование кабеля

Спецификации разъема и кабеля для Порт 2

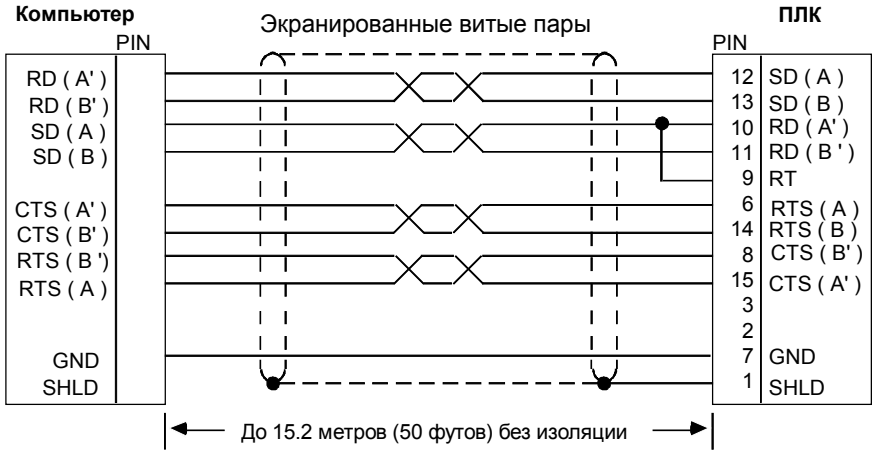
Изготовитель комплектующих указан только для примера. Могут быть использованы любые комплектующие, удовлетворяющие спецификации.

<table>
<tr><td>Кабель:
Belden 8105</td><td colspan="4">Компьютерный кабель маленькой емкости 5 витых пар с общим экраном †
Провод подключения экрана †
30 Вольт / 80°C (176°F)
24 AWG медные проводники, скрутка 7x32
Скорость распространения = 78%
Номинальный импеданс = 100Ω †</td></tr>
<tr><td rowspan="2">Разъем
(вилка на 15 контактов):</td><td>Тип:
Crimp
(под обжим)</td><td>Изготовитель:I
TT/Cannon
AMP</td><td>Штепсель:
DAA15PK87F0
205206-1</td><td>Контакт:
030-2487-017
66506-9</td></tr>
<tr><td>Solder
(под пайку)</td><td>ITT/Cannon
AMP</td><td>ZDA15P
747908-2</td><td>--
--</td></tr>
<tr><td>Корпус разъема:</td><td colspan="4">Комплект * – ITT Cannon DA121073-50:
Металлизированный пластик (Пластик с покрытием из Cu и Ni) †
Зажим заземления кабеля (включен)
Выход кабеля под углом 40° обеспечивает низкопрофильную установку
Плюс – ITT Cannon 250-8501-009 [Дополнительные винты]:
Предназначены для крепления разъема к порту CPU001 †
Заказывайте 2 шт. для каждого заказанного корпуса</td></tr>
</table>

† Критичная информация – другие выбранные комплектующие должны соответствовать этим критериям или превышать их.

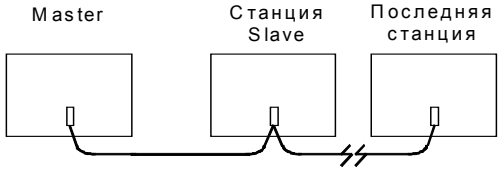
RS-485 подключение точка-точка с квитированием

В конфигурации точка-точка, два устройства подключаются к одной коммуникационной линии. Для RS-485 максимальная длина кабеля - 1200 метров (4000 футов). Для увеличения расстояния можно использовать модемы.

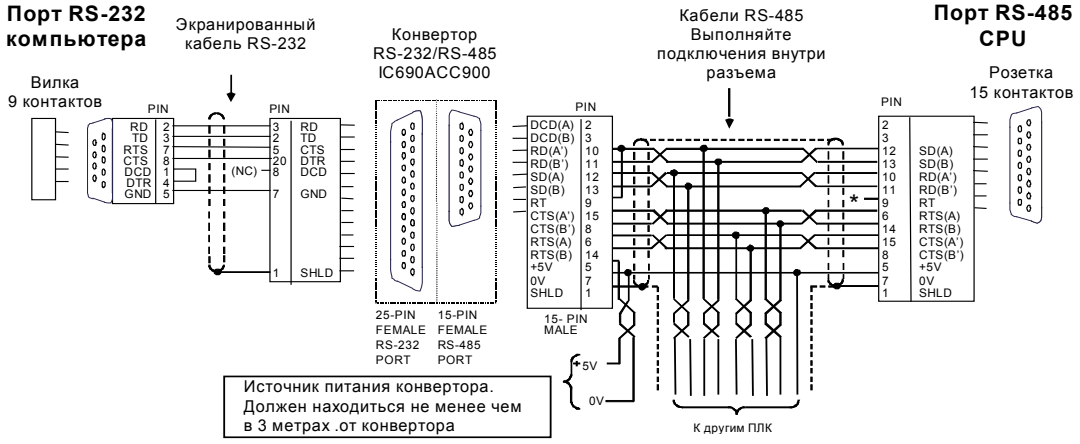


RS-485 многоточечные последовательные подключения

При многоточечной конфигурации, основное устройство конфигурируется как мастер, а один или более ПЛК конфигурируются как слэйв. Максимальное расстояние между мастер и любым слэйв -устройством не может превышать 4000 футов (1200 метров). Для такой конфигурации требуются кабели хорошего качества и умеренно шумные условия. До 8 слэйв-устройств может быть подключено с использованием RS-485 при шлейфовой или многоточечной конфигурации. Линия RS-485 должна включать квитирование и использовать тип провода, указанный ранее.



При подключении RS-485 многоточечными кабелями, отраженные волны в передающей линии могут быть уменьшены при использовании подключения шлейфом, как показано ниже. Подключения выполняются внутри разъема, подсоединяемого к ПЛК. Избегайте использования клеммных колодок по всей длине линии связи.



Согласующий резистор для сигнала Прием Данных (RD) должен быть подключен только на устройствах, находящихся на концах линий. На ЦПУ это выполняется установкой перемычки между контактами 9 и 10 разъема.

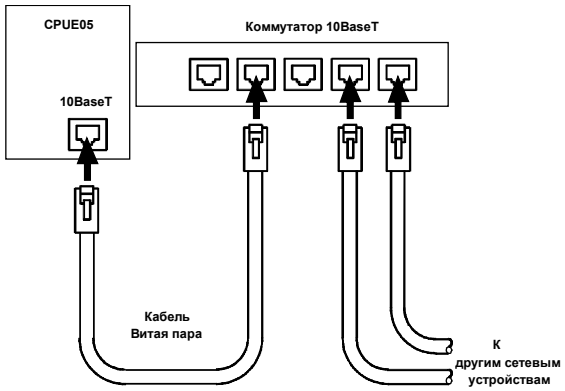
Заземление: Устройства, не подключенные к одному и тому же источнику питания, должны иметь общее заземление или быть изолированы от заземления для обеспечения надлежащей работы системы.

Ethernet подключение CPUE05

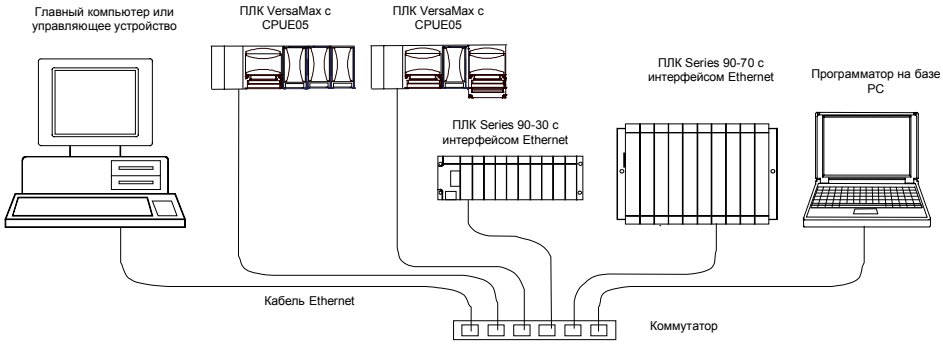
Ethernet порт модуля IC200CPUE05 подключается к сети 10BaseT (витая пара) без внешнего трансивера. Подключите порт к внешнему коммутатору или репитеру 10BaseT с автоматическим переключением 10/100, используя кабель типа “витая пара”. Кабели можно приобрести у соответствующих дистрибьюторов. GE Fanuc рекомендует кабели не изготавливать, а приобретать. Ваши кабели 10BaseT должны соответствовать стандартам IEEE 802.

Сетевое подключение

Подключение CPUE05 к сети 10BaseT показано ниже:



Длина кабеля между каждым узлом и коммутатором или репитером не должна превышать 100 метров. Обычно коммутаторы и репитеры поддерживают от 4 до 12 узлов, подключенных по топологии “звезда”.



4

Требования к установке CE Mark

Следующие требования по защите от бросков напряжения, электростатического разряда и импульсных помех должны быть выполнены для приложений, требующих CE Mark:

- ПЛК VersaMax требует установки в шкафу (не ниже IP54).
- Это оборудование предназначено для использования в типичных промышленных условиях, с применением антистатических материалов, таких как бетон или деревянный настил. Если оборудование используется в условиях с применением статических материалов, таких как ковры, персонал, перед работой с оборудованием, должен снять с себя статический заряд, дотронувшись до надежно заземленной поверхности.
- Если для питания ввода-вывода используется переменный ток, на линиях должны быть установлены фильтры, ограничивающие помехи до допустимого уровня. Фильтрация линий питания переменного тока может быть осуществлена с помощью варисторов, подключаемых между линиями или между линией и заземлением. Для варисторов линия-заземление должно быть выполнено хорошее высокочастотное заземление.
- Источники питания переменного и постоянного тока с напряжением менее 50В предназначены для местного преобразования питания от основной линии питания переменного тока. Длина проводников между этими источниками питания и ПЛК должна быть более 10 метров.
- Контроллер должен быть установлен в помещении. В помещении должна быть предусмотрена защита от бросков напряжения на входящих линиях питания переменного тока.
- При повышенных шумах связь по последовательному протоколу может прерываться.

Глава
5

Конфигурирование ЦПУ

В этой главе описан процесс конфигурирования ЦПУ VersaMax® и обслуживаемых им модулей. Конфигурирование определяет параметры работы модуля, а также устанавливает адреса, используемые каждым модулем системы.

- Автоконфигурирование или конфигурирование с помощью программатора.
- Конфигурирование крейтов и слотов.
- Конфигурирование параметров ЦПУ.
- Конфигурирование распределения памяти ЦПУ.
- Конфигурирование параметров последовательного порта.
- Сохранение конфигурации с помощью программатора.
- Автоконфигурирование.

5

Использование автоконфигурирования или конфигурирование с помощью программатора

ПЛК VersaMax можно сконфигурировать автоматически или с помощью программатора, используя программный конфигуратор. Оба способа конфигурирования описаны в этой главе.

АВТОКОНФИГУРИРОВАНИЕ

Автоконфигурирование происходит при включении питания. Модули, имеющие функцию программного конфигурирования, после автоконфигурирования могут использовать только настройки по умолчанию.

ПРОГРАММНЫЙ КОНФИГУРАТОР

Большинство систем ПЛК использует пользовательскую конфигурацию, которая создается с помощью программного конфигуратора и сохраняется в ЦПУ с помощью программатора.

ЦПУ сораняет конфигурацию при включении и выключении. После того как конфигурация сохранена в ЦПУ, ЦПУ не будет автоконфигурироваться при последующих включениях.

Программный конфигуратор может быть использован для:

- Создания новых конфигураций
- Сохранения (записи) конфигурации в ЦПУ
- Загрузки (считывания) существующей конфигурации из ЦПУ
- Сравнения конфигурации в ЦПУ с конфигурацией, сохраненной в программаторе
- Удаления конфигурации, ранее сохраненной в ЦПУ

ЦПУ сохраняет конфигурацию в своей энергонезависимом ОЗУ. Сохранение конфигурации отменяет автоконфигурацию, таким образом, ПЛК не перезапишет конфигурацию при последующих включениях.

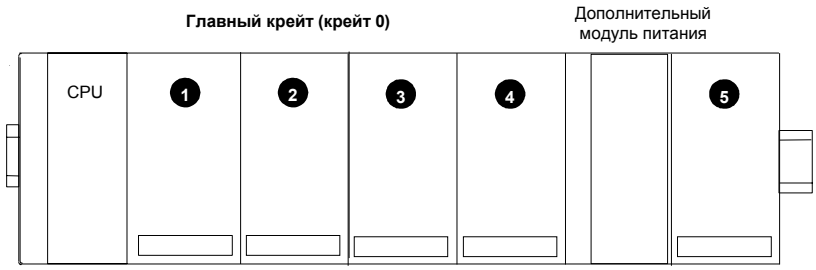
Удаление конфигурации из программатора повлечет новое автоконфигурирование. В этом случае автоконфигурирование будет производиться до тех пор, пока конфигурация из программатора не будет заново сохранена.

Один из параметров, который может управляться программным конфигуратором, это возможность ЦПУ при включении считывать конфигурацию и программу из флэш памяти или из ОЗУ. Если выбрана флэш память, то ЦПУ при включении будет считывать ранее сохраненную

конфигурацию из флэш памяти. Если выбрано ОЗУ, то ЦПУ при включении будет считывать конфигурацию и прикладную программу из ОЗУ.

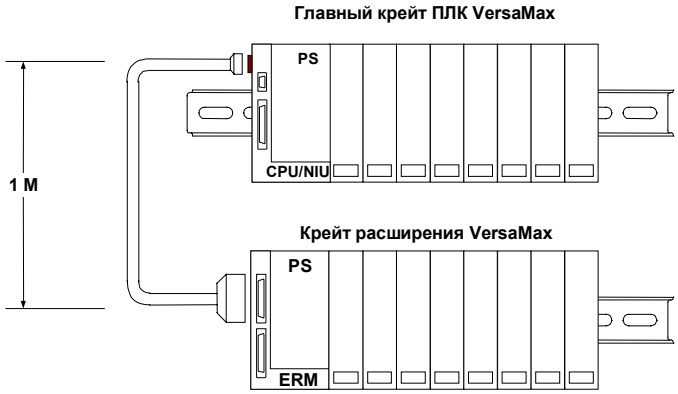
Конфигурирование крейтов и слотов

Не смотря на то, что ПЛК VersaMax не имеет крейта как такового, автоконфигурирование и программное конфигурирование используют традиционную терминологию крейтов и слотов для определения положения модуля в системе. Каждый логический крейт состоит из ЦПУ или принимающего модуля расширения плюс до восьми дополнительных модулей ввода-вывода и вспомогательных модулей установленных на одну DIN-рейку. Каждый модуль ввода-вывода или вспомогательный модуль занимает один слот. Модуль, следующий после ЦПУ, или принимающий модуль расширения устанавливается в слот 1. Дополнительные источники питания не учитываются при подсчете слотов.

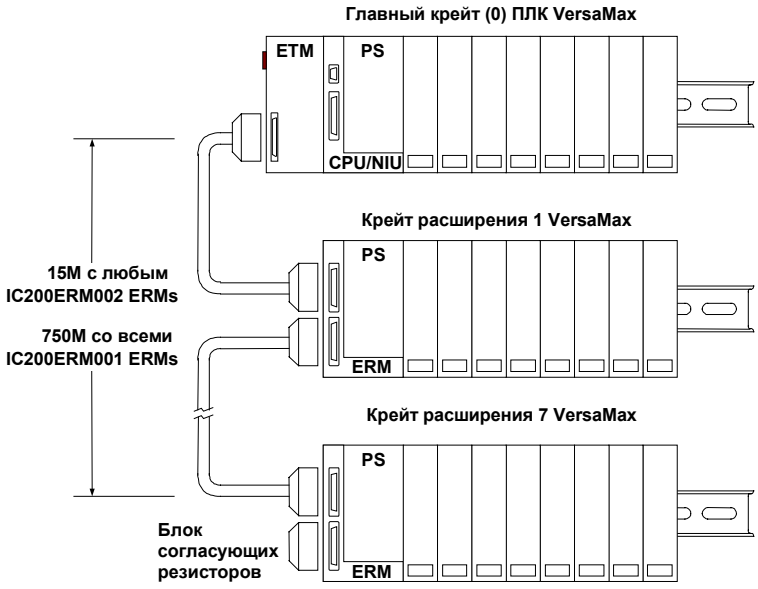


Главный крейт - крейт 0. Дополнительные крейты пронумерованы от 1 до 7.

В системе, где используется только один крейт расширения, который присоединен к шине расширения с помощью не изолированного принимающего модуля расширения (IC200ERM002), крейт расширения должен быть сконфигурирован как крейт 1.



В системе, где есть передающий модуль расширения (IC200BTM001) и до семи крейтов расширения, каждый с заизолированным принимающим модулем расширения (IC200ERM001или IC200ERM002), дополнительные крейты конфигурируются как крейты от 1 до 7.



5

Программный конфигуратор

Программный конфигуратор позволяет создавать пользовательскую конфигурацию для систем ПЛК VersaMax. При использовании CPUE05, он также используется для конфигурирования Ethernet Global Data.

Когда вы входите в каталог оборудования VersaMax конфигуратора, экран по умолчанию- Крейт (Главный). Новая конфигурация уже включает по умолчанию источники питания (PWR001) и ЦПУ (CPU001). Они оба могут быть легко изменены в соответствии с имеющимся оборудованием ПЛК.

Для конфигурирования ПЛК следует:

- Сконфигурировать тип крейта (не расширяемый, расширяемый с единичным подключением, или расширяемый с мультикрейтовым подключением).
- Сконфигурировать тип источника питания и любого дополнительного источника питания, и шасси. (Примечание: CPU005 и CPUE05 требуют источника питания с повышенной нагрузочной способностью по напряжению 3.3В .)
- Сконфигурировать ЦПУ. Эта операция включает изменение типа ЦПУ, если требуется, и определение его параметров, как описано в этой главе.
- Сконфигурировать параметры последовательных портов ЦПУ, как описано в этой главе.
- Для CPUE05, сконфигурировать параметры Ethernet, как описано в главе 6.
- Сконфигурировать модули расширения, если в системе есть крейты расширения.
- Добавить шасси модулей и определить внешние подключения.
- Разместить модули на шасси и выбрать их параметры. Конфигурируемые параметры модулей ввода-вывода описаны в *Модули, источники питания и шасси VersaMax. Руководство пользователя (VersaMax Modules, Power Supplies, and Carriers User's Manual)* (GFK-1504).
- Сохранить конфигурационный файл, чтобы он мог быть записан в ПЛК .

Пошаговая инструкция по использованию програмного конфигуратора описана в *Программное обеспечение VersaPro. Руководство пользователя (VersaPro Software User's Manual)* (GFK-1670). Дополнительную информацию можно получить в службе поддержки.

КОНФИГУРИРОВАНИЕ ЦПУ И ЕГО ПАРАМЕТРОВ

Приведенная ниже таблица описывает конфигурируемые параметры ЦПУ ПЛК VersaMax и крейтов расширения.

Параметр	Описание	Значение по умолчанию	Возможные значения
Параметры Цикла			
Sweep Mode	Normal: цикл выполняется до завершения. Constant: цикл выполняется в течение времени указанного в Sweep Tmr.	Normal	Normal, Constant Sweep
Sweep Times (mSecs)	Время цикла в режиме constant (в миллисекундах).	100мс	5–200мс
Настройки			
I/O Scan-Stop	Определяет, будет ли быть модуль ввода-вывода опрашиваться, когда ПЛК находится в режиме STOP.	No	Yes, No
Powerup Mode	Режим работы при включении.	Last	Last, Stop, Run
Logic/Configuration From	Источник программы и конфигурации при включении ПЛК.	RAM	RAM, флэш
Registers	Выбор источника данных регистров при включении ПЛК.	RAM	RAM, флэш
Passwords	Определяет включены или выключены пароли. (Если пароли отключены, единственная возможность их включения- это очистка памяти ПЛК.)	включен	Enabled, Disabled
Checksum Words per Sweep	Количество слов контрольной суммы прикладной программе, подсчитываемое при каждом цикле.	8	8 to 32
Default Modem Turnaround Time	Время переключения модема (10ms/ед.). Это время нужное модему, для начала передачи информации после получения запроса.	0мс	0–255мс
Default Idle Time	Время (в секундах), в течение которого ЦПУ ожидает следующее сообщение от программирующего устройства до того, как оно установит, что программирующее устройство неисправно, и вернется к исходному состоянию. Связь с программатором разрывается и должна быть заново восстановлена.	10	1 – 60
SFC Timer Faults	Включает или выключает просмотр ошибок SFC таймера.	Disabled	Enabled/ Disabled
SNP ID		None	Editable
Switch Run/Stop	Определяет возможность переключателя управлять работой режимами работы Run/Stop.	включен	Enabled, Disabled
Switch Memory Protect	Определяет возможность переключателя управлять защитой ОЗУ.	Disabled	Enabled, Disabled
Diagnostics	Если нужное вам приложение не нуждается в более быстром запуске, оставьте эту функцию включенной. Если эта настройка выключена, то ПЛК запускается	включен	Enabled, Disabled

	без диагностики.		
Fatal Fault Override	Определяет, будут ли фатальные ошибки принудительно установлены или сброшены.	Disabled	Enabled, Disabled
EZ Program Store	Уточняет, куда будет загружена информация, считанная с устройства хранения программ EZ.	RAM only	RAM only, RAM & Flash

Конфигурирование распределения памяти ЦПУ

CPU001 и CPU002 (версия 2.0 или более поздняя), CPU005 и CPUE05 имеют конфигурируемую пользовательскую память. Конфигурируется распределение общего объема памяти между прикладной программой, конфигурацией оборудования, регистрами (%R), аналоговыми входами (%AI) и аналоговыми выходами (%AQ) . Объем памяти, отведенной прикладной программе и конфигурации оборудования, автоматически определяется программой и конфигурацией, введенными с программатора.

Остальную память можно легко распределить в соответствии с приложением. Например, приложение может содержать относительно большую программу, которая использует небольшой объем памяти. Одновременно может быть маленькая программа, но больший объем памяти требуется для регистров и аналоговых входов и выходов.

Конфигурируемая Память ЦПУ модуля IC200CPU001, CPU002, CPU005

Конфигурируемая память	CPU001: 34Кбайт максимум. CPU002: 42Кбайт максимум. CPU005: 64Кбайт максимум
Размер прикладной программы (не конфигурируемый) CPU001, для совместимости с версией 1.50 CPU002, для совместимости с версией 1.50	128 байт минимум 12K байт 20K байт
Размер конфигурации оборудования (не конфигурируемый)	400 байт минимум
Регистры (%R) CPU001/002, для совместимости с версией 1.50	256 байт (128 слов) минимум 4,096 байт (2048 слов) минимум
Аналоговые Входы (%AI)	256 байт (128 слов) минимум
Аналоговые Выходы (%AQ)	256 байт (128 слов) минимум

Конфигурируемая Память ЦПУ модуля IC200CPUE05

Конфигурируемая память	64Кбайт максимум
Размер прикладной программы (не конфигурируемый)	128 байт минимум
Размер конфигурации оборудования (не конфигурируемый)	528 байт минимум
Регистры (%R)	256 байт (128 слов) минимум
Аналоговые Входы (%AI)	256 байт (128 слов) минимум
Аналоговые Выходы (%AQ)	256 байт (128 слов) минимум

Если переконфигурировать распределение памяти от значений по умолчанию, последующее сохранение конфигурации оборудования в ПЛК сотрет содержимое памяти. Если вы хотите сохранить содержимое памяти, сначала загрузите содержимое памяти из ПЛК в программатор. А потом сохраните заново после того, как конфигурация оборудования будет сохранена из программатора в памяти ПЛК.

Конфигурирование параметров последовательного порта.

Оба порта на ЦПУ ПЛК VersaMax конфигурируются для работы в качестве SNP слэйв или RTU слэйв. Поддерживаются 4-проводный и 2-проводный интерфейсы RTU . Только для CPUE05 порт 1 может быть сконфигурирован (на другой закладке) для работы локального монитора станции. Параметры локального монитора станции могут отличаться от параметров Порта А.

Название	Описание	Значение по умолчанию	Возможные значения
Port Mode	Определяет протокол.	SNP	SNP, Serial I/O, RTU, Disabled. Для CPUE05 может быть также сконфигурирован в качестве локального монитора станции.
Parity	Определяет способ проверки паритета.	Odd. Для CPUE05, когда порт в режиме локального монитора станции, настройка по умолчанию None.	Odd, Even, None
Data Rate (bps)	Скорость обмена данными (Бит в секунду).	Связь по последовательному порту: 19200	SNP 4800, 9600, 19200, 38400
			RTU: 1200, 2400, 4800, 9600, 19200, 38400, 57600
			Serial I/O: 4800, 9600, 19200, 38400, 57600
		CPUE05 в режиме локального монитора станции : 9600	Режим локального монитора станции: 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200
Flow Control (Не требуется если порт в режиме SNP)	Определяет метод использования контроля связи. При переключении контроля связи с"None" на “Hardware”, Turnaround Delay сбрасывается на 0.	None	Режим RTU: None, Hardware
			Serial I/O mode: None, Hardware, Software
			CPUE05 в режиме локальной монитора станции: None, Hardware
Timeout (Если порт в режиме SNP)	Определяет набор значений перерывов связи для использования протоколом.	Long	Long, Medium, Short, None
Stop Bits (Если порт в режиме SNP или последоват ельный порт ввода-вывода)	Число стоповых битов, испольуемое при передаче.. (Большинство устройств подключаемых последовательно используют 1стоповый бит; более медленые устройства используют 2.)	1	1, 2
SNP ID	8 битовый ID для порта 1.	None	Редактируемые
Receive to transmit delay	Задержка между получением последнего символа сообщения до установки сигнала RTS	0	SNP: Не используются RTU и Serial IO: 0-255 (в десятках мс, т. е. 10=100мс)
Turnaround delay	Задержка между установкой сигнала RTS и передачей сообщения.	SNP: none RTU & Serial IO: 0	SNP: Long, Medium, Short, none RTU & Serial IO: 0-255 (units of 10ms, e.g. 10=100ms)
RTS drop delay	Задержка между отправкой последнего символа сообщения и сбросом сигнала RTS.	0	SNP Не используются RTU и Serial IO: 0-255 (в десятках мс, т. е. 10=100мс)

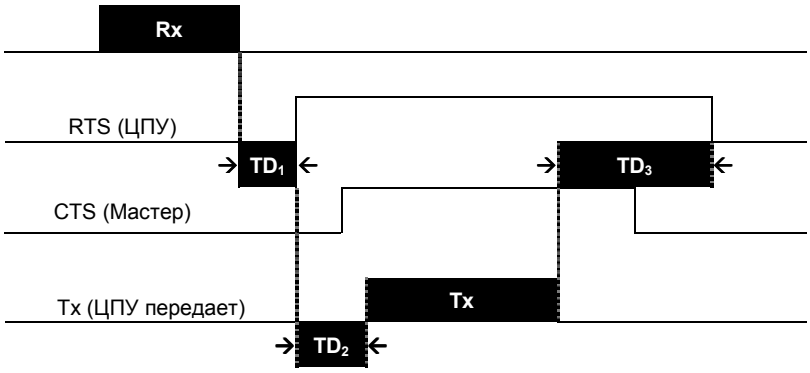
Программное обеспечение VersaPro позволяет сконфигурировать RTU и Serial I/O на скорость 115.2Кбод. Однако, ЦПУ не поддерживает эту скорость. Если конфигурация, использующая эту скорость, сохранена в ПЛК, то:

1. Для протокола RTU регистрируется ошибка “Unsupported Feature in Configuration” и ПЛК переходит в режим Stop Faulted.
2. Для протокола Serial I/O регистрируется та же самая ошибка при переходе в режим Run. ПЛК немедленно перейдет в режим Stop Faulted.

Задержки RTU и протокола Serial IO

Параметры задержек "между приемом и передачей", "переключения", и "задержка сброса сигнала RTS" могут быть сконфигурированы для синхронизации работы радиомодемов.

- **receive to transmit delay (задержка между приемом и передачей):** Минимальный промежуток времени между тем, как ЦПУ примет последний символ входящего сообщения и будет установлен сигнал RTS. Эта задержка сконфигурирована на "минимум" т.к. действительная задержка зависит от длительности цикла ЦПУ.
- **turnaround delay (задержка переключения):** Промежуток времени между установкой сигнала ЦПУ с RTS и началом передачи сообщения ЦПУ.
- **RTS drop delay (задержка сброса сигнала RTS):** Промежуток времени между тем, как ЦПУ передаст последний символ исходящего сообщения и сбросит сигнал RTS. Задержка сброса сигнала RTS может различаться на ± 1 ms.



- TD_1 **задержка между приемом и передачей**
- TD_2 **задержка переключения**
- TD_3 **задержка сброса сигнала RTS**

Конфигурация, необходимая для использования Winloader

Утилита Winloader, которая может быть использована для обновления системного программного обеспечения, требует конфигурирования SNP. Если порт 1 сконфигурирован для другого режима или для работы в режиме локального монитора станции, Winloader не сможет произвести обновление системного программного обеспечения через порт 1.

Примечание для соединения через RTU

Когда используется соединение через RTU, возможно придется увеличить перерыв RTU, сконфигурированный в мастер-устройстве т.к. длительность опроса слэйв-ПЛК увеличивается. Однако, само ЦПУ VersaMax в конфигурировании не нуждается.

Сохранение конфигурации из программатора

Обычно ПЛК VersaMax конфигурируются следующим образом: в программаторе (компьютере) создается файл конфигурации, затем файл предается из программатора в ЦПУ ПЛК через порт ЦПУ . ЦПУ сохраняет файл конфигурации в энергонезависимом ОЗУ. Конфигурация сохраняется независимо от того, включен ли опрос В/В. После сохранения конфигурации, опрос В/В включается либо выключается в соответствии с параметрами новой конфигурации.

Автоконфигурирование и Сохранение Конфигурации

Удаление конфигурации с помощью программатора повлечет новое автоконфигурирование. Автоконфигурирование остается включенным до тех пор, пока конфигурация не будет сохранена из программатора. Сохранение конфигурации выключает автоконфигурирование.

Сохранение Конфигурации с Нестандартным Распределением Памяти

Если вы переконфигурируете исходные параметры, сохранение конфигурации оборудования в ПЛК в будущем повлечет удаление содержимого памяти. Если вы хотите сохранить содержимое памяти, сначала загрузите исходное содержимое памяти из ПЛК в программатор. А потом сохраните заново после того, как конфигурация оборудования будет сохранена из программатора в память ПЛК.

Параметры Последовательного Порта по Умолчанию

Когда программатор подключается впервые, связь с ПЛК происходит в соответствии с параметрами по умолчанию: 19,200 бод, четный паритет, один стартовый бит, один стоповый бит, и 8 информационных бит. Если эти

параметры изменены, то новые параметры вступят в силу после включения питания.

Конфигурация Последовательного Порта Вступает в Силу после Отсоединения Программатора

Если конфигурация оборудования сохранена в ЦПУ, конфигурация последовательного порта, к которому подключен программатор, не вступает в силу до отключения программатора. После отключения программатора новый протокол вступит в силу после задержки. Эта задержка равна сконфигурированному времени T3’.

Автоконфигурирование

Когда автоконфигурация включена и предыдущая автоконфигурация не существует, при включении ЦПУ автоматически считывает конфигурации модулей, установленных в системе, и создает суммарную конфигурацию системы. Если предшествующая автоконфигурация существовала на момент включения, процесс конфигурирования производится, как описано на следующей странице.

Модули с возможностью программного конфигурирования, при автоконфигурировании используют настройки по умолчанию. Эти возможности описаны в *Модули, источники питания и шасси. Руководство пользователя VersaMax (VersaMax Modules, Power Supplies, and Carriers Manual)* (GFK-1504).

При включении ЦПУ по умолчанию создает автоконфигурацию, которая охватывает все модули установленные в системе, начиная со слота 1 крейта 0 (основной крейт). Автоконфигурирование крейта останавливается на первом пустом слоте или неисправном модуле, и продолжается со следующего крейта. Пример: если модули установлены в слоты 1,2,3,5 и 6,то автоконфигурирование не затрагивает модули в слотах 5 и 6 .

Чтобы произвести автоконфигурированиее с крейтами расширения, или все крейты должны питаться от одного источника питания, или крейт расширения должен быть включен раньше основного.

Назначение адресов при автоконфигурировании

Модулям автоматически присваиваются адреса в порядке возрастания. Например, если система включает модуль ввода на 16 точек, модуль ввода на 8 точек, модуль вывода на 16 точек и еще модуль ввода на 16 точек, в этом порядке, модулям ввода будут присвоены адреса %I0001, %I0017, и %I0025 соответственно. Для модулей, использующих данные нескольких типов (например, комбинированных модулей ввода-вывода), для данных каждого типа адрес присваивается отдельно.

Диагностика при Автоконфигурировании

Модуль установлен, но не работает во время автоконфигурирования: Если модуль установлен, но не работает во время автоконфигурирования, модуль не конфигурируется и ЦПУ выводит диагностику *extra module (Дополнительный модуль)*.

Пустой слот при автоконфигурировании: Автоконфигурирование крейта останавливается на первом пустом слоте. Модули, установленные после пустого слота, автоматически не конфигурируются. ЦПУ дает диагностику дополнительного модуля для каждого из них.

Ранеее сконфигурированые модули присутствуют при автоконфигурировании: Ранее сконфигурированые модули не удаляются из конфигурации при автоконфигурировании, если они в системе отсутствуют. Пример: Если модули сконфигурированы в слотах 1,2 и 3, а затем было отключено питание и модуль из слота 1 удален, при включении питания модули в слотах 2 и 3 автоконфигурируются в обычном режиме. Сам модуль в слоте 1 не удаляется из конфигурации. *ЦПУ дает диагностику потеря модуля (loss of module) для слота 1*.

При автоконфигурировании обнаружен другой модуль: *Если слот был сконфигурирован под один тип модуля, но при автоконфигурировании был установлен другой модуль, ЦПУ выводит диагностику несовпадение конфигурации (Configuration Mismatch)*. Конфигурация слота не изменяется.

Неконфигурированный модуль установлен после автоконфигурирования: *Если ранее не конфигурировавшийся модуль установлен после включения питания, ЦПУ выводит диагностику дополнительного модуля, и модуль в конфигурацию не добавляется*.

Ранее конфигурированный модуль установлен после автоконфигурирования: *Если ранее сконфигурированый модуль отсутствовал при включении и был установлен после включения, ЦПУ выводит диагностику добавление модуля (Module Addition)* и модуль будет заново добавлен в цикл ввода-вывода.

Все модули удалены после автоконфигурирования: Если все модули отсутствуют при включении, ЦПУ удаляет конфигурацию. Это позволяет при последующем включении устанавливать и добавлять модули в конфигурацию.

Краткий Обзор Диагностических Сообщений

Добавление Модуля	Несконфигурированный модуль присутствует при включении. Модуль добавлен в конфигурацию. Автоконфигурирование включено и модуль может быть сконфигурирован.
Добавление Модуля	Ранее сконфигурированный модуль установлен после включения. ЦПУ возобновляет опрос модуля.
Несовпадение Конфигурации	Модуль, не соответствующий конфигурации слота, был обнаружен при включении или после него.
Добавлен Модуль, конфигурация невозможна	1. Не сконфигурированный модуль присутствует при включении. 2. Автоконфигурирование не включено. 3. Не конфигурированный модуль установлен после включения
Потеря Модуля	Сконфигурированный модуль отсутствует при включении или во время работы.
Добавление Крейта	1. Во время конфигурирования обнаружен ранее не сконфигурированный принимающий модуль расширения 2. Во время работы связь с отсутствовавшим или неисправным принимающим модулем расширения восстановлена. ЦПУ начинает опрос модулей ввода-вывода в крейтах. Ошибки “Добавление модуля” не выводятся, когда опрос возобновляется. Однако, если связь не может быть восстановлена с каким-либо из модулей в крейте, выводится ошибка “Потеря Модуля”.
Потеря Крейта	1. Ранее сконфигурированый принимающий модуль расширения не найден при конфигурировании. 2. При нормальной работе работавший ранее принимающий модуль расширения перестает работать. Модули, установленные в этот крейт расширения, не опрашиваются.
Добавлен Крейт, конфигурация невозможна	Ранее не сконфигурированный принимающий модуль расширения установлен после включения. Модули, установленные в крейт расширения, игнорируются.
Несовпадение Передатчика Расширения	1. Передающий модуль расширения (IC200ETM001) установлен, но не сконфигурирован. 2. Передающий модуль расширения (IC200ETM001) сконфигурирован, но не установлен.
Изменение Скорости Шины Расширения	Скорость шины расширения, подсчитываемая ЦПУ автоматически, при автоконфигурировании изменилась.
Неподдержи-ваемое оборудование	Установленный модуль не поддерживается ЦПУ.

Глава
6

Конфигурация Ethernet

В этой главе описана конфигурация Ethernet интерфейса ЦПУ VersaMax® модуля IC200CPUE05:

- Обзор Конфигурации Ethernet
- Конфигурирование параметров Ethernet интерфейса.
- Конфигурирование Ethernet Global Data
- Конфигурирование Дополнительных Пользовательских Параметров

Конфигурирование интерфейса Ethernet, описанное в этой главе, производится совместно с конфигурированием основных параметров, описанных в главе 5.

6

Обзор Конфигурации Ethernet

Конфигурирование Ethernet для модуля ЦПУ IC200CPUE05 включает:

- Конфигурирование параметров Ethernet интерфейса. Является частью конфигурации ЦПУ.
- Конфигурирование Ethernet Global Data. Выполняется через конфигурацию “работа крейта”.
- (Дополнительно. Не требуется в большинстве систем). Конфигурирование дополнительных параметров. Требует создания отдельного файла конфигурации ASCII, который сохраняется в ПЛК вместе с конфигурацией оборудования.
- (Дополнительно. Не требуется в большинстве систем). Настройку порта 1 для работы в режиме локального монитора станции. Является частью основной конфигурации ЦПУ описанной в главе 5. Примечание: параметры локального монитора станции конфигурируются отдельно от параметров порта 1.

После того, как конфигурирование закончено, и конфигурация сохранена в ПЛК, она хранится в памяти ЦПУ ПЛК. Конфигурация может быть сохранена во флэш памяти и загружена из нее, что практически всегда сохраняет резервную копию данных конфигурации, даже при отключении основного и резервного питания. Каждый раз при включении CPUE05, изменении или удалении его конфигурации, данные конфигурации Ethernet загружаются обратно в интерфейс Ethernet.

Интерфейс Ethernet CPUE05 сохраняет данные конфигурации в памяти с поддержкой от батарейки. Если поддержка батарейки не исправна и конфигурация не была сохранена во флэш памяти, то сохраненные параметры конфигурации интерфейса Ethernet будут утрачены. Если это происходит, то после включения вступают в силу заводские настойки интерфейса Ethernet по умолчанию, действительные до нового конфигурирования. Переход к настройкам по умолчанию включает возвращение к IP адресу 0.0.0.0. Так как резервные данные конфигурации Ethernet сохраняются интерфейсом Ethernet CPUE05, они не могут быть удалены из ПЛК процедурой очистки конфигурации (Clear Configuration). Когда конфигурация ПЛК удалена, ЦПУ работает в режиме Автоконфигурации, как описано ниже.

Автоконфигурация

Если конфигурация из программатора не сохранена в ЦПУ ПЛК, то при включении автоматически создается новая конфигурация. Чтобы создать новую автоконфигурацию, ЦПУ считывает данные конфигурации из каждого модуля и из интерфейса Ethernet. Эта процедура также затрагивает файл дополнительных настроек интерфейса Ethernet.

Когда автоконфигурация находится в ЦПУ ПЛК, возможно редактирование некоторых параметров конфигурации Ethernet с монитора станции. Это изменяет параметры, хранящиеся в самом интерфейсе Ethernet. При включении-выключении ПЛК или удалении конфигурации, отредактированная конфигурация будет загружена в ЦПУ из интерфейса Ethernet.

Конфигурирование интерфейса Ethernet.

Основные параметры работы Ethernet, должны быть правильно сконфигурированы для корректной работы сети Ethernet. *Конфигурация по умолчанию не может обеспечить корректный сетевой адрес.*

Параметры	Описание
Configuration Mode (Режим конфигурации)	Всегда TCP/IP. Не может быть изменен.
IP Address, Subnet Mask, and Gateway IP Address (IP адрес, Маска подсети, и IP адрес шлюза)	IP адрес является уникальным адресом интерфейса Ethernet как узла сети. В большой сети маска подсети может быть использована для идентификации секции всей сети. Адрес шлюза может быть использован для идентификации шлюза, соединяющего одну сеть с другой. Эти параметры должны быть правильно настроены, иначе интерфейс Ethernet не сможет работать в сети, и/или работа сети может быть нарушена. Особенно важно чтобы каждому узлу сети был назначен IP address. Эти значения должны назначаться человеком, управляющим сетью (Администратор сети). Администраторы сетей TCP/IP знакомы с этими параметрами. Если у вас нет сетевого администратора и вы используете простую изолированную сеть без шлюзов*, вы можете использовать следующие значения в качестве IP адресов*: 10.0.0.2 Первый ПЛК 10.0.0.3 Второй ПЛК 10.0.0.4 Третий ПЛК . . 10.0.0.254 программатор или хост Также, в этом случае, установите IP адреса маски подсети и шлюза на 0.0.0.0. Подробности о назначении IP адресов и шлюзах в главе 13. Примечание**:** Если эта простая замкнутая сеть когда-либо будет подключена к другой сети, IP адреса с 10.0.0.2 по 10.0.0.254 не должны использоваться и маска подсети и IP адреса шлюза должны назначаться сетевым администратором. IP адреса должны назначаться в соответствии с подключенной сетью**.**
Status Address (Адрес Статуса)	Начальный адрес блока данных статуса Ethernet длиной 10 байт. Содержимое этих данных описано в главе 13, “Проверка состояния интерфейса Ethernet.” Адрес статуса может быть назначен в %I, %Q, %R, %AI или %AQ памяти. Значение по умолчанию следующий доступный %I адрес. Примечание: Не используйте 10 байт, назначенных для битов Статуса, для других целей, иначе ваша информация будет перезаписана.
Status Length (Длина Статуса)	Это значение автоматически устанавливается 80 бит (для %I и %Q местонахождений адреса статуса) или 5 слов (для %R, %AI, и %AQ местонахождений адреса статуса).
Network Time Servers **(Сервер сетевого времени)**	IP адреса до трех NTP серверов времени используемых для синхронизации отметок времени в производимых обменах Ethernet Global Data. Если нет сконфигурированных NTP серверов, интерфейс Ethernet инициализируется в соответствии с таймером ЦПУ. Подробности в разделе “Метки времени обменов Ethernet Global Data” главы 13.

Конфигурирование Ethernet Global Data

ЦПУ IC200CPUE05 VersaMax может быть сконфигурировано для выполнения до 32 обменов Ethernet Global Data (любое сочетание входящих или исходящих обменов). (Описание этого параметра в разделе “Ethernet Global Data” главы 13). Конфигурация определяет: содержание обмена, диапазоны данных и характеристики работы. Каждый обмен Ethernet Global Data входящими и исходящими данными должен быть сконфигурирован отдельно для каждого ПЛК.

Вы можете сконфигурировать:

- До 1200 диапазонов данных для всех обменов Ethernet Global Data для CPUE05.
- До 100 диапазонов данных для одного обмена.
- Длину данных для одного обмена от 1 байт до 1400 байт. Общий объем обмена, равен сумме длин всех диапазонов данных, сконфигурированных, для данного обмена.

Различные обмены могут содержать различные диапазоны данных. Несколько обменов могут совместно использовать одинаковые диапазоны данных полностью или частично, даже если они происходят на разных скоростях. (Примечание: Инструментальное программное обеспечение не позволит совместно использовать диапазоны данных при их получении).

Меню конфигурации Ethernet Global Data находится в меню конфигурации крейта (но не ЦПУ).

Перед конфигурированием обменов EGD

Перед конфигурированием обменов Ethernet Global Data, вам необходимо собрать информацию о ПЛК, которые будут производить обмены данных. *Примечание: эта информация будет нужна для конфигурации каждого ПЛК*. Подробности в главе 13.

- Для каждого ПЛК определить, какие данные нужно отправить и принять.
- Создать список IP адресов для Ethernet интерфейсов ПЛК, которые используются при обменах.
- Определить состав до 32 групп устройств, которые будут совместно использовать Ethernet Global Data.
- Выбрать для обменов подходящие периодичности повторов и длительности задержек.
- Определить содержание каждого обмена в отправителе, и определить в получателях соответствующие диапазоны данных для приема.

- Каждому получателю не обязательно принимать все исходящие данные отправителя. Принимаемый обмен может быть сконфигурирован так, чтобы игнорировать заданные диапазоны данных.

Конфигурирование обмена Global Data для Отправителя

Каждый обмен Global Data должен быть сконфигурирован, в отправителе как описано ниже. Обмен также должен быть сконфигурирован в каждом получателе.

Параметры	Описание
Local Producer ID (ID локального отправителя)	Адрес, который однозначно идентифицирует CPUE05 как устройство Ethernet Global Data в сети. Это десятичные числа, разделенные точками. По умолчанию он совпадает с IP адресом CPUE05. Этот адрес может быть изменен.
Exchange ID (ID обмена)	Число, которое идентифицирует определенный обмен данными.
Adapter Name (Название адаптера)	Для CPUE05 всегда 0.0.
Consumer Type (Тип Получателя)	Выбрать будут ли данные направлены на одно устройство (IP адрес) или одну из 32 ранее определенных групп оборудования (ID группы). Подробности в разделе “Группы Ethernet Global Data” главы 13.
Consumer Address (Адрес Получателя)	Если "Тип получателя" IP адрес, то это IP адрес одного устройства для обмена. Если "Тип получателя" ID группы, то это ID группы номер (1–32). Подробно об IP адресах в главе 13.
Send Type (Тип отправки)	В данный момент доступна только настройка ”always (всегда)". Ethernet Global Data будет отправляться всегда при включенном сканировании каналов B/B . При выключенном сканировании каналов B/B данные отправляться не будут.
Producer Period (Время Отправки)	Установленный промежуток времени для отправки данных по сети. Диапазон 10–3,600,000 миллисекунд (от 10 миллисекунд до 1 часа). По умолчанию промежуток равен 200 миллисекундам. Перед вводом округляйте это значение до ближайшего десятка. Время отправки имеет разрешение в 10 миллисекунд. Если вы введете такое значение как 12 миллисекунд, то реальная периодичность отправки будет округлена до 20 миллисекунд. Для облегчения наладки и эффективного использования сети установите одинаковое значение времени отправки и приема. Не отправляйте данные быстрее, чем требует ваше приложение. Например, обычно бесполезно устанавливать периодичность отправки меньше, чем время цикла отправляющего или принимающего ПЛК. Такой подход снижает нагрузку на сеть и устройства, освобождая ресурсы для других коммуникаций.
Reply Rate (Скорость Ответа)	В данный момент не используется.
Status Word (Слово статуса)	Место в памяти, где будет храниться значение статуса, отправленных данных. Подробности в разделе “Проверка статуса обмена” главы 13. Примечание: адрес слова статуса должен быть уникальным; ему не назначается автоматически следующий свободный адрес.

Пример:	Смещение	Тип памяти	Нижняя точка	Верхняя точка	Описание
	Статус:	%R	99	99	Статус: Куда ПЛК поместит данные статуса.
Exchange Data Ranges (Диапазон данных обмена)	Список диапазонов данных от 1 до 100, которые будут отправлены при обмене. Данные отправляются как непрерывный блок байтов. Подробности в разделе “Проверка статуса обмена” главы 13. Общий объем может достигать 1400 байт. Список диапазона данных для отправки при обмене имеет следующую структуру:				
Пример:	Смещение	Тип памяти	Нижняя точка	Верхняя точка	Описание
	0.0	%R	100	105	Конвейер 1 в ПЛК1
	10.0	%I	345	352	Концевой выключатель конвейера 1 в ПЛК1

Конфигурирование обмена Global Data для получателя

Для получения Global Data, сконфигурируйте следующее:

Параметры	Описание
Local Producer ID (ID локального отправителя)	Адрес, который однозначно идентифицирует CPUE05 как устройство Ethernet Global Data в сети. По умолчанию он совпадает с IP адресом CPUE05. Этот адрес может быть изменен.
Exchange ID (ID обмена)	Число, которое идентифицирует определенный обмен данных. Оно должно соответствовать ID исходящего обмена (в отправляющем устройстве).
Adapter Name (Название адаптера)	Для CPUE05 всегда 0.0.
Producer ID (ID отправителя)	ID локального отправителя, устройства отправляющего данные.
Group ID (ID группы)	Используется, если одни и те же данные получает более чем одно устройство. Введите тот же ID группы, что был сконфигурирован как "Адрес получателя" в отправителе.
Consumer Period (Время получения)	Не используется. По умолчанию 200 мс.
Update Timeout (Задержки обновления)	Максимальный промежуток времени, который интерфейс Ethernet допускает между обнаружением посылок в сети без обновления статуса ошибки. Статус ошибки означает, что первый или последующий пакет данных не прибыл в указанное время. Диапазон 0, или 10–3,600,000 миллисекунд. Это значение должно быть хотя бы в 2 раза больше периода отправления отправителя. Значение по умолчанию 0, что отключает обнаружение задержки. Задержка между обновлениями должен быть больше чем время отправки данных. (Рекомендуется устанавливать значение, которое хотя бы в 2 раза превышает время отправки) Перед вводом округляйте это значение до ближайшего десятка. Перерыв между обновлениями имеет шаг в 10 миллисекунд. Если вы введете 22 миллисекунды, то перерыв между обновлениями округлится до 30 миллисекунд
Status Word (Слово статуса)	Место в памяти, где будет храниться значение статуса, принятых данных. Подробности о значении статуса в главе 13. Примечание: адрес слова статуса должен быть уникальным; ему не назначается автоматически следующий свободный адрес.

Пример:	Смещение	Тип памяти	Нижняя точка	Верхняя точка	Описание
	Статус:	%R	99	99	Статус: Куда ПЛК поместит данные статуса.

Определение обмена Global Data для получателя (продолжение)

<table>
<tr><th>Параметры</th><th>Описание</th></tr>
<tr><td>Time Stamp (Программная отметка времени)</td><td>Блок данных, который определяет место в памяти, где будет сохранена отметка времени последнего пакета данных. Отметка времени не является действительной датой; это 8 байтное значение которое представляет собой время, прошедшее с 00:00, 1 января, 1970. Первые четыре байта содержат целое число со знаком, секунды, а вторые 4 байта содержат целое число, наносекунды. Это значение представляет собой время отправки данных по часам отправителя. Эту информацию можно использовать, чтобы узнать, принят ли по сети новый пакет, или эти данные были приняты ранее.

Отметка времени, проставляемая ПЛК, в настоящий момент имеет разрешение в 100 микросекунд, если не используется синхронизация сети. Если NTP выполняет синхронизацию времени в сети, отметка времени имеет разрешение в 1 миллисекунду, и погрешность синхронизации ПЛК в сети ±10 миллисекунд .

NTP может быть включено в конфигурации CPUE05. Как только NTP синхронизация времени сконфигурирована, CPUE05 будет синхронизироваться с внешними NTP серверами времени, если они есть.</td></tr>
<tr><td>Пример:</td><td>
<table>
<tr><th>Смещение</th><th>Тип памяти</th><th>Нижняя точка</th><th>Верхняя точка</th><th>Описание</th></tr>
<tr><td>Програм-мная отметка времени</td><td>%R</td><td>91</td><td>94</td><td>Отметка времени: Произвольное место, куда ПЛК ставит программную отметку времени.</td></tr>
</table>
</td></tr>
<tr><td>Exchange Data Ranges (Диапазон данных обмена)</td><td>Список от 1 до 100 диапазонов данных, которые будут получены при обмене. Данные принимаются как непрерывный блок байтов. Общий объем всех элементов может достигать 1400 байт. Не допускается использовать для приема данных память типа %S и ячейки с принудительно заданными значениями. Допустимые типы памяти приведены в таблице 4-2 .

Примечание: Если длина принятых данных не совпадает с длиной отправленных данных, происходит регистрация ошибки ПЛК и коллизии Ethernet.

Список диапазонов данных для приема при обменах имеет следующую структуру:</td></tr>
<tr><td>Пример:</td><td>
<table>
<tr><th>Смещение</th><th>Тип памяти</th><th>Нижняя точка</th><th>Верхняя точка</th><th>Описание</th></tr>
<tr><td>0.0</td><td>%R</td><td>100</td><td>104</td><td>Конвейер 1 в ПЛК1</td></tr>
<tr><td>10.0</td><td>%I</td><td>257</td><td>264</td><td>Концевой выключатель конвейера 1 в ПЛК1</td></tr>
</table>
</td></tr>
</table>

Выборочный прием

Не каждому ПЛК нужно получать все отправленные диапазоны данных. Например, отправитель оправляет данные, состоящие из числа с плавающей точкой (4 байта), за ним целое число (2 байта), затем аналоговое значение (2 байта). Если принимающему ПЛК нужно принять только аналоговое значение и поместить его в %AI003, получатель может быть сконфигурирован как описано ниже.

Смещение	Тип памяти	Нижняя точка	Верхняя точка	Описание
0	Игнорировать (Байты)	1	6	Игнорировать плавающую точку и целое число
6	%AI	3	3	

Примечание: общая длина обмена должна быть одинакова в отправителе и получателе, даже если получатель игнорирует байты в конце сообщения. Ошибка конфигурации игнорированных байтов при приеме приведет к тому, что данные не будут переданы, будут занесены аварийные сообщения в таблицу ошибок обмена, таблицу неисправностей ПЛК и статус обмена.

Конфигурирование Дополнительных Пользовательских Параметров

Дополнительные пользовательские параметры являются внутренними параметрами работы, используемыми Ethernet интерфейсом. *Для большинства приложений, не следует изменять Дополнительные пользовательские параметры, установленные по умолчанию*.

Если необходимо изменить какие-либо из этих параметров, то нужно создать файл Дополнительных пользовательских настроек, используя любой текстовый редактор, поддерживающий формат ASCII. Этот файл должен содержать названия и значения только тех параметров, которые вы изменяете. Файл должен быть назван “AUP_0_0.apf”. Завершенный файл должен быть помещен в ПЛК в папку, содержащую конфигурацию ПЛК. Когда вся конфигурация оборудования сохраняется из программатора в ПЛК, инструментальное программное обеспечение также сохраняет параметры из файла AUP_0_0.apf.

Формат Файла Дополнительных Пользовательских Параметров

Файл дополнительных пользовательских параметров должен иметь следующий формат:

```
AUP_0_0
<название параметра> = <значение параметра>
<название параметра> = <значение параметра>
<название параметра> = <значение параметра>
```

Все названия параметров пишутся строчными буквами. Знак равенства (=) должен ставиться между названием параметра и значением параметра.

Значения параметров должны быть переведены в нижний регистр, если они не будут заключены в двойные кавычки. Формат значений каждого параметра зависит от самого параметра. Числовые параметры вводятся в десятичном или шестнадцатиричном формате; шестнадцатеричные значения должны оканчиваться буквами ‘h’ или ‘H’. IP адрес должен вводиться как обычно в десятичном формате через точку. Значения буквенных параметров вводятся с учетом регистра. Параметры, введенные в верхнем регистре, должны быть заключены в двойные кавычки. (Все охватывающие кавычки не являются частью данных и при обработке удаляются).

Комментарии в файле должны начинаться с точки с запятой. Все символы в строке после точки с запятой игнорируются. Пустые строки также игнорируются.

В приведенном ниже примере устанавливается пароль монитора станции “system” и задается значение 4 для времени жизни сообщения при обмене Ethernet Global Data в конфигурации точка-точка.

Пример файла дополнительных пользовательских настроек

```
AUP_0_0
stpasswd = “system”      ;      установить пароль на "систему"
gucast_ttl=4             ;      установить время жизни сообщения в
                                режиме точка-точка 4
```

Описание дополнительных пользовательских параметров

Для CPUE05 Ethernet интерфейса могут быть сконфигурированы следующие дополнительные пользовательские параметры.

Название	Описание	По умолчанию	Диапазон
staudp	UDP порт удаленного монитора станции	18245 (4745h)	0-65535 (ffffH)
stpasswd	Пароль монитора станции	“system”	0-8 символов, различать регистры, без пробелов
crsp_tout	Задержка между передачей/ответом (в секундах)	16 (0010H)	10 – 3600 (0e10H)
fflush	Задержка кэш ARP (в секундах)	0 – 604800 (93a80H)	600 (0258H)
gctl_port	Контрольное сообщение порта UDP Ethernet Global Data	7937 (1f01H)	0-65535 (ffffH)
gdata_port	UDP порт для сообщений Ethernet Global Data, отправляемых в режиме точка-точка	18246 (4746H)	0-65535 (ffffH)
gbcast_ttl	IP время жизни для широковещательной рассылки сообщений (количество повторов)	1 (1H)	0-255 (00ffH)
gucast_ttl	IP время жизни сообщений отправляемых в режиме точка-точка (количество повторов)	16 (10H)	0-255 (00ffH)
gXX_udp	UDP порт для хост группы XX	18246 (4746H)	0-65535 (ffffH)
gXX_ttl	IP время действия сообщений (количество повторов) хост группы (многоабонентская рассылка)	1 (1H)	0-255 (00ffH)
gXX_addr	IP адрес группы для хост группы XX (должен быть адрес класса D)	224.0.7.XX	224.0.0.2 - 239.255.255.255
ittl	Время жизни по умолчанию заголовка IP (количество повторов)	64 (0040H)	0-255 (00ffH)
ifrag_tmr	Длительность задержки фрагмента IP (в секундах)	3 (0003H)	0-65535 (ffffH)
wnodelay	TCP режим без задержек (0=выключен, 1=включен)	0 (000H)	0,1
wkal_idle	Значение таймера удержания связи TCP (в секундах)	240 (00f0H) = 4.0 минуты	0-65535 (ffffH)
wkal_cnt	Количество попыток установки связи TCP	2 (0002H)	
wkal_intvl	Интервал перед установкой связи TCP (в секундах)	60 (003Ch)	
wmsl	Максимальное время жизни сегмента TCP (в секундах)	30 (001eH)	
wsnd_buf	Размер буфера отправки TCP в байтах.	4096 (1000h)	0-32767 (7fffH)
wrcv_buf	Размер буфера приема TCP в байтах.	4096 (1000h)	

nmin_poll1	NTP минимальный интервал опроса хоста 1. Значение интервала равно двоичному логарифму от времени в секундах (пример: значение 3=8 секунд, 4=16 секунд и т.д)	6 (0006H) = 64 секунды.	4 – 14 (000eH) (16 – 16384 секунды)
nmax_poll1	NTP максимальный интервал опроса хоста 1 (двоичный логарифм времени в секундах)	10 (000aH) = 1024 секунды.	
nmin_poll2	NTP минимальный интервал опроса хоста 2 (двоичный логарифм от времени в секундах)	6 (0006H) = 64 секунды.	
nmax_poll2	NTP максимальный интервал опроса хоста 2 (двоичный логарифм от времени в секундах)	10 (000aH) = 1024 секунды.	
nmin_poll3	NTP минимальный интервал опроса хоста 3 (двоичный логарифм от времени в секундах)	6 (0006H) = 64 секунды.	
nmax_poll3	NTP максимальный интервал опроса хоста 2 (двоичный логарифм от времени в секундах)	10 (000aH) = 1024 секунды.	
nsync_tout	NTP период перерыва синхронизации (в секундах). (Максимальный промежуток времени между обновлениями, при котором сохраняется синхронизация).	300 (012cH)	150 – 65535 (0096H – ffffH)

Глава 7 Работа ЦПУ

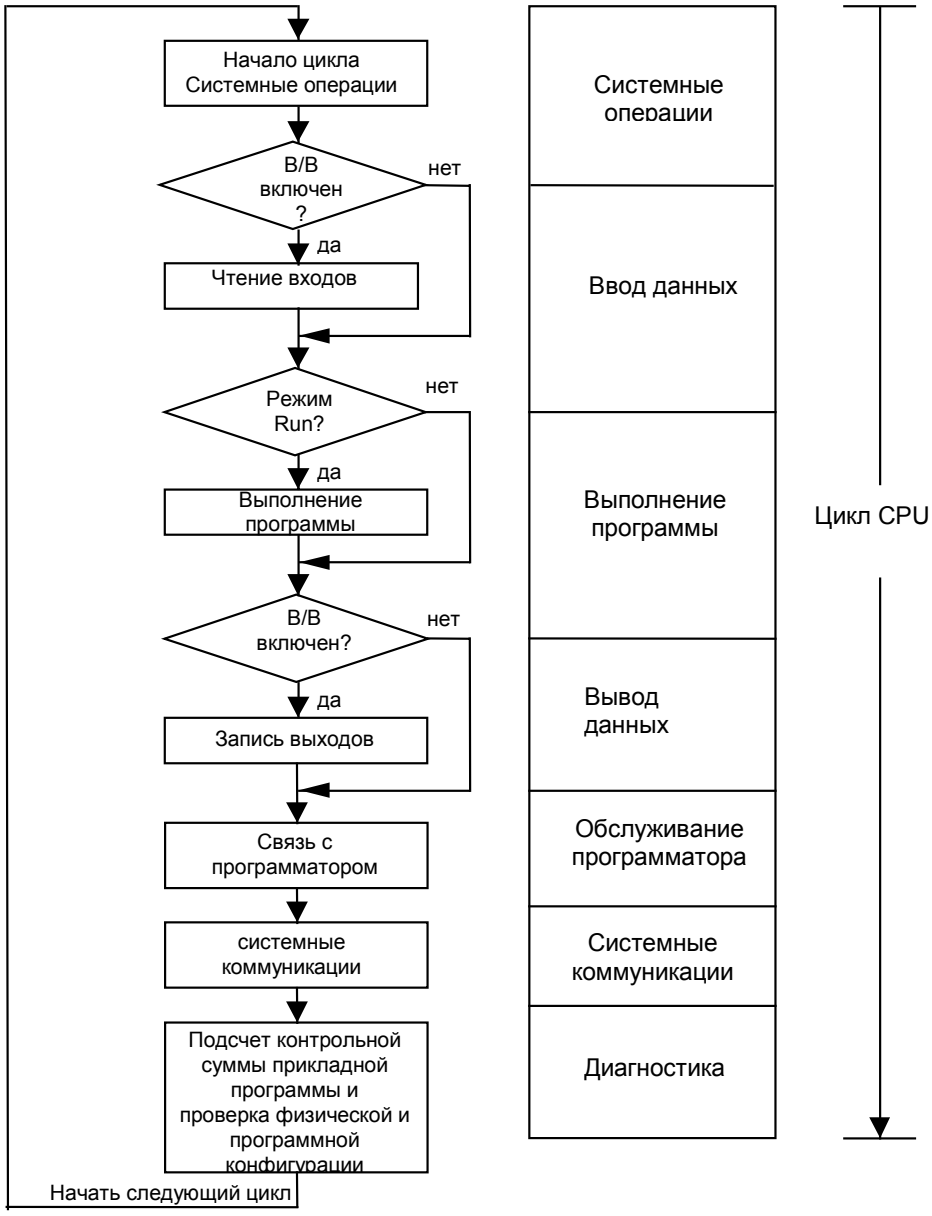
В этой главе описаны режимы работы ЦПУ ПЛК VersaMax®, а также связь между работой прикладной программы и другими задачами, выполняемыми ЦПУ.

Режимы работы ЦПУ

Прикладная программа выполняется в ПЛК циклически. Помимо выполнения прикладной программы, ЦПУ ПЛК регулярно считывает данные с устройств ввода, отправляет данные на устройства вывода, выполняет внутренние системные операции, и задачи связи . Эта последовательность операций называется циклом (**sweep)**.

- Стандартный режим работы ПЛК называется режимом стандартного цикла (**Standard Sweep)**. В этом режиме ЦПУ выполняет все части цикла в нормальном режиме. Каждый цикл выполняется как можно быстрее, но может занимать разное время.
- ПЛК может работать в режиме цикла с постоянным временем (**Constant Sweep Time)**. В этом случае ЦПУ выполняет тот же самый набор действий, но каждый цикл занимает одинаковое время.
- ПЛК может находиться в одном из двух режимов STOP:
  - Режим Stop с отключенным сканированием В/В
  - Режим Stop со включенным сканированием В/В

Составные части цикла ЦПУ



Составные части цикла ЦПУ

Начало цикла, системные операции	Системные операции включают в себя задачи по подготовке к началу цикла. Перед началом самого цикла ЦПУ: Подсчитывает длительность цикла Определяет время начала следующего цикла Определяет режим следующего цикла Обновляет таблицы ошибок Обнуляет сторожевой таймер Если ПЛК находится в режиме цикла с постоянным временем, начало цикла откладывается на установленное время. Если установленное время уже настало, то контакт OV_SWP %SA0002 установлен, и цикл продолжается без задержек. Затем ЦПУ обновляет значения таймера (сотые, десятые, и секунды).
Чтение входов	Когда начинается цикл, ЦПУ сперва опрашивает модули ввода и вспомогательные модули, которые предоставляют входные данные. Модули опрашиваются в порядке возрастания их адресов. Дискретные модули ввода опрашиваются раньше, чем аналоговые модули ввода. ЦПУ сохраняет новые входные данные в памяти соответствующих типов. Если ЦПУ было сконфигурировано не сканировать модули ввода-вывода в режиме STOP, опрос модулей ввода не производится, когда ЦПУ находится в режиме STOP. Для CPUE05: если ЦПУ находится в режиме RUN и время ожидания обмена Ethernet Global Data в получающем устройстве истекло, ЦПУ копирует данные этого обмена из интерфейса Ethernet в соответствующие участки памяти.
Решение Логики Прикладной программы	Следующий шаг. ЦПУ решает логику прикладной программы. Всегда начинается с первой команды в программе. Заканчивается при выполнении команды END. Решение логики создает новый набор выходных данных.
Сканирование выходов	Сразу после решения логики ЦПУ сканирует все модули вывода в возрастающем порядке адресов. Сканирование модулей вывода заканчивается после отправки всех данных. Если ЦПУ было сконфигурировано не опрашивать модули ввода-вывода в режиме STOP, опрос модулей вывода также не производится, когда ЦПУ находится в режиме STOP. Для CPUE05: если сканирование ввода-вывода включено и время ожидания обмена Ethernet Global Data в отправляющем устройстве истекло, ЦПУ копирует данные этого обмена из памяти в Ethernet интерфейс.
Окно связи с програм-матором	Если подключено программирующее устройство, следующим шагом ЦПУ обслуживает окно связи с программатором. Окно связи с программатором не будет обрабатываться, если программатор не подключен. В ограниченном по умолчанию режиме окна в каждом цикле ЦПУ обрабатывает один запрос. Ограничение времени одного сеанса связи с программатором равно 6 миллисекунд. Если программатор посылает запрос, обслуживание которого требует более 6 миллисекунд связи, он будет обработан в течение нескольких циклов.
Окно системных коммуникаций	Затем ЦПУ обрабатывает запросы интеллектуальных модулей. Модули опрашиваются поочередно, ни один модуль не является приоритетным. По умолчанию установлен режим работы до конца (“Run to Completion”), время соединения окна системных коммуникаций ограничено 400 миллисекундами. Если модуль посылает запрос, для обработки которого требуется свыше 400 миллисекунд, запрос разделяется на несколько циклов. В ограниченном режиме модули, связывающиеся с ПЛК через системное окно, оказывают меньше влияния на длительность цикла, но реакция системы медленее.
Диагностика	Подсчет контрольной сумму производится прикладной программой в конце каждого цикла. Вы можете определить размер контрольной суммы от 0 до 32 слов. Если подсчитанная контрольная сумма не совпадает с установленным значением, выставляется флаг ошибки контрольной суммы программы. Ошибка заносится в таблицу неисправностей ПЛК и ПЛК переходит в режим STOP. Если подсчет контрольной суммы не удается, это не влияет на окно связи с программатором. В каждом цикле ЦПУ сверяет физическую конфигурацию одного модуля с его запрограммированной конфигурацией. При обнаружении потери модуля, дополнительного модуля, или несовпадающего модуля выдается сообщение об ошибке.

7

Работа ЦПУ в стандартном цикле

Работа в режиме стандартного цикла, является обычным режимом работы ЦПУ ПЛК. В режиме стандартного цикла ЦПУ многократно выполняет прикладную программу, обновляет каналы ввода-вывода, выполняет сеансы связи и другие задачи приведенные ниже:

1. ЦПУ в начале цикла выполняет системные операции.
2. Считывает входные данные.
3. Выполняет прикладную программу.
4. Обновляет выходы
5. Если программатор подключен, ЦПУ осуществляет сеанс связи с ним.
6. Осуществляет связь с другими устройствами.
7. Выполняет диагностику

За исключением связи с программатором, все остальные функции выполняются в каждом цикле. Соединение с программатором происходит только когда оно требуется.

В этом режиме ЦПУ выполняет все части цикла в нормальном режиме. Каждый цикл выполняется как можно быстрее, но может занимать разное время.

Окна цикла

Окно связи с программатором и окно связи с системой имеют два режима работы:

Ограниченный режим	Время действия окна 6 миллисекунд. Окно закрывается, когда все задачи выполнены или 6 миллисекунд истекло.
Режим работы до завершения	Независимо от выделенного определенному окну времени, оно действует до выполнения всех поставленных задач (время действия до 400 миллисекунд).

Чтобы получить в прикладной программе текущие значения времен окон, можно использовать SVCREQ.

Сторожевой таймер

Когда ЦПУ работает в режиме стандартного цикла, сторожевой таймер отслеживает ошибки, которые могут вызвать слишком длинный цикл. Уставка сторожевого таймера равна 500 миллисекунд. Он обнуляется в начале нового цикла.

Если цикл длится больше чем 500 мс, индикатор OK модуля ЦПУ гаснет. ЦПУ перезагружается, выводит ошибку сторожевого таймера, и переходит в режим Stop. Все коммуникации временно прекращаются.

Режим работы цикла с постоянным временем

Если приложение требует, чтобы каждый цикл ЦПУ занимал одинаковое время, ЦПУ может быть сконфигурировано на работу в режиме цикла с постоянным временем. Этот режим работы обеспечивает постоянную периодичность обновления каналов ввода-вывода в системе. Также этот режим может быть использован для установки более длинных циклов, чтобы после формирования управляющего воздействия в системе успели установиться новые входные данные.

Изменение настроек по умолчанию режима цикла с постоянным временем

Если ПЛК находится в режиме STOP, режим работы цикла с постоянным временем может быть отредактирован. Чтобы внесенные изменения вступили в силу, конфигурация должна быть сохранена в ЦПУ. Сохраненный однажды, режим цикла с постоянным временем становится режимом по умолчанию.

Таймер цикла с постоянным временем

Во время работы в режиме цикла с постоянным временем таймер цикла с постоянным временем ЦПУ контролирует длительность цикла. Уставка таймера от 5 до 500 миллисекунд. Уставка должна быть хотя бы на 10 миллисекунд больше, чем длительность цикла ЦПУ при работе в режиме стандартного цикла, чтобы избежать случайного превышения времени цикла.

Если время таймера постоянного времени цикла истечет до окончания цикла, то ЦПУ завершит цикл полностью, включая все окна. Однако автоматически выводится извещение, что цикл слишком длинный. В цикле, следующем за циклом с превышением времени, ЦПУ заносит сообщение о превышении цикла в таблицу ошибок ПЛК. Затем, в начале следующего цикла, ЦПУ устанавливает контакт ошибки OV_SWP (%SA0002). ЦПУ автоматически сбрасывает ошибку OV_SWP, если длительность цикла больше не превышает уставку таймера цикла с постоянным временем. ЦПУ также обнуляет OV_SWP ошибку, если оно не находится в режиме цикла с постоянным временем.

Прикладная программа может наблюдать за этим контактом, как и за другими контактами ошибок, чтобы сигнализировать о превышении времени цикла.

Включение/выключение с постоянным временем цикла, Чтение или Изменение длительности таймера.

SVCREQ 1 может быть включен в прикладную программу для включения и выключения режима цикла, изменения длительности цикла с постоянным

временем, проверки, включен ли цикл с постоянным временем или чтения уставки его таймера.

Режимы STOP ЦПУ

ПЛК может находиться в одном из двух режимов STOP:

- Режим Stop с отключенным сканированием B/B
- Режим Stop со включенным сканированием B/B

Когда ПЛК находится в режиме STOP, ЦПУ не выполняет логику прикладной программы. Вы можете сконфигурировать будут ли каналы B/B сканироваться в режиме Stop. Связь с программатором и интеллектуальными модулями будет продолжаться в режиме Stop. Кроме того, в режиме STOP продолжается опрос отказавших модулей и переконфигурирование модулей.

SVCREQ 13 может быть использован в прикладной программе для остановки ПЛК в конце следующего цикла. Все каналы B/B перейдут в состояние, сконфигурированное по умолчанию, и диагностическое сообщение будет записано в таблицу ошибок ПЛК.

7

Управление выполнением программы

Набор команд ЦПУ VersaMax содержит несколько мощных функций управления, которые могут быть включены в прикладную программу для ограничения или изменения хода выполнения программы и сканирования В/В.

Вызов подпрограммы

Функцию вызова (CALL) можно использовать для того, чтобы выполнение программы перешло к конкретной подпрограмме. Условные операторы, помещенные перед функцией вызова (CALL), определяют условия, при которых ЦПУ выполняет логику подпрограммы. После окончания подпрограммы выполнение основной программы возобновляется со следующего, сразу после команды вызова места в логике.

Создание временного окончания логики

Функцию Окончания (END) можно использовать, для создания временного окончания логики. Она может быть помещена в любом месте программы. Логика после функции окончания не выполняется, и выполнение программы начинается сначала. Эта возможность делает функцию окончания очень полезной для отладки программы.

Функция окончания не должна помещаться в логику, связанную с управляющей структурой Языка функциональных схем или вызываемую из нее. Если это произойдет, ПЛК в конце цикла перейдет в режим STOP/FAULT и будет записана ошибка SFC_END.

Выполнение звеньев логики без подачи питания

Функция принудительного исполнения Master Control Relay может быть использована для выполнения части логики программы без подачи логического питания. Логика выполняется в направлении вперед, и катушки в этой части питания выполняются без питания. Глубина вложения функции Master Control Relay может достигать 8 уровней.

Переход в другую часть программы

Функция перехода (JUMP) может быть использована для перехода выполнения программы к другой части логики либо вперед, либо назад. Когда функция перехода включена, катушки в пропущенной части программы остаются в их исходном положении. Функции перехода также могут быть вложены.

Переходы не могут осуществляться внутри блоков, операций ЯФС, переходов ЯФС, или пре- и постпроцессоров ЯФС.

Переключатель режима RUN/STOP

Переключатель режима Run/Stop ЦПУ может быть сконфигурирован для переключения режимов работы ЦПУ Run/Stop. Он также может быть использован для блокировки перезаписи в памяти программы или конфигурации, и принудительной установки дискретных данных. По умолчанию выбор режима Run/Stop включен, а защита памяти отключена.

Конфигурируемые операции режима Run/Stop

Если переключатель режимов Run/Stop задействован, то его можно использовать для переключения ЦПУ в режим RUN.

- Если ЦПУ имеет не фатальные ошибки и не находится в в режиме Stop/Fault, перевод переключателя в положение Run приведет к переходу ЦПУ в режим Run. Ошибки не стираются.
- Если ЦПУ имеет фатальные ошибки и находится в режиме Stop/Fault, то при переводе переключателя в положение Run, светодиод Run будет мигать 5 секунд. Пока светодиод Run мигает, переключатель ЦПУ можно использовать для очистки таблицы ошибок и переключения ЦПУ в режим Run. После того как переключатель находился в положении Run хотя бы ½ секунды, переведите его в положение Stop хотя бы на ½ секунды. Потом верните обратно в положение Run. Ошибки будут удалены, и ЦПУ перейдет в режим Run. Светодиод перестанет мигать и останется постоянно включенным. При необходимости, эту операцию можно повторить.
- Еспли переключатель не был переключен, как описано выше, то после 5 секунд светодиод Run погаснет и ЦПУ останется в режиме Stop/Fault. Ошибки останутся в таблице ошибок.

Защита конфигурируемой памяти

Переключатель можно сконфигурировать для блокировки перезаписи в памяти программы или конфигурации, и принудительной установки дискретных данных.

Варианты работы переключателя Run/Stop

Конфигурация режима Run/Stop	Конфигурация сканирования B/B в режиме STOP	Положение переключателя	Работа ЦПУ
Выключен	не действует	не действует	Все режимы доступны.
Включен	не действует	Run/On	Все режимы доступны.
Включен	не действует	Stop/Off	ЦПУ не может перейти в режим Run.

Выключен	не действует	Переключен из положения Stop в положение Run	ЦПУ переходит в режим Run если нет фатальных ошибок; в противном случае, светодиод Run мигает 5 секунд.
Включен	Нет	Переключен из положения Run в положение Stop	PLC переходит в режим STOP–NO IO (без сканирования B/B)
Включен	Да	Переключен из положения Run в положение Stop	PLC переходит в режим STOP–IO (без сканирования B/B)

Флэш память

ПЛК VersaMax сохраняет текущую конфигурацию и приложение в энергонезависимом ОЗУ с поддержкой от батарейки. Инструментальное программное обеспечение можно использовать для записи текущей конфигурации, прикладной программы и данных (исключая принудительно установленные значения) во флэш памяти. Программатор можно также использовать для чтения ранее сохраненной конфигурации, прикладной программы, или таблиц ссылок из флэш памяти в ОЗУ, или для проверки ОЗУ и флэш памяти на идентичность.

По умолчанию ПЛК при включении считывает конфигурацию, логику программы и таблицы ссылок из ОЗУ. Однако, его можно сконфигурировать для загрузки из флэш памяти. Это рекомендуется, так как флэш память энергонезависимая, программа не будет потеряна даже в случае отказа батарейной поддержки.

7

Уровни доступа и Пароли

Пароли являются дополнительным конфигурируемым сервисом ПЛК VersaMax. Пароли предоставляют различные уровни права доступа в ПЛК, когда программатор находится в режиме Online или Monitor. Пароли не используются, если программатор находится в режиме Offline. Пароли могут ограничивать:

- Изменение данных конфигурации В/В и ПЛК
- Изменение программ
- Чтение данных ПЛК
- Чтение программ

В ПЛК устанавливается один пароль на каждый уровень доступа. Можно устанавливать одинаковые или разные пароли для нескольких уровней. Длинна пароля составляет от 1 до 7 ASCII знаков.

По умолчанию все пароли выключены. Пароли устанавливаются, изменяются или отменяются при помощи инструментального программного обеспечения. После установки паролей доступ к ПЛК запрещен до ввода правильного пароля. Ввод правильного пароля предоставляет доступ к указанному уровню и ко всем более низким уровням. Например, пароль уровня 3 предоставляет доступ к уровням 0, 1, 2 и 3. Если связь с ПЛК прерывается, защита автоматически возвращает доступ на самый высокий незащищенный уровень. Например, если пароль установлен на уровни 2 и 3, и не установлен на уровень 4, и если программатор отключается и подключается заново, после восстановления соединения будет уровень доступа 4. Для уровня 1 пароль не устанавливается, поэтому он всегда доступен.

Уровень	Описание Доступа
4 Наименее защищенный	▪ Полный доступ к записи конфигурации и логики. Конфигурацию можно записать в режиме Stop; логика может быть записана в режимах Stop или Run (если сохранение в режиме Run поддерживается). ▪ Установка или удаление паролей для любого уровня. ▪ Плюс полный доступ к возможностям уровней 1, 2 и 3. ▪ Примечание: это разрешено по умолчанию, если не установлен пароль.
3	▪ Полный доступ к записи конфигурации и логики, когда ЦПУ находится в режиме Stop, включая замену слова на слово (если поддерживается), добавление/удаление логики программы, и принудительную установку дискретных каналов В/В. ▪ Считывание/Запись/Проверка пользовательской флэш памяти. ▪ Запись принудительной установки. ▪ Изменение режима цикла. ▪ Плюс полный доступ к возможностям уровней 1 и 2.
2	▪ Запись всех данных памяти, кроме таблиц. ▪ Принудительная установка памяти, но не каналов В/В. ▪ ПЛК можно запускать или останавливать.

	▪ Таблицы ошибок ПЛК и В/В могут быть очищены. ▪ Плюс полный доступ к возможностям уровня 1.
1 Наиболее защищенный	Чтение любых данных ПЛК, кроме паролей. Включает в себя: чтение таблиц ошибок, текущего статуса, выполнение datagram, проверку логики/конфигурации и загрузку программ и конфигурации из ПЛК. Никакие данные памяти ПЛК не могут быть изменены.

Запрос уровня защиты из программатора

После соединения с ЦПУ инструментальное программное обеспечение автоматически запрашивает у ЦПУ переход на самый высокий незащищенный уровень. Это дает возможность программатору иметь доступ к самому высокому незащищенному уровню без уточняющего запроса какого-либо уровня.

Доступ может быть изменен на более низкий или высокий уровень. Уровень доступа изменяется из программатора при вводе нужного уровня и правильного пароля для этого уровня. Если введен неправильный пароль, то изменения не принимаются, и ошибка записывается в таблицу ошибок ПЛК. Запрос на изменение доступа к незащищенному паролем уровню осуществляется введением нового уровня и пустого пароля.

Примечания по использованию паролей

- Для включения паролей после их отмены ПЛК нужно выключить, и снять батарейку на время, достаточное для разрядки конденсатора и удаление памяти ПЛК.
- Если пароли запрещают переключение режимов Run/Stop, обновления системного программного обеспечения не могут производиться, когда ПЛК находится в режиме Run.
- Переключатель режимов Run/Stop (если сконфигурирован) будет переключать режимы Run и Stop независимо от паролей.

Сервис OEM защиты

Сервис OEM защиты аналогичен паролям, но предоставляет более высокий уровень защиты. Он включается и отключается вводом пароля (1-7 знаков), так называемого ключа *OEM*. Когда OEM защита включена, не разрешается запись программ и конфигурации в ПЛК. Чтение конфигурации ПЛК разрешается. В этом режиме операции над пользовательской флэш памятью не разрешаются

Когда ключ OEM создан, защиту можно включить двумя способами: выбрать в инструментальном программном обеспечении установку защиты или выключить и включить ПЛК. (Статус защиты OEM не изменяется, если связь с ПЛК прерывается.)

Удаление логики, конфигурации и данных

Логика, конфигурация и данные могут быть удалены с помощью программатора из ЦПУ при любом уровне доступа, и даже с включенной защитой OEM. Операторы могут удалять логику, конфигурацию и ссылки, а также сохранять новую прикладную программу в ЦПУ, не зная паролей.

Если пароли и/или защита OEM были установлены и записаны во флэш память, чтение флэш памяти обновляет уровень защиты. В этом случае не обязательно вводить пароль для получения доступа к определенному уровню. Функция Удалить все (Clear All) не удаляет содержимое пользовательской флэш памяти.

Глава
8

Организация прикладной программы

В этой главе предоставляется основная информация о прикладной программе ПЛК VersaMax®.

- Структура прикладной программы
- Подпрограммы
- Языки программирования
- Набор команд

8

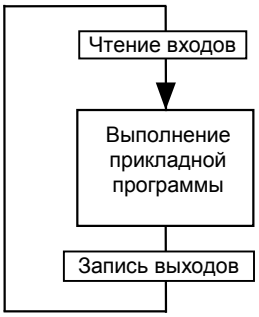
Структура прикладной программы

Прикладная программа состоит из всей логики, необходимой для управления работой ЦПУ ПЛК и модулей системы.

Прикладные программы создаются с помощью инструментального программного обеспечения и записываются в ПЛК. Программы хранятся в энергонезависимой памяти ЦПУ.

Во время цикла ЦПУ (описанного в предыдущей главе) ЦПУ считывает входные данные из модулей системы и сохраняет их в сконфигурированных для входных данных ячейках памяти. ЦПУ выполняет сразу всю прикладную программу, используя эти обновленные входные данные. Выполнение прикладной программы создает новые выходные данные, которые помещаются в сконфигурированные ячейки памяти.

После выполнения прикладной программы ЦПУ записывает выходные данные в модули системы.

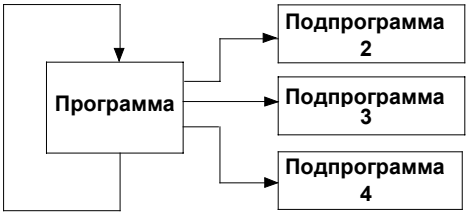


Подпрограммы

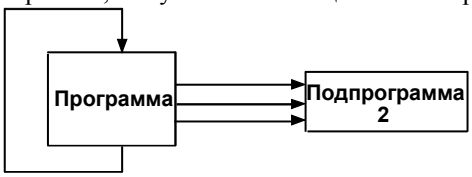
Программа может состоять из одной основной программы, которая выполняется полностью в течение каждого цикла ЦПУ.

Или программу можно разделить на подпрограммы. Максимальный размер основной программы или блока подпрограммы 64 Кб. Программа может содержать до 255 подпрограмм.

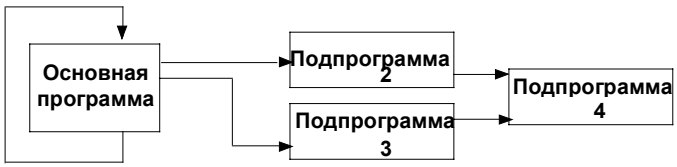
Использование подпрограмм может упростить программирование и уменьшить общее количество логики. Каждая подпрограмма вызывается, когда она требуется. Основную программу можно использовать для организации вызова подпрограмм.



К блоку подпрограммы можно обращаться несколько раз во время выполнения программы. Логика, которая должна повторяться, может быть помещена в блок подпрограммы, это уменьшает общий объем программы.



Помимо того, что вызов подпрограммы может осуществляться из основной программы, он также может осуществляться из других подпрограмм. Блок подпрограммы может вызывать сам себя.



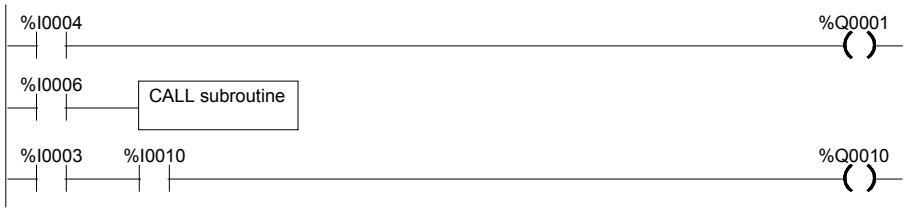
Основная программа находится на уровне 1. Программа может включать до восьми вложенных уровней обращения.

Объявление подпрограммы

Подпрограмма должна быть объявлена при помощи редактора объявлений блоков инструментального программного обеспечения.

Вызов подпрограммы

Подпрограмма интегрируется в программу при помощи команды вызова CALL. В каждом блоке программы допускается 64 декларации блоков подпрограмм и 64 команды вызова CALL.



Языки программирования

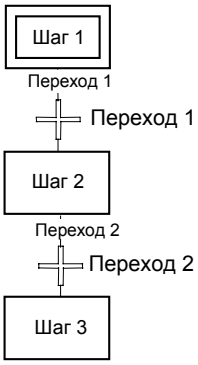
Программы создаются на языке релейно-контактной логики или на языке текстовых команд. Основная программа или подпрограммы внутри программы могут также создаваться на языке функциональных схем. Инструментальное программное обеспечение можно использовать для создания логики обоих типов.

Язык функциональных схем

Язык функциональных схем (ЯФС) представляет собой графический метод изображения функций последовательной автоматизированной системы в виде последовательности шагов и переходов. Каждый шаг представляет собой команды или действия, которые либо активны, либо не активны.

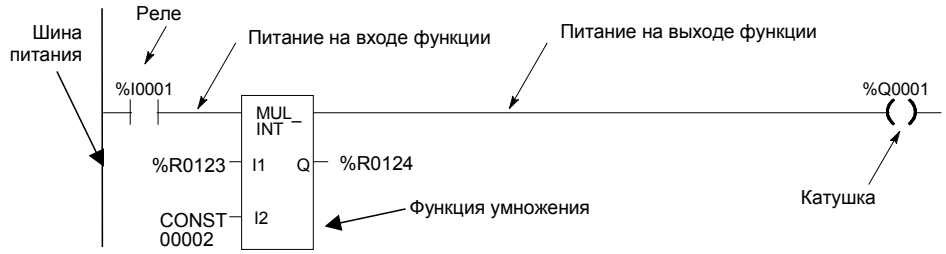
Управление передается от одного шага к другому через условные переходы, которые либо истинны (1), либо ложны (0). Если условие перехода истинно (1), то управление передается с текущего шага (он становится неактивным) к следующему, который становится активным.

Логика, соответствующая шагу, выполняется, когда шаг активен. Эта логика программируется на языке релейно-контактной логики. Переходы между шагами также программируются на языке релейной логики.



Язык релейно-контактной логики

Это традиционный язык программирования ПЛК, логика ступенчатой структуры выполняется сверху вниз. Выполнение логики аналогично протеканию электрического тока, происходящему сверху вниз по левой шине схемы и слева направо в каждом звене.



Прохождение логического питания через каждое звено управляется набором простых команд, которые работают как механические реле и выходные катушки. Пропустит ли реле питание логики через звено или нет, зависит от содержимого ячейки памяти, с которой реле связано в программе. Например, реле может пропустить питание, если оно связанно с ячейкой памяти содержащей значение 1. Это реле не пропустит питание, если ячейка памяти содержит значение 0.

Если реле или другая функция в звене не пропускает логическое питание, оставшаяся часть звена не выполняется. Тогда питание проходит по левой шине к следующему звену.

Внутри звена может находиться много сложных функций, которые могут быть использованы для перемещения данных в памяти, выполнения математических операций и управления связью между ЦПУ и другими устройствами в системе.

Некоторые функции, такие как Переход (Jump) и Master Control Relay, могут быть использованы для управления выполнением самой программы.

Совокупность реле, катушек и функций релейно-контакной логики, называется системой команд ЦПУ.

Набор команд

ЦПУ ПЛК VersaMax имеет мощную систему команд для создания прикладной программы.

Для справки о программных возможностях ПЛК VersaMax все реле, катушки, функции, и другие элементы системы команд собраны на последующих страницах. Полная информация включена в документацию и систему справок инструментального программного обеспечения.

Контакты

–| |–	**Нормально разомкнутый**	Пропускает питание, если соответствующая ячейка включена.
–|/|–	**Нормально замкнутый**	Пропускает питание, если соответствующая ячейка выключена.
<+>——	**Продолжение**	Пропускает питание направо, если предшествующая катушка продолжения включена (запитана).

Катушки

–( )–	**Обычная**	Включает соответствующую ячейку, если катушка получает питание. В противном случае ячейка отключена.
–(/)–	**Инверсия**	Включает соответствующую дискретную ячейку, если катушка не получает питание. В противном случае ячейка отключена.
–(↑)–	**Положительный переход**	Если питание этой катушки было отключено в предыдущем цикле и в данный момент включено, то катушка включается на время текущего цикла. В противном случае катушка отключена.
–(↓)–	**Отрицательный переход**	Если питание этой катушки было включено в предыдущем цикле и в данный момент отключено, то катушка включается на время текущего цикла. В противном случае катушка отключена.
–(S)–	**SET (фиксация)**	Если катушка получает питание, включает соответствующую дискретную ячейку. Ячейка остается включенной до выключения катушкой –(R)–.
–(R)–	**RESET (сброс)**	Если катушка получает питание, выключает соответствующую дискретную ячейку. Ячейка остается выключенной до включения катушкой –(S)–.
–(SM)–	**Записываемая фиксация**	Если катушка получает питание, включает соответствующую ячейку. Ячейка остается включенной до отключения катушкой –(RM)–. Состояние ячейки остается неизменным при перебоях в питании и **останове и перезапуске контроллера.**
–(RM)–	**Записываемый сброс**	Если катушка получает питание, выключает соответствующую дискретную ячейку. Ячейка выключена до включения катушкой –(SM)–. Состояние ячейки остается неизменным при перебоях в питании и **останове и перезапуске контроллера.**
–(/M)–	**Записываемая инверсия**	Включает соответствующую дискретную ячейку, если катушка не получает питание. Состояние ячейки остается неизменным при перебоях в питании и **останове и перезапуске контроллера.** В противном случае ячейка

		отключена.
–(M)–	**Записываемая катушка**	Если катушка получает питание, включает соответствующую дискретную ячейку. Состояние ячейки остается неизменным при перебоях в питании и **останове и перезапуске контроллера.** В противном случае ячейка отключена.
—<+>	**Продолжение**	Если на эту катушку подается питание, катушка продолжения включает следующий контакт продолжения. Если питание отключено, катушка выключает следующий контакт.

Таймеры и счетчики

ondtr	**Таймер задержки по включению со сбросом**	Суммирует время при включенном питании. Текущее значение обнуляется при прохождении питания на вход сброса (Reset).
oftd	**Таймер задержки по выключению**	Суммирует время при выключенном питании.
tmr	**Таймер задержки по включению**	Суммирует время при включенном питании. Текущее значение обнуляется при отключении питания.
upctr	**Инкрементный счетчик**	Увеличивается на 1 каждый раз, когда на вход поступает импульс.
dnctr	**Декрементный счетчик**	Уменьшается от исходного значения на 1, каждый раз, когда на вход поступает импульс.

Математические функции

add	**Сложение**	Складывает 2 числа.
sub	**Вычитание**	Вычитает одно число из другого.
mul	**Умножение**	Умножает 2 числа.
div	**Деление**	Делит одно число на другое, выводя частное.
mod	**Деление по модулю**	Делит одно число на другое, выводя остаток.
expt	**Сепень X**	Возводит X в степень IN и помещает результат в Q.
sin	**Тригонометрический Синус**	Находит тригонометрический синус вещественного числа.
cos	**Тригонометрический Косинус**	Находит тригонометрический косинус вещественного числа.
tan	**Тригонометрический Тангенс**	Находит тригонометрический тангенс вещественного числа.
asin	**Арксинус**	Находит арксинус вещественного числа.
acos	**Арккосинус**	Находит арккосинус вещественного числа.
atan	**Арктангенс**	Находит арктангенс вещественного числа.
deg	**Перевести в градусы**	Выполняет преобразование вещественного значения радиан в градусы.
rad	**Перевести в радианы**	Выполняет преобразование вещественного значения градусов в радианы.
scale	**Масштабирование**	Масштабирует константу или слово.
sqroot	**Квадратный корень**	Находит квадратный корень целого или вещественного числа.
Log	**Десятичный логарифм**	Находит десятичный логарифм вещественного числа.
ln	**Натуральный логарифм**	Находит натуральный логарифм вещественного числа.
exp	**Экспонента**	Возводит основание натурального логарифма в указанную степень.

Функции сравнения

eq	**Равно**	Проверяет равенство двух чисел.
ne	**Не равно**	Проверяет неравенство двух чисел.
gt	**Больше**	Проверяет, больше ли одно число, чем другое. Пропускает питание если первое число больше второго.
ge	**Больше или равно**	Проверяет, больше или равно одно число, чем другое.
lt	**Меньше**	Проверяет, больше ли одно число, чем другое.
le	**Меньше или равно**	Проверяет, меньше или равно одно число, чем другое.
range	**Диапазон**	Проверяет, находится ли число в диапазоне, ограниченном двумя числами.

Функции Битовых Операций

and	**Логическое "и"**	Выполняет логическое "и" для двух битовых строк.
or	**Логическое "или"**	Выполняет логическое "или" для двух битовых строк.
xor	**Логическое ислючающее "или"**	Выполняет логическое исключающее "или" для двух битовых строк.
not	**Логическое инвертирование**	Выполняет логическое инвертирование битовой строки.
shl	**Смещение влево**	Смещает влево битовую строку.
shr	**Смещение вправо**	Смещает вправо битовую строку.
rol	**Циклическое**	Циклически смещает влево битовую строку.
ror	**Циклическое смещение вправо**	Циклически смещает вправо битовую строку.
bittst	**Тест бита**	Тестирует бит в битовой строке.
bitset	**Установка бита**	Устанавливает для одного из битов в строке значение "истинно"
bitclr	**Сброс бита**	Устанавливает для одного из битов в строке значение "ложно".
bitpos	**Позиция бита**	Определяет набор "истинных" битов в строке.
mskcmp	**Маскированное сравнение**	Выполняет маскированное сравнение двух массивов..

Функции Пересылки Данных

move	**Пересылка**	Пересылает 1 или более бит данных.
blkmov	**Перемещение блока**	Пересылает блок до 7 констант
blkclr	**Очистка блока**	Обнуляет один или более байт/слов памяти.
shfreg	**Сдвиг регистра**	Сдвигает одно или более слов или бит данных по блоку памяти.
bitseq	**Одноконтактный командоаппарат**	Поочередно выдает по 1 биту из хранящейся в памяти ПЛК последовательности. Эмулирует одноконтактный командоаппарат
comreq	**Запрос связи**	Посылает запрос связи.

Табличные функции

arrmov	**Пересылка массива**	Копирует указанное число элементов данных из исходного массива в указанный массив.
srh eq	**Поиск равных**	Производит поиск в массиве значений, равных указанному.
srh ne	**Поиск не равных**	Производит поиск в массиве значений, не равных указанному.
srh gt	**Поиск больших**	Производит поиск в массиве значений, больших указанного.
srh ge	**Поиск больших или равных**	Производит поиск в массиве значений, больших или равных указанному.
srh lt	**Поиск меньших**	Производит поиск в массиве значений, меньших указанного.
srh le	**Поиск меньших или равных**	Производит поиск в массиве значений, меньших или равных указанному.

Функции преобразования

→bcd4	**Преобразовывает в BCD4 (из INT)**	Преобразует целое число в 4значный BCD (двоично-десятичный) формат.
→word	**Преобразовывает в WORD (из REAL)**	Преобразует вещественное число в формат слова.
→int	**Преобразовывает в INT (из BCD4 или REAL)**	Преобразует число в формат целого числа со знаком.
→tdint	**Преобразовывает в DINT (из BCD4 или REAL)**	Преобразует число в формат целого числа двойной точности.
→real	**Преобразовывает в REAL (из INT, DINT, BCD4 или WORD)**	Преобразует значение в формат вещественного числа.
→→int	**Отбрасывание до INT (из REAL)**	Преобразует в 16 битовое число со знаком. Диапазон от –32,768 до +32,767 (дробная часть отбрасывается).
→→dint	**Отбрасывание до INT двойной точности (из REAL)**	Преобразует в 32 битовое число со знаком. Диапазон от –2,147,483,648 до +2,147,483,647 (дробная часть отбрасывается).

Функции управления

call	**Вызов**	Вызывает указанный блок подпрограммы.
do io	**выполнить I/O**	Немедленно использует указанный диапазон вводов и выводов (все вводы и выводы модуля будут использованы, если какие-либо адреса этого модуля включены в функцию – частичные обновления модулей I/O **не выполняются)**
pidind	**независимый алгоритм ПИД**	Выбирает алгоритм ПИД в независимом представлении.
pidisa	**алгоритм ISA ПИД**	Выбирает алгоритм ПИД в формате ISA .
end	**Временное окончание логики**	Программа выполняется от первого до последнего звена, или до команды END, смотря что встретится раньше. Команда полезна для отладки.
commnt	**Комментарий**	Пояснение звена.
svcreq	**Системный запрос**	Специальная сервисная функция PLC.
mcr	**Master Control Relay (принудительное исполнение)**	Начинает диапазон принудительного исполнения. MCR вызывает принудительное выполнение всех звеньев между MCR и соответствующей командой ENDMCR без подачи питания. Уровень вложения MCR может достигать 8.
endmcr	**End Master Control Relay (конец принудительного исполнения)**	Заканчивает диапазон действия Master Control Relay.
jump	**Переход**	Переход в место в логике, указанное меткой.
label	**Метка**	Цель команды перехода. Несколько команд перехода могут обращаться к одной метке.
drumseq	**Имитатор командоаппарата**	(В перспективе) Работает как механический командоаппарат с барабаном, который формирует 16-разрядный выход, используя массив хранящихся шаблонов .

Глава
9

Структура данных

В этой главе описаны типы данных, используемых прикладной программой, и объясняется, как эти данные хранятся в памяти ПЛК VersaMax®.

- Ячейки данных в памяти
- Сохранность данных
- Использование имен и описаний для программных ссылок
- Ячейки состояния системы
- Импульсные контакты времени
- Как функции в программе обрабатывают числовые данные

Ячейки данных в памяти

ПЛК хранит данные в памяти, ориентированной как на хранение бит, так и на хранение слов. Оба вида памяти разделены на различные типы, каждый со своими особенными характеристиками.

Обычно, каждый тип используется для данных определенного вида, как описано ниже. Однако обеспечивается большая гибкость при использовании памяти на практике.

Отдельные ячейки памяти адресуются при помощи буквенно-цифровой идентификатора, называемого ссылкой. Буквенная приставка ссылки идентифицирует участок памяти. Цифровое значение- это смещение в пределах участка памяти.

Адреса слов в памяти

Каждый адрес слова в памяти (ссылка) указывает на 16-битовое слово. ПЛК использует три типа ссылок для данных хранящихся в памяти слов.

%AI	Используется для аналоговых входов.
%AQ	Обычно используется для аналоговых выходов.
%R	Регистры, обычно используются для хранения данных.

Память слов представлена ниже. Пример показывает 10 адресов. По каждому адресу находятся 16 бит, составляющих одно значение. ПЛК не может обращаться к отдельным битам в памяти слов.

адрес		
	1	12467
	2	12004
	3	231
	4	359
	5	14
	6	882
	7	24
	8	771
	9	735
	10	000

Адреса бит в памяти

Каждый адрес бита (ссылка) указывает на отдельный бит. Данные хранятся в битовой памяти, как показано ниже. На иллюстрации представлены 160 отдельно адресуемых бит; бит с адресом 1 находится в левом верхнем углу, бит с адресом 160 - в правом нижнем.

addresses

1	**2**	**3**	**4**	**5**	**6**	**7**	**8**								
0	0	1	1	0	0	0	0	0	0	0	0	1	0	0	0
0	0	1	1	0	0	0	0	0	0	1	0	1	0	0	0
1	1	1	1	0	0	0	1	1	0	0	1	0	0	0	0
1	1	0	0	0	0	0	1	1	1	0	0	1	0	1	0
0	1	0	1	0	0	0	1	0	1	0	1	0	0	0	0
1	1	0	0	0	0	0	1	1	1	0	0	1	0	1	0
1	1	0	1	0	0	0	1	1	1	0	1	0	0	0	0
1	1	0	0	0	0	0	1	1	0	1	1	1	0	1	1
1	0	0	1	0	0	0	1	1	0	1	1	1	0	0	1
0	0	0	1	0	0	0	0	1	0	1	0	1	0	0	1

... 160

ПЛК использует 6 типов ссылок для данных, хранящихся в битовой памяти.

%I	Обычно используется для дискретных входов и просматривается в таблице входов.
%Q	Обычно используется для физических каналов вывода и просматривается в таблице входов. %Q ячейка может быть записываемой или не записываемой в зависимости от ее использования в программе.
%M	Обычно используются для представления внутренних переменных. %M ячейка может быть записываемой или не записываемой в зависимости от ее использования в программе.
%T	Используется для временных данных, которые можно использовать в программе несколько раз. Данные с %T адресами не сохраняются при отключении питания или остановке или перезапуске контроллера. %T ячейки нельзя использовать с записывающими катушками.
%S	Ячейки статуса системы: ■ %S, %SA, %SB, и %SC можно использовать с любыми контактами релейно-контактной логики. ■ %SA, %SB, и %SC можно использовать с записывающими катушками. ■ %S можно использовать в качестве входных данных функций и функциональных блоков. ■ %SA, %SB, и %SC можно использовать на входе и выходе функциональных блоков.
%G	Используются для Global Data. Данные в %G сохраняются при потере питания. %G ячейки можно использовать с контактами и записывающими катушками, но нельзя использовать с не записывающими катушками.

Биты перехода и биты принудительной установки

%I, %Q, %M, и %G ячейки имеют соответствующие биты перехода и принудительной установки.
%T, %S, %SA, %SB, и %SC ячейки имеют только соответствующие биты перехода.

ЦПУ использует биты перехода для катушек перехода. Когда биты принудительной установки установлены, содержимое соответствующих ячеек можно изменить только из программатора.

Сохранность данных

Некоторые данные автоматически сохраняются при остановке или включении/выключении питания ПЛК. Следующие данные сохраняются:

- Логика программы
- Таблицы ошибок и диагностики
- Принудительная установка
- Содержимое памяти слов (%R, %AI, %AQ)
- Битовые данные (%I, %SC, %G, биты ошибок и зарезервированные биты)
- Слова, сохраненные в %Q и %M.
- Данные в %Q или %M ячейках, используемых как выходы функционального блока или с записывающими катушками:

  -(M)- записывающие катушки
  -(/M)- записывающие катушки с инверсией
  -(SM)- записывающие SET катушки
  -(RM)- записывающие RESET катушки

  Тип катушки, последний раз использованной с ячейкой %Q или %M, определяет, будут ли данные сохранены. Например, если последнее обращение к ячейке %Q0001было запрограммировано с использованием записывающей катушки, данные %Q0001 будут сохраняться. Однако, если последнее обращение к ячейке %Q0001 было запрограммировано с помощью не записывающей катушки, данные %Q0001 не будут сохраняться.
- %Q или %M ячейки объявленные, как записываемые. %Q и %M ячейки по умолчанию сохраняются.

Следующие данные не записываемые:

- Состояния катушек перехода.
- %T данные
- %S, %SA, и %SB данные (но данные %SC записываемые).
- %Q и %M ячейки, которые не были объявлены как записываемые.
- %Q и %M ячейки, которые используются с не записывающими катушками:

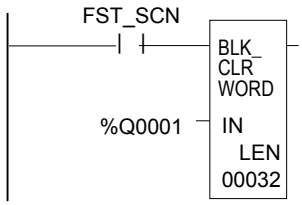
  -( )- катушки
  -(/)- катушки с инверсией
  -(S)- SET катушки
  -(R)- RESET катушки

Ячейки состояния системы

ПЛК сохраняет состояние системы в заранее определенных ячейках в памяти %S, %SA, %SB, и %SC. Каждая ссылка состояния имеет поясняющее название. Например, ячейки периодических контактов времени называются T_10MS, T_100MS, T_SEC, и T_MIN. Примеры удобной мнемоники включают: FST_SCN, ALW_ON, и ALW_OFF.

Использование ячеек состояния системы

Ячейки состояния системы могут быть, при необходимости, использованы в прикладной программе. Например, приведенный ниже функциональный блок использует ячейку состояния FST_SCN (первый цикл) для активизации функции очистки блока (Block Clear). В этом примере при включении контроллера 32 слова в %Q памяти (512 точек), начиная с %Q0001, заполняются нулями.



%S ячейки

Ячейки в %S памяти доступны только для чтения.

Ячейка	Название	Описание
%S0001	FST_SCN	Установлено в 1, когда текущий цикл является первым.
%S0002	LST_SCN	Сбрасывается с 1 на 0, когда текущий цикл является последним
%S0003	T_10MS	Периодический контакт времени 0.01 секунды.
%S0004	T_100MS	Периодический контакт времени 1секунды.
%S0005	T_SEC	Периодический контакт времени 1.0 секунда.
%S0006	T_MIN	Периодический контакт времени 1.0 минута
%S0007	ALW_ON	Всегда включено.
%S0008	ALW_OFF	Всегда выключено.
%S0009	SY_FULL	Устанавливается, когда таблица ошибок ПЛК заполняется. Сбрасывается, когда удаляется запись или очищается вся таблица ошибок ПЛК.
%S0010	IO_FULL	Устанавливается, когда таблица ошибок В/В заполняется. Удаляется когда удаляется запись или очищается вся таблица ошибок I/O.
%S0011	OVR_PRE	Устанавливается, когда в памяти %I, %Q, %M, или %G присутствует принудительная установка.
%S0012		Зарезервировано
%S0013	PRG_CHK	Устанавливается, когда осуществляется проверка программы в фоновом режиме.
%S0014	PLC_BAT	Устанавливается при неисправности батареи в ЦПУ. Состояние контакта обновляется в каждом цикле.
%S0015, 16		зарезервировано
%S0017	SNPXACT	SNP-X хост подключен к ЦПУ через порт 1. (Порт 2 по умолчанию выключен, и должен быть включен при помощи COMMREQ).
%S0018	SNPX_RD	SNP-X хост считал данные из порта 1 ЦПУ.
%S0019	SNPX_WT	SNP-X хост записал данные в порт 1 ЦПУ.
%S0020		Включается, когда функция сравнения, использующая данные формата REAL, выполняется успешно. Сбрасывается, когда на одном из входов NaN (Не число).
%S0021	FF_OVR	Устанавливается при игнорировании фатальных ошибок.
%S0022	USR_SW	Отображает состояние переключателя режимов ЦПУ. 1 = Run/On 0 = Stop/Off
%S0023-32		Зарезервировано

Ячейки %SA, %SB, и %SC

Ячейки памяти %SA, %SB, %SC доступны для чтения и записи.

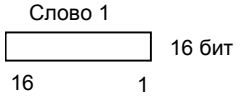
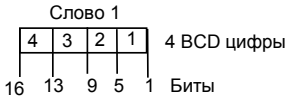
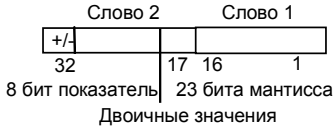
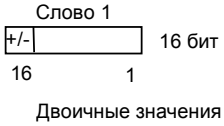
Ячейка	Название	Описание
%SA0001	PB_SUM	Устанавливается, когда подсчитанная контрольная сумма прикладной программы не совпадает с опорной контрольной суммой. Если ошибка была вызвана временным сбоем, этот дискретный бит можно сбросить, заново сохранив программу в ЦПУ. Если ошибка была вызвана неисправностью ОЗУ, то ЦПУ нужно заменить.
%SA0002	OV_SWP	Устанавливается, когда ПЛК в режиме цикла с постоянным временем обнаруживает, что предыдущий цикл занял больше времени, чем сконфигурировано. Сбрасывается, когда ПЛК определяет что предыдущий цикл не превысил отведенное ему время. Также сбрасывается при переходе из режима STOP в режим RUN.
%SA0003	APL_FLT	Устанавливается, когда имеет место ошибка приложения. Сбрасывается, когда ПЛК переходит из режима STOP в режим RUN.
%SA0004-8		Зарезервировано
%SA0009	CFG_MM	Устанавливается, когда обнаружено несовпадение конфигурации при подаче питания или сохранении конфигурации. Сбрасывается при включении ПЛК после устранения неисправности.
%SA0010	HRD_ЦПУ	Устанавливается, когда диагностика определяет неисправность оборудования ЦПУ. Сбрасывается заменой модуля ЦПУ.
%SA0011	LOW_BAT	Устанавливается при разряде батареи. Сбрасывается при включении питания ПЛК после замены батареи .
%SA0012,13		Зарезервировано
%SA0014	LOS_IOM	Устанавливается при обрыве связи модуля В/В с ЦПУ. Сбрасывается при включении/выключении после замены модуля.
%SA0015	LOS_SIO	Устанавливается при обрыве связи вспомогательного модуля с ЦПУ. Удаляется при включении/выключении питания главного крейта после замены модуля
%SA0016-18		Зарезервировано
%SA0019	ADD_IOM	Устанавливается при добавлении модуля В/В. Сбрасывается при включении/выключении ПЛК, если оборудование соответствует конфигурации.
%SA0020	ADD_SIO	Устанавливается при добавлении вспомогательного модуля. Сбрасывается при включении/выключении ПЛК, если оборудование соответствует конфигурации.
%SA0021-26		Зарезервировано
%SA0027	HRD_SIO	Устанавливается при обнаружении неисправности во вспомогательном модуле. Сбрасывается при включении/выключении питания ПЛК после замены модуля.

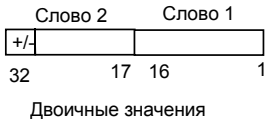
%SA0028-30		зарезервировано
%SA0031	SFT_SIO	Устанавливается при обнаружении непоправимой программной ошибки во вспомогательном модуле. Сбрасывается при включении/выключении питания ПЛК, если оборудование соответствует конфигурации.

Ячейка	Название	описание
%SB0001-9		Зарезервировано
%SB0010	BAD_RAM	Устанавливается, если при включении питания ЦПУ обнаруживает ошибку ОЗУ. Сбрасывается, если ОЗУ исправно при включении.
%SB0011	BAD_PWD	Устанавливается, если были нарушены права доступа, ограниченные паролем. Сбрасывается после очистки таблицы ошибок ПЛК.
%SB0012		Зарезервировано
%SB0013	SFT_ЦПУ	Устанавливается, если ЦПУ обнаруживает непоправимую ошибку в программе. Сбрасывается после очистки таблицы ошибок ПЛК.
%SB0014	STOR_ER	Устанавливается, если возникает ошибка при сохранении из программатора. Сбрасывается, когда процедура сохранения выполнена успешно.
%SC0001-8		Зарезервировано
%SC0009	ANY_FLT	Устанавливается при возникновении любой ошибки. Сбрасывается, когда отсутствуют записи в обеих таблицах.
%SC0010	SY_FLT	Устанавливается при возникновении любой ошибки, которая заносится в таблицу ошибок ПЛК. Сбрасывается, когда отсутствуют записи в таблице ошибок ПЛК.
%SC0011	IO_FLT	Устанавливается при возникновении любой ошибки, которая заносится в таблицу ошибок В/В. Сбрасывается, когда отсутствуют записи в таблице ошибок В/В.
%SC0012	SY_PRES	Удерживается активным до тех пор, пока в таблице ошибок ПЛК есть хотя бы одна запись. Сбрасывается, когда отсутствуют записи в таблице ошибок ПЛК.
%SC0013	IO_PRES	Удерживается активным до тех пор, пока в таблице ошибок В/В есть хоть одна запись Сбрасывается, когда отсутствуют записи в таблице ошибок I/O.
%SC0014	HRD_FLT	Устанавливается, когда возникает ошибка оборудования. Сбрасывается, когда отсутствуют записи в обеих таблицах ошибок.
%SC0015	SFT_FLT	Устанавливается когда возникает ошибка программного обеспечения. Сбрасывается, когда отсутствуют записи в обеих таблицах ошибок .

Как функции в программе обрабатывают числовые данные

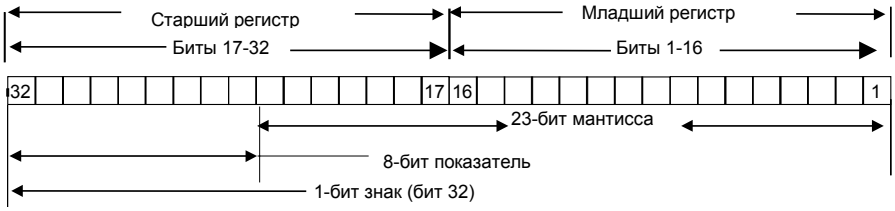
Независимо от того, где в памяти сохранены данные (в одном из типов памяти с битовым обращением или в памяти, ориентированной на хранение слов) прикладная программа может обрабатывать их как данные разных типов.

Тип	Название	описание	Формат данных
BIT	Бит	Бит данных - это самая маленькая единица информации. У бита есть два состояния: 1 или 0. Функции программатора используют термин BOOL для битовых данных.	
BYTE	Байт	Байт данных состоит из 8 битов. Действительный диапазон от 0 до 255 (от 0 до FF в шестнадцатеричном формате).	
WORD		Слово данных использует 16 последовательных бит памяти данных; Каждый бит находится в собственном двоичном состоянии (1 или 0). Допустимый диапазон от 0 до +65,535 (FFFF).	
BCD-4	Четырех значное двоично-десятичное число	Четырехзначные BCD числа используют 16 битовые ячейки памяти. Каждая BCD цифра использует 4 бита и может представлять числа от 0 до 9. 16 битное BCD число имеет диапазон значений от 0 до 9999.	
REAL	С плавающей точкой	Вещественные числа используют две последовательные 16 битовые ячейки памяти. Диапазон чисел, хранящихся в этом формате, от ± 1.401298E-45 to ± 3.402823E+38. Подробности на следующей странице.	
INT	Целое число со знаком	Целое число со знаком использует 16 битовую ячейку памяти. Целое число со знаком представлено в двоичной системе исчисления. Бит 16-знаковый бит, (0 = положительное, 1 = отрицательное). Диапазон от -32,768 до +32,767.	

DINT	Целое число двойной точности со знаком	Целые числа со знаком двойной точности используют две последовательные 16 битовые ячейки памяти. Они представлены в двоичной системе исчисления. Бит 32- знаковый бит, (0 = положительное, 1 = отрицательное). Диапазон от -2,147,483,648 до +2,147,483,867.	 Двоичные значения

Вещественные числа

REAL - тип данных, который может быть использован для некоторых математических и арифметических функций, - на самом деле является представлением числа с плавающей точкой. Числа с плавающей точкой сохраняются в формате IEEE единичной точности. Этот формат требует 32 бита, который занимают два (смежных) 16 битовых слова в памяти ПЛК.



На пример, если число с плавающей точкой занимает регистры %R0005 и %R0006, тогда %R0005 является младшим регистром, а %R0006 старшим регистром.

Диапазон чисел, которые можно хранить в этом формате, от ± 1.401298E–45 до ± 3.402823E+38, а также цифра ноль.

Ошибки в вещественных числах и Операциях

Переполнение происходит, когда числа большие чем 3.402823E+38 или меньшие чем -3.402823E+38 выводятся функцией имеющей формат вывода REAL. Вывод "ok" (норма) функции отключается; и результату присваивается значение плюс бесконечность (для чисел больших, чем 3.402823E+38) или минус бесконечность (для чисел меньших, чем –3.402823E+38). Можно проверить, где это происходит, проанализировав значения выходов “ok”.

POS_INF	= 7F800000h	– IEEE отображение плюс бесконечности в шестнадцатиричной системе.
NEG_INF	= FF800000h	– IEEE отображение минус бесконечности в шестнадцатиричной системе.

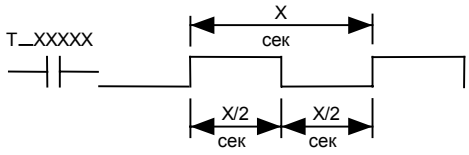
Если бесконечности, вызванные переполнением, используются в качестве операндов другими функциями, работающими с форматом REAL, они могут вызвать неопределенный результат. Этот результат называется NaN (Не число, Not a Number). Например, результат сложения плюс бесконечности и минус бесконечности неопределен. Когда функция ADD_REAL работает с операндами плюс бесконечность и минус бесконечность, получается результат NaN.

Периодические контакты времени

Существует четыре периодических контакта времени. Их можно использовать для передачи другим функциям программы периодических импульсов. Четыре периодических контакта времени имеют периодичность: 0.01 секунды, 0.1 секунды, 1.0 секунда и 1 минута.

Состояние этих контактов не изменяется во время выполнения цикла. Длительность импульса для этих контактов равна длительности паузы.

Ячейки контактов: T_10MS (0.01 секунды), T_100MS (0.1 секунды), T_SEC (1.0 секунда), и T_MIN (1 минута).
Приведенная ниже диаграмма показывает длительность импульсов и пауз для этих контактов.



Периодические контакты времени отображают определенные ячейки в памяти %S.

Глава
10

Система команд

В этом разделе приведена справочная информация по системе команд ПЛК VersaMax®:

Функции битовых операций Логическое "и" (AND), логическое "или" (OR), Логическое исключающее "или" (Exclusive OR), Логическое инвертирование (NOT) Циклический сдвиг влево/вправо (Rotate Right/Left), Сдвиг влево/вправо (Shift Right/Shift Left) Тест бита (Bit test) Установка бита (Bit Set), очистка бита (Bit Clear) Маскированное сравнивание (Masked Compare) Позиция бита (Bit Position) Одноконтактный командоаппарат (Bit Sequencer)	**Математические и арифметические функции** Сложение (Add), Вычитание (Subtract), Умножение (Multiply), Деление (Divide), Деление по модулю (Modulo Division), Масштабирование (Scaling), Квадратный корень (Square Root), Тригонометрические функции (Trigonometric Functions), Логарфмические функции и функции экспоненты (Logarithmic/Exponental Functions), Преобразование Радиан/Градусов (Convert Radians/Degrees)
Функции Управления Выполнить B/B (Do I/O) Вызов (Call) Временное окончание логики (End) Коментарий (Comment) Master Control Relay (принудительное выполнение) Имитатор командоаппарата (Drum Sequencer) Запрос на обслуживание (Service Request) (подробности в главе 11) ПИД (PID) (подробности в главе 14)	**Функции сравнения** Равно (Equal) Не равно (Not Equal) Больше (Greater Than) Меньше (Less Than) Больше или равно (Greater or Equal) Меньше или равно (Less or Equal) Диапазон (Range)

<table>
<tr>
<td rowspan="2">Функции пересылки данных
Пересылка (Move)
Пересылка блока (Block Move)
Очистка блока (Block Clear)
Сдвиг регистра (Shift Register)
Запрос связи (Communication Request)</td>
<td>Функции реле
Контакты (Contacts)
Катушки (Coils)</td>
</tr>
<tr>
<td>Табличные функции
Пересылка массива (Array Move)
Поиск (Search)</td>
</tr>
<tr>
<td>Функции преобразования типов данных
Преобразовать BCD-4 (Convert to BCD-4)
Преобразовать в целое число со знаком (Convert to Signed Integer)
Преобразовать в целое число со знаком двойной точности (Convert to Double Precision Signed Integer)
Преобразовать в Real (Convert to Real)
Преобразовать Real в Word (Convert Real to Word)
Округлить вещественное число (Truncate Real Number)
ПИД (PID) (подробности в главе 14)</td>
<td>Функции таймеров и счетчиков
Периодические контакты времени (Time-tick Contacts)
Таймер задержки по включению со сбросом (On Delay Stopwatch Timer)
Таймер задержки по включению (On Delay Timer)
Таймер задержки по выключению (Off Delay Timer)
Инкрементный счетчик (Up Counter)
Декрементный счетчик (Down Counter)</td>
</tr>
</table>

Функции битовых операций

Функции битовых операций включают сравнение, логические операции и операции по пересылке битовых строк. Они включают в себя:

- Логическое "и"(AND)
- Логическое "или"(OR)
- Логическое исключающее "или"(Exclusive OR)
- Логическое инвертирование (NOT)
- Сдвиг влево/вправо (Shift Right/Shift Left)
- Циклический сдвиг влево/ вправо (Rotate Left/Right)
- Тест бита
- Установка бита (Bit Set), очистка бита (Bit Clear)
- Маскированное сравнение
- Позиция бита
- Одноконтактный командоаппарат

Длина данных для функций битовых операций

Логические функции и, или, исключающее или и инвертирование (AND, OR, exclusive OR, NOT) работают с одним словом данных. Остальные функции битовых операций могут обрабатывать до 256 слов.

Все функции битовых операций требуют данных типа WORD. Однако, они работают с данными как с битовыми строками, где бит 1 - наименее значимый бит. Последний бит - наиболее значимый (MSB). Например, если вы указали три слова данных, начиная с %R0100, они будут обработаны как 48 последовательных бит.

%R0100	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1 ← бит 1 (LSB)
%R0101	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17
%R0102	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33

↑

(MSB)

Не рекомендуется работа с совмещением диапазонов входных и выходных данных в функциях, обрабатывающих несколько строк, т.к. возможен непредсказуемый результат.

Функции битовых операций
Логическое "и" (AND), "или" (OR)

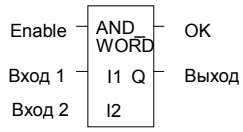
Во время каждого цикла, когда питание получено, функция логическое "и" (AND), или логическое "или" (OR) анализирует каждый бит в битовой строке I1 и соответствующий бит в битовой строке I2, начиная с наименее значимого бита в каждой последовательности. Можно выбрать строку длиной до 256 слов.

Логическое "и" (AND)

Если оба проанализированных функцией логического "и" (AND) бита установлены в 1, 1 ставится в соответствующее место в выходной строке Q. Если один или оба бита установлены в 0, то на это место в строке Q ставится 0. Функция логического "и" (AND) может быть использована для создания масок или экранов, пропускающих только определенные биты (биты, соответствующие 1 в маске), а все остальные биты устанавливающих в 0. Функция логическое "и" (AND) может быть использована для очистки участка памяти с пословным обращением с помощью операции логического умножения битов с другой битовой строкой содержащей только 0. Битовые строки I1 и I2 могут перекрываться.

Логическое "или" (OR)

Если один или оба бита проанализированных функцией логического "или" (OR) установлены в 1, 1 ставится в соответствующее место в выходной строке Q. Если один или оба бита установлены в 0, то на это место в строке Q ставится 0. Функция логическое "или" (OR) может использоваться для комбинирования строк или для управления несколькими выходами с помощью одной простой логической структуры. Функция логическое "или" (OR) эквивалентна двум параллельным контактам реле, умноженным на количество бит в строке. Она может быть использована для управления индикаторными лампами непосредственно от входов или для наложения условий мигания ламп.



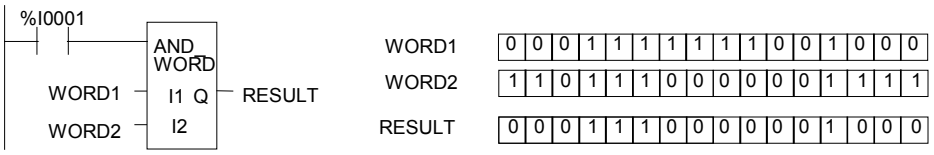
Функции битовых операций Логическое "и" (AND), "или" (OR)

Параметры функций логическое "и" (AND) и логическое "или" (OR)

Вход/ Выход	Варианты	Описание
Enabled (включено)	питание	Операция выполняется, когда подано питание.
I1	I, Q, M, T, S, G, R, AI, AQ, константы	Константа или адрес первого слова первой строки.
I2	I, Q, M, T, S, G, R, AI, AQ, константы	Константа или адрес первого слова второй строки
ok	питание, отсутствие питания	Выход OK включен, если на выход enable подано питание.
Q	I, Q, M, T, SA, SB, SC (не S), G, R, AI, AQ	Выход Q содержит результат операции.

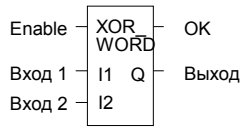
Пример функции логического "и" (AND)

В примере когда вход %I0001 установлен в 1, анализируются 16 битовые строки, представленные WORD1 и WORD2. Результат логического "и" (AND) помещен в выходную строку результата (RESULT)



Функция битовых операций
Исключающее "или" (Exclusive OR)

Функция исключающее "или" (Exclusive OR) сравнивает каждый бит в строке I1 с соответствующим битом в строке I2. Если биты отличаются, 1 устанавливается в соответствующее место в выходной битовой строке.



Во время каждого цикла, когда получено питание, функция исключающее "или" (Exclusive OR) анализирует каждый бит в битовой строке I1 и соответсвующий бит в битовой строке I2, начиная с младшего бита в каждой строке. Для каждых двух проверенных битов, если только один из них установлен в 1, 1 устанавливается в соответствующее место в битовой строке Q. Функция исключающего "или" (Exclusive OR) пропускает питание направо всякий раз, когда она его получает.

Если битовая строка I2 и выходная битовая строка Q начинаются по одному и тому же адресу, 1, установленная в битовой строке I1, будет переключать соответствующий бит в битовой строке I2 из 0 в 1 и обратно, изменяя состояние с каждым циклом до тех пор пока получает питание. Более длинные циклы могут быть запрограммированы подачей импульсов на вход питания. Частота импульсов должна быть в два раза выше требуемой частоты включений. Длительность импульса питания должна равняться одному циклу ПЛК (используйте катушку перехода или самосбрасывающийся таймер)

Функция исключающее "или" (Exclusive OR) полезна для быстрого сравнивания двух битовых строк или для мигания группы битов со скоростью одного включения в два цикла.

Функция битовых операций Исключающее "или" (Exclusive OR)

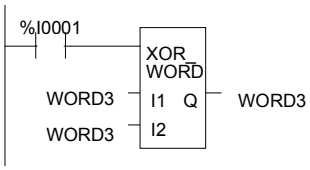
Параметры функции исключающее "или" (Exclusive OR)

Вход/ Выход	Варианты	Описание
enable (включить)	питание	Операция выполняется, когда подано питание.
I1	I, Q, M, T, S, G, R, AI, AQ, константы	Константа или адрес первого слова для проведения операции исключающего "или"
I2	I, Q, M, T, S, G, R, AI, AQ, константы	Константа или адрес второго слово для проведения операции исключающего "или".
ok	питание, отсутствие питания	Выход OK включен, если на вход enable подано питание.
Q	I, Q, M, T, SA, SB, SC (не S), G, R, AI, AQ	Выход Q содержит результат операции.

Функция битовых операций Исключающее "или" (Exclusive OR)

Пример

В этом примере всякий раз, когда %I0001 устанавливается в 1, битовая строка с именем WORD3 очищается (все биты устанавливаются в 0).

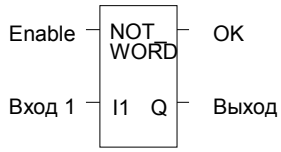


I1 (WORD3)	0	0	0	1	1	1	1	1	1	1	0	0	1	0	0	0
I2 (WORD3)	0	0	0	1	1	1	1	1	1	1	0	0	1	0	0	0
Q (WORD3)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Функции битовых операций
Логическое инвертирование (NOT)

Функция логического инвертирования (NOT) устанавливает состояние для каждого бита в выходной битовой строке Q в состояние, противоположное соответствующим битам в битовой строке I1.

В каждом цикле, когда подается питание, все биты изменяются, в результате чего выходная строка Q становится логическим дополнением I1. Функция пропускает питание направо всякий раз, когда она его получает. Длина строки может быть до 256 слов.

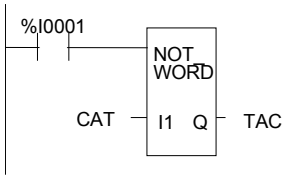


Параметры функции логического инвертирования (NOT)

Вход/ выход	Варианты	Описание
enable	питание	Операция выполняется, когда подано питание.
I1	I, Q, M, T, S, G, R, AI, AQ, константы	Константа или адрес инвертируемого слова.
ok	питание, отсутствие питания	Выход OK включен, если на вход enable подано питание.
Q	I, Q, M, T, SA, SB, SC (не S), G, R, AI, AQ	Выход Q содержит результат операции.

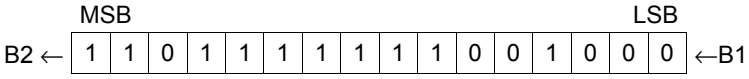
Пример

В этом примере всякий раз, когда %I0001 устанавливается в 1, битовая строка с именем TAC формируется как инверсия битовой строки CAT.

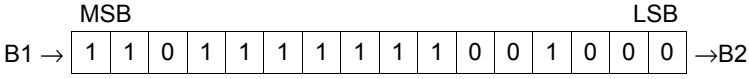


Функции битовых операций
Сдвиг битов влево/вправо (Shift Bits Left/Right)

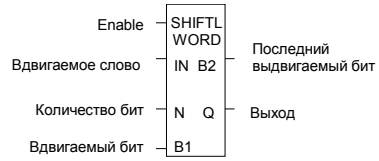
Функция сдвига влево (Shift Left) смещает все биты в слове или в группе слов влево на указанное количество позиций. Когда происходит сдвиг, указанное количество бит смещается влево из выходной строки наружу. По мере того, как биты вытесняются наружу с левого края строки, такое же количество бит добавляется с правого края.



Функция сдвига вправо (Shift Right) смещает все биты в слове или в группе слов на указанное количество позиций. Когда происходит сдвиг, указанное количество бит смещается вправо из выходной строки наружу. По мере того, как биты вытесняются наружу с правого края строки, такое же количество бит добавляется с левого края.



Для каждой функции можно выбрать длину строки от 1 до 256 слов.



Если задан сдвиг количества бит (N), большего, чем количество бит в массиве *16, массив Q заполняется копиями входного бита (B1), входной бит копируется на выход (B2). Если задано нулевое смещение, сдвиг не выполняется, входной массив копируется в выходной массив; входной бит (B1) копируется на выход (B2).

Биты, подставляемые в начало строки, указываются на входе B1. Если сдвигается больше одного бита, каждому биту присваивается одно и тоже значение (0 или 1). Это может быть:

- Логический выход другой функции в программе.
- Все 1. Для этого используйте на входе B1 специальную ячейку ALW_ON .
- Все 0. Для этого используйте на входе B1 специальную ячейку ALW_OFF .

Функции битовых операций
Сдвиг битов влево/вправо (Shift Bits Left/Right)

Функция пропускает питание направо, если для сдвига указано ненулевое количество бит. На выходе Q находится копия входной строки со сдвигом. Если вы хотите реализовать сдвиг во входной строке, выходной параметр Q должен занимать то же самое место в памяти, что и входной параметр IN. Вся строка записывается со сдвигом в каждом цикле, когда поступает питание. На выход B2 подается последний вытесненный наружу бит. Например, если осуществляется сдвиг на 4 бита, B2 будет четвертым вытесненным битом.

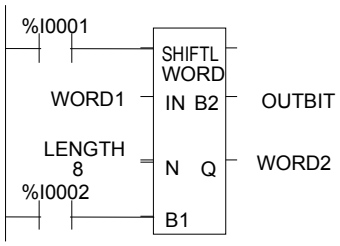
Функции битовых операций
Сдвиг бит влево/вправо (Shift bits Left/Right)

Параметры функций сдвига влево/вправо (Shift bits Left/Right)

Вход/ выход	Варианты	Описание
enable	питание	Операция выполняется, когда подано питание.
IN	I, Q, M, T, S, G, R, AI, AQ	IN содержит первое слово исходной строки.
N	I, Q, M, T, G, R, AI, AQ, константы	N содержит количество позиций (бит) на которые требуется сдвиг.
B1	питание	B1 содержит значение, подставляемое в массив.
B2	питание, отсутствие питания	B2 содержит значение последнего вытесненного из массива бита.
Q	I, Q, M, T, SA, SB, SC, G, R, AI, AQ	Вывод Q содержит первое слово массива со сдвигом.

Пример

В этом примере всякий раз, когда вход %I0001 устанавливается в 1, в выходную строку с именем WORD2 заносится копия входной строки WORD1. Выходная последовательность сдвигается влево на 8 бит, как указано на входе LENGTH. Освобождающимся в результате операции битам выходной строки присваивается значение %I0002.



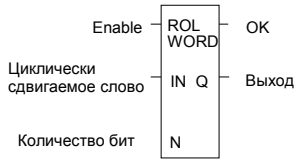
Функции битовых операций Циклический сдвиг влево/вправо (Rotate bits Left/Right)

Функция циклического сдвига влево (Rotate Left) смещает по кругу все биты в строке влево на заданное количество мест. Когда производится сдвиг, заданное количество бит во входной строке сдвигается из нее наружу слева и подставляется обратно в нее справа.

Функция циклического сдвига вправо (Rotate Right) смещает по кругу вправо биты в строке на заданное количество мест. Когда производится сдвиг, заданное количество бит во входной строке сдвигается из нее наружу справа и подставляется обратно в нее слева.

Для каждой функции можно выбрать длину строки от 1 до 256 слов. Количество мест для сдвига должно быть больше чем 0 и меньше чем количество бит в строке.

Функция циклического сдвига пропускает питание направо, если количество мест для сдвига не превышает длину строки и не отрицательно. Результат помещается в выходную строку Q. Если вы хотите реализовать циклический сдвиг во входной строке, выходной параметр Q должен использовать ту же самую ячейку памяти, что и входной параметр IN. Вся строка со сдвигом записывается при каждом цикле, когда поступает питание.



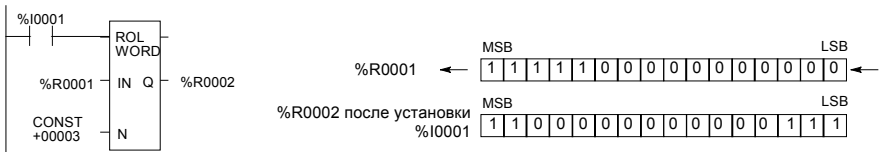
Параметры функций циклического сдвига влево/вправо (Rotate bits Left/Right)

Вход/ выход	Варианты	Описание
enable	питание	Операция выполняется, когда подано питание.
IN	I, Q, M, T, S, G, R, AI, AQ	IN содержит первое слово для циклического сдвига.
N	I, Q, M, T, G, R, AI, AQ, константы	N содержит количество мест в массиве для сдвига.
ok	питание, отсутствие питания	Выход OK включен, когда подано питание и величина сдвига не превышает размера массива.
Q	I, Q, M, T, SA, SB, SC, G, R, AI, AQ	Вывод Q содержит первое слово массива с циклическим сдвигом.

Функции битовых операций
Циклический сдвиг влево/вправо
(Rotate bits Left/Right)

Пример

В этом примере всякий раз, когда выход %I0001 устанавливается в 1, входная строка по адресу %R0001 циклически сдвигается на 3 бита. Результат помещается в %R0002. Входная битовая строка %R0001 не изменяется функцией. Если те же самые ячейки использовать для IN и Q, циклический сдвиг произойдет в исходной строке.

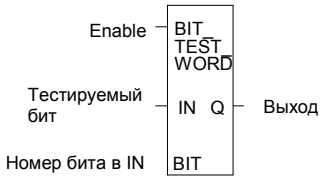


Функции битовых операций
Тест бита (Bit Test)

Функция тест бита (Bit Test) тестирует бит в строке для определения его значения в данный момент (1 или 0). Результат теста помещается в выход Q.

В каждом цикле, когда поступает питание, функция тест бита переводит выход Q в то же состояние, в котором находится указанный бит. Если номер бита указывается регистром, а не константой, один и тот же функциональный блок может проверять разные биты в последующих циклах. Если значение BIT находится за пределами диапазона (1 ≤ BIT ≤ (16 * длина)), Q устанавливается в 0.

Можно выбрать длину строки от 1 до 256 слов.



Параметры функции теста битов (Bit Test)

Вход/ выход	Варианты	Описание
enable	Питание	Операция выполняется, когда подано питание.
IN	I, Q, M, T, S, G, R, AI, AQ	IN содержит первое слово данных для работы функции.
Бит	I, Q, M, T, G, R, AI, AQ, константы	BIT содержит номер бита IN для тестирования. Допустимый диапазон (1 ≤ BIT ≤ (16 * длинна)).
Q	питание, отсутствие питания	Питание подается на вывод Q, если протестированный бит находится в 1.

Функции битовых операций
Тест бита (Bit Test)

Пример

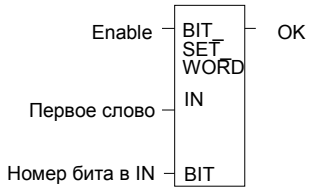
В этом примере всякий раз, когда вход %I0001 устанавливается в 1, тестируется бит по адресу PICKBIT. Бит является частью строки PRD_CDE. Если он установлен в 1, выход Q пропускает питание и катушка %Q0001 включается.

Функции битовых операций
Установка и очистка бита (Bit Set/Clear)

Функция установки бита (Bit Set) устанавливает бит в строке в 1. Функция очистки бита (Bit Clear) устанавливает бит в строке в 0.

В каждом цикле, когда поступает питание, функция устанавливает значение указанному биту. Если для указания номера бита используется переменная (регистр), а не константа, один и тот же функциональный блок может задавать разные биты в последующих циклах.

Можно выбрать длину строки от 1 до 256 слов. Функция пропускает питание направо, если значение BIT не выходит за пределы диапазона (1 ≤ BIT ≤ (16 * длина)). В противном случае OK устанавливается в 0.



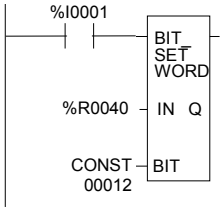
Параметры функции установки и очистки бита (Bit Set/Clear)

Вход/ выход	Варианты	Описание
enable	питание	Операция выполняется, когда подано питание.
IN	I, Q, M, T, SA, SB, SC, G, R, AI, AQ	IN содержит первое слово данных для работы функции.
Бит	I, Q, M, T, G, R, AI, AQ, константы	BIT содержит номер бита IN который должен быть установлен или очищен. Допустимый диапазон (1 ≤ BIT ≤ (16 * длинна)).
ok	питание, отсутствие питания	Питание на выход OK подается, когда значение на входе BIT находится в допустимом диапазоне и подано питание на вход enable.

Функции битовых операций
Установка и очистка бита (Bit Set/Clear)

Пример

В этом примере всякий раз, когда вход %I0001 устанавливается в 1, бит 12 строки, начинающейся с ячейки %R0040, устанавливается в 1.



Функции битовых операций
Маскированное сравнение (Masked Compare)

Функция маскированного сравнения (Masked Compare) сравнивает содержание двух битовых строк. Она предоставляет возможность маскировать выбранные биты. Входная битовая строка 1 может содержать состояния выходов, таких, как соленоиды и пускатели двигателей. Входная битовая строка 2 может содержать состояния входов, обеспечивающих обратную связь, таких, как концевые выключатели или контакты.

Когда функция получает питание, она начинает сравнивать биты первой битовой строки с соответствующими битами второй битовой строки. Сравнение продолжается или до обнаружения несовпадения, или до окончания строки.

Вход BIT сохраняет номер бита, с которого должно начаться следующее сравнение (*0* обозначает первый бит в строке). Выход BN сохраняет номер бита, с которого началось последнее сравнение (*1* означает первый бит в строке). Использование одинаковых ячеек для BIT и BN начинает сравнение со следующей позиции после несовпадающей позиции; или, если сравнение всех битов прошло успешно перед следующим запуском функционального блока, сравнение начинается сначала.

Если вы хотите, чтобы следующее сравнение началось с какого-нибудь другого места в строке, то вам надо ввести разные адреса для BIT и BN. Если значение, установленное для BIT, находится за пределами строки, то BIT устанавливается в 0 перед началом следующего сравнения.

Функции битовых операций
Маскированное сравнение (Masked Compare)

Параметры функции маскированного сравнения (Masked Compare)

Вход/ выход	Варианты	Описание
enable	питание	Разрешающая логика для функции.
I1	R, AI, AQ Для WORD только: I, Q, M, T, S, G	Адрес первой битовой строки для сравнения.
I2	R, AI, AQ Для WORD только: I, Q, M, T, S, G	Адрес второй битовой строки для сравнения.
M	R, AI, AQ Для WORD только: I, Q, M, T, SS, SB, SC, G	Адрес битовой маски строки.
Бит	I, Q, M, T, S, G, R, AI, AQ, константа	Адрес номера бита начала следующего сравнения.
MC	питание, отсутствие питания	Пользовательская логика для определения возникшего несовпадения.
Q	R, AI, AQ Для WORD только: I, Q, M, T, SA, SB, SC, G	Выходная копия битовой строки маски (M).
BN	I, Q, M, T, S, G, R, AI, AQ	Номер бита, где обнаружено последнее несовпадение.
Длина	константа	Количество слов в битовой строке. Максимальное количество 4095 для WORD и 2047 для DWORD.

Функции битовых операций
Маскированное сравнение (Masked Compare)

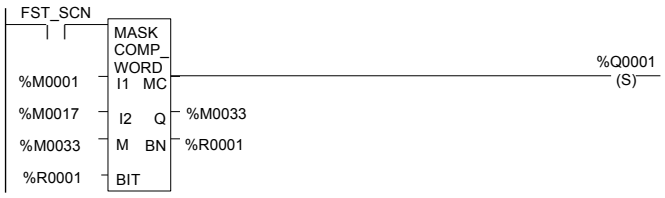
Принцип работы маскированного сравнения (Masked Compare)

Если все соответствующие биты в строках I1 и I2 совпадают, то функция устанавливает выход MC “miscompare (несовпадения)” в 0 и BN в самый большой номер бита во входных строках. После чего сравнение останавливается. При следующем вызове Masked Compare Word, он устанавливается в 0. Если два сравниваемых бита неодинаковы, то функция проверяет бит с соответствующим номером в строке M (маска). Если бит маски установлен в *1*, то сравнение продолжается до обнаружения другого несовпадения или до окончания входных строк. Если обнаружено несовпадение, и соответствующий бит маски установлен в 0, то функция выполняет следующие действия:

1. Устанавливает соответствующий бит маски в строке M в 1.
2. Устанавливает выход несовпадения (MC) в 1.
3. Обновляет выходную строку Q так, чтобы она совпадала с новым содержимым строки маски M.
4. Устанавливает на выходе номера бита (BN) номер несовпадающего бита.
5. Останавливает сравнение.

Пример

В этом примере выполняется функция маскированного сравнения (Masked Compare). Она сравнивает %M0001–16 с %M0017–32. Маска содержится в %M0033–48. Значение в %R0001 определяет в двух входных строках позицию бита, с которого начнется сравнение.



Перед выполнением функционального блока, содержимое указанных выше ячеек:

(I1) – %M0001 = 6C6Ch =	0	1	1	0	1	1	0	0	0	1	1	0	1	1	0	0
(I2) – %M0017 = 606Fh =	0	1	1	0	1	1	0	1	0	1	1	0	1	1	1	1
(M/Q) – %M0033 = 000Fh =	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1

(BIT/BN) – %R0001 = 0
(MC) – %Q0001 = OFF

Функции битовых операций
Маскированное сравнение (Masked Compare)

После выполнения функционального блока, содержимое указанных выше ячеек будет:

(I1) – %M0001	= (same)	=	0	1	1	0	1	1	0	0	0	1	1	0	1	1	0	0
(I2) – %M0017	= (same)	=	0	1	1	0	1	1	0	1	0	1	1	0	1	1	1	1
(M/Q) – %M0033		=	0	0	0	0	0	0	0	1	0	0	0	0	1	1	1	1
(BIT/BN) – %R0001		= 8																
(MC) – %Q0001		= ON																

В этом примере контакт %T1 и катушка %M100 вызывают одно и только одно выполнение функции, иначе функция будет повторяться с непредсказуемым результатом.

Функции битовых операций
Позиция бита (Bit Position)

Функция позиции бита (Bit Position) .

Во время каждого цикла, когда получено питание, функция проверяет битовую строку, начиная с IN. Функция заканчивает проверку, если был обнаружен бит, равный 1, или вся последовательность была опрошена.

POS устанавливается на место первого ненулевого бита в строке; POS устанавливается в 0, если таковых найдено не было.

Строка может состоять от 1 до 256 слов. Функция пропускает питание направо всегда, когда на вход enable подано питание.

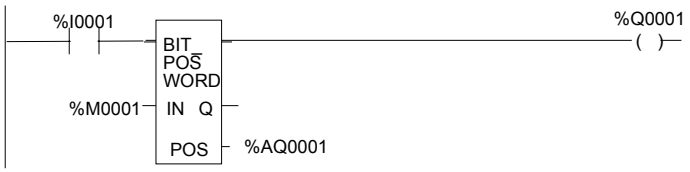
Параметры функции позиции бита (Bit Position)

Вход/ выход	Варианты	Описание
enable	питание	Операция выполняется, когда подано питание.
IN	I, Q, M, T, S, G, R, AI, AQ	IN содержит первое слово данных для работы функции.
ok	питание , отсутствие питания	Выход OK включен, если на выход enable подано питание.
POS	I, Q, M, T, G, R, AI, AQ	Позиция первого ненулевого бита в строке, или 0, если таковых найдено не было .

Функции битовых операций
Позиция бита (Bit Position)

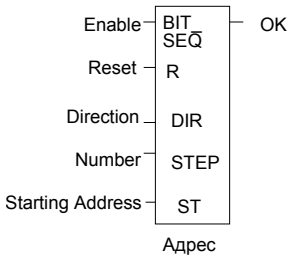
Пример

В этом примере всякий раз, когда %I0001 устанавливается в 1, битовая строка начиная с %M0001 проверяется до тех пор, пока не будет обнаружен бит, равный 1. Катушка %Q0001 включена. Если бит, равный 1, обнаружен, то его положение в строке записывается в %AQ001. Если %I0001 установлен в 1, бит %M0001 установлен в 0, и бит %M0002 установлен в 1, то значение 2 записывается в %AQ001.



Функции битовых операций
Одноконтактный командоаппарат (Bit Sequencer)

Функция одноконтактного командоаппарата Bit Sequencer выполняет смещение порядка битов в массиве.



Работа функции зависит от предыдущего значения параметра EN:

R Текущий шаг	EN Предыдущий шаг	EN Текущий шаг	Выполнение Bit Sequencer
0	0	0	Не выполняется.
0	0	1	+/- 1.
0	1	0	Не выполняется.
0	1	1	Не выполняется.
1	1/0	1/0	Возвращается в исходное положение.

Вход сброса (R) игнорирует вход разрешения Enable (EN) и всегда возвращает командоаппарат в исходное положение. Когда вход R активен, текущему номеру шага устанавливается значение параметра номера шага. Если номер шага отсутствует, шаг устанавливается в 1. Все биты командоаппарата устанавливаются в 0, кроме бита, на который указывает текущий шаг, который устанавливается в 1.

Когда вход Enable установлен в 1, а вход Reset - в 0, бит, указанный текущим номером шага, очищается. Текущий номер шага увеличивается/уменьшается на 1 в зависимости от значения параметра направления. Затем бит, указанный номером нового шага, устанавливается в 1.

Параметр ST не является обязательным. Если он не используется, функция одноконтактного командоаппарата (Bit Seqencer) работает, как описано выше, кроме того, что биты не устанавливаются и не очищаются. Функция просто циклически изменяет номер текущего шага в допустимом диапазоне.

Функции битовых операций
Одноконтактный командоаппарат (Bit Sequencer)

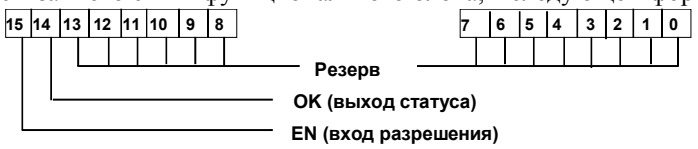
Память требуемая Одноконтактному командоаппарату (Bit Sequencer)

Каждый одноконтактный командоаппарат использует три слова (регистра) памяти %R для хранения информации:

слово 1	номер текущего шага
слово 2	длина последовательности (в битах)
слово 3	управляющее слово

Функции битовых операций
Одноконтактный командоаппарат (Bit Sequencer)

Слово 3 (управляющее слово) сохраняет состояние логических входов и выходов связанного с ним функционального блока, в следующем формате:



Параметры функции Одноконтактный командоаппарат (Bit Sequencer)

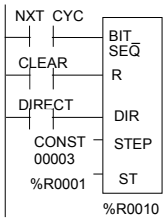
Вход/ выход	Варианты	Описание
address	R	Адрес хранения текущего шага, длины, и последнего статуса OK и Enable одноконтактного командоаппарата.
enable	питание	Когда на вход enable подано питание, при условии, что в предыдущем цикле питание на него не подавалось и на вход R питание не подано, выполняется сдвиг битовой строки.
R	питание	Когда на вход R подано питание, номеру шага одноконтактного командоаппарата присваивается значение STEP (по умолчанию = 1), и одноконтактный командоаппарат заполняется нулями, кроме бита номера текущего шага.
DIR	питание	Когда на вход DIR подано питание, перед сдвигом увеличивается номер шага одноконтактного командоаппарата. В противном случае, он уменьшается.
STEP	I, Q, M, T, G, R, AI, AQ, константы, отсутствие питания	Если на вход R подано питание, номер шага устанавливается в это значение.
ST	I, Q, M, T, SA, SB, SC, G, R, AI, AQ, отсутствие питания	ST содержит первое слово одноконтактного командоаппарата. Необязательно.
ok	питание, отсутствие питания	Выход OK включен, когда на вход enable подано питание.

Функции битовых операций
Одноконтактный командоаппарат (Bit Sequencer)

Пример

Например, одноконтактный командоаппарат (Bit Sequencer) работает с памятью регистров %R0001. Его статические данные хранятся в регистрах %R0010–12. Когда вход CLEAR установлен в 1, командоаппарат устанавливается в исходное состояние, и номер текущего шага устанавливается в значение 3. Первые восемь бит регистра %R0001 устанавливаются в 0.

Когда вход NXT_CYC установлен в 1, а вход CLEAR установлен в 0, бит номера шага 3 очищается, и устанавливается бит номера шага 2 или 4 (в зависимости от состояния входа DIR).



Функции управления

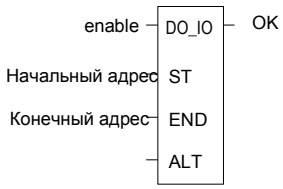
В этом разделе описаны функции управления, с помощью которых можно ограничивать выполнение программы и изменять выполнение прикладной программы ЦПУ.

- Выполнить ввод-вывод (В/В): DO IO
- Перейти к блоку подпрограммы: CALL
- Временное окончание программы: END
- Принудительно выполнить группу звеньев логики без подачи питания: MCR
- Перейти в указанное место программы: JUMP, LABEL
- Поместить текстовое пояснение в логику программы: COMMENT
- Предопределенная схема включения/выключения для установки шестнадцати дискретных выводов аналогично механическому командоаппарату.

Более сложные функции управления - системный запрос (Service Request) и алгоритмы ПИД (PID) описаны в других главах этой инструкции пользователя.

Функции управления
Выполнить B/B (Do I/O)

Функция Выполнить B/B (Do I/O) обновляет входы или выходы в течение одного цикла во время работы программы. Функцию выполнить B/B (Do I/O) можно также использовать для обновления выбранных входов/выходов во время выполнения программы в добавление к обычному опросу B/B. B/B обслуживается по модульно, по возрастанию; при необходимости ПЛК корректирует значения адресов во время выполнения функции.



Выполнение функции продолжается до тех пор пока все входы не предоставили данные и не были обслужены все выходы модулей B/B в выбранном диапазоне. Затем выполнение программы переходит к следующей функции.

Если диапазон ячеек включает дополнительный модуль, все входные данные (%I и %AI) или все выходные данные (%Q и %AQ) этого модуля будут опрошены. Параметр ALT игнорируется во время опроса интеллектуальных модулей B/B или интерфейса Ethernet.

Функция пропускает питание направо всегда, когда получено питание, за исключением:

- Не все ссылки указанного типа присутствуют в выбранном диапазоне.
- ЦПУ не может корректно обработать временный список B/B, созданный функцией.
- В указанном диапазоне имеются модули, с которыми связана ошибка “Loss of I/O".

Функции управления
Выполнить В/В (Do I/O)

Параметры функции Выполнить В/В (Do I/O)

Вход/ выход	Варианты	Описание
enable	питание	Ограниченный опрос входов/выходов выполняется, когда подано питание.
ST	I, Q, AI, AQ	Начальный адрес обслуживаемых входов/выходов.
END	I, Q, AI, AQ	Конечный адрес обслуживаемых входов/выходов.
ALT	I, Q, M, T, G, R, AI, AQ, константа, отсутствие питания	При опросе входов параметр ALT указывает, по какому адресу сохранять значения опрошенных входных точек/слов. При опросе выходов параметр ALT указывает с какого адреса брать значения выходных точек/слов для пересылки их модулям В/В.
ok	питание, отсутствие питания	Выход OK включен, если опрос выполнен без ошибок.

Функции управления
Выполнить B/B (Do I/O)

Выполнить B/B (Do I/O) для Входов

Если указаны входные ячейки, то, когда функция получает питание, ПЛК опрашивает входные каналы с начального адреса (ST) до конечного адреса (END). Если указана ячейка для параметра ALT, копии новых входных значений помещаются в память, начиная с этой ячейки, а действительные значения не обновляются. Параметр ALT должен быть той же размерности, что и опрашиваемые ячейки. Если для ST и END используются дискретные ячейки, параметр ALT тоже должен быть дискретным. Если для параметра ALT ячейки не указаны, обновляются действительные входные значения. Это предоставляет возможность опрашивать входы несколько раз во время выполнения программы в цикле ЦПУ.

Пример функции выполнить B/B (Do I/O) для входов:

В этом примере при получении функцией питания , ПЛК опрашивает ячейки %I0001-64 и включается ячейка %Q0001 . Копии опрошенных входов помещаются во внутреннюю память %M0001-64. Т.к. указана ячейка для параметра ALT , действительные входы не обновляются. Это позволяет сравнивать текущие значения входов с их значениями в начале цикла.

Выполнить B/B (Do I/O) для выходов

Если указаны выходные ячейки, то, когда функция получает питание, ПЛК записывает в выходные модули последние выходные значения с начальной ячейки ST до ячейки END. Если выходные данные нужно записать из внутренней памяти, а не из %Q или %AQ, в параметре ALT может быть указана начальная ячейка.

Функции управления
Выполнить B/B (Do I/O)

Пример Выполнить B/B (Do I/O) для выходов:

В данном примере, когда функция получает питание, ПЛК записывает значения из ячеек %R0001-0004 в аналоговые выходные каналы %AQ001-004 и включается выход %Q0001. Т.к. введена ячейка для параметра ALT, значения из ячеек памяти %AQ001-004 не записываются в выходные модули.

Если бы для параметра ALT ячейка не была указана, ПЛК записал бы значения ячеек %AQ001-004 в выходные аналоговые каналы.

Функции управления
Выполнить B/B (Do I/O)

Выполнить B/B (Do I/O) для одного модуля (Расширенный Do I/O))

Функцию Выполнить B/B (Do I/O) можно использовать для одного выходного дискретного модуля, расположенного в главном ПЛК. Функция выполняется намного быстрее, когда данные записываются и считываются только для одного модуля.

Модуль для записи/чтения указывается в параметре ALT. Например, константа 2 для этого параметра указывает ЦПУ, что оно должно выполнить функциональный блок Выполнить B/B (Do I/O) для модуля, расположенного в слоте 2. Начальные и конечные ячейки должны быть или %I, или %Q. Эти ячейки указывают первую и последнюю ячейки, сконфигурированные для модуля.

Пример выполнить B/B (Do I/O) для одного модуля

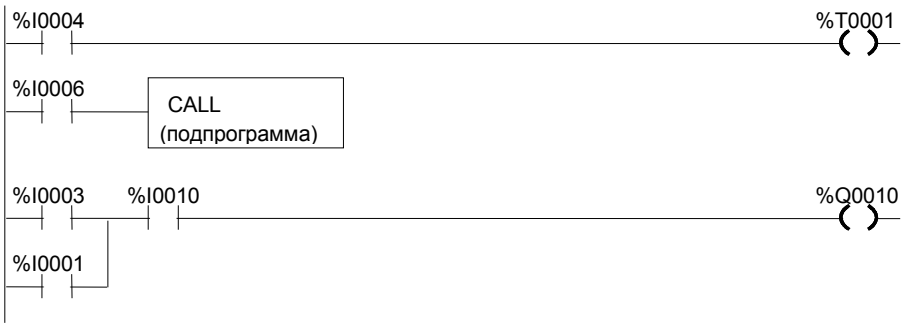
В этом примере функция Выполнить B/B (Do I/O) выполняется только для 16-канального входного модуля, который сконфигурирован в слоте 2 с ячейками памяти от %I0001 до %I0016.

Функции управления
Вызов (Call)

Функция Вызов (Call) перемещает выполнение программы в указанную подпрограмму.

Когда функция Вызова (Call) получает питание, цикл немедленно перемещается в блок подпрограммы и выполняет его. После выполнения блока подпрограммы управление возвращается в место в логике сразу после команды вызова (Call).

Пример



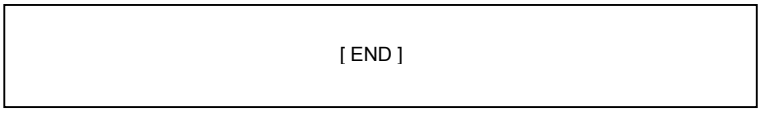
Функции управления
Окончание логики (End of Logic)

Функция Окончания Логики (End of Logic) обеспечивает временное окончание логики. Программа выполняется от первого до последнего звена или до функции Окончания Логики (End of Logic)в зависимости от того, что встретится раньше.

Функция Окончания Логики (End of Logic) безусловно завершает выполнение программы. После функции окончания логики (End of Logic) в звене ничего не может находиться. После функции окончания логики (End of Logic) логика не выполняется, и управление переходит в начало программы в следующем цикле.

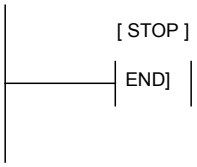
Функция Окончания Логики (End of Logic) полезна для целей отладки, т.к. она блокирует выполнение любой последующей логики.

В инструментальном программном обеспечении имеется маркер [END OF PROGRAM LOGIC] для указания окончания выполнения программы. Этот маркер используется, если функция окончания логики (End of Logic) не применяется в логике.



Пример

В данном примере функция окончания логики (End of Logic) установлена для завершения текущего цикла.



Функции управления Принудительное исполнение (MCR)/ окончание принудительного исполнения (End MCR)

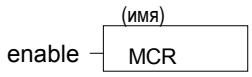
Все звенья между активной функцией принудительно исполнения (MCRN) и соответствующей ей функцией окончания принудительного исполнения (ENDMCRN) выполняются без подачи питания на катушки. Функция окончания принудительного исполнения (ENDMCRN), связанная с функцией принудительного исполнения, используется для возобновления нормального выполнения программы. В отличие от функций Перехода (Jump), функции принудительного исполнения могут действовать только вперед; функция окончания принудительного исполнения (ENDMCRN) должна находиться после соответствующей команды принудительного исполнения в программе.

Вложенная функция принудительного исполнения (MCRN)

Функция принудительного исполнения может быть полностью вложена в другую пару MCRN/ENDMCRN.

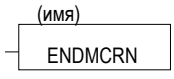
Несколько функций принудительного исполнения можно использовать с одной ENDMCRN.

У функции принудительного исполнения есть вход enable и имя. Это имя заново используется функцией окончания принудительного исполнения (ENDMCRN). Функция принудительного исполнения не имеет выходов; и после нее в звене ничего не может находиться.



Функциональные блоки в зоне действия функции принудительного исполнения *выполняются без питания, и катушки выключаются*.

Функция окончания принудительного исполнения (ENDMCRN) должна быть соединена с шиной питания, и перед ней в звене не может быть логики. Имя функции окончания принудительного исполнения (ENDMCRN) связывает ее с соответствующей функцией(ями) принудительного исполнения. Функция ENDMCR не имеет выходов, и после нее в звене ничего не может находиться.

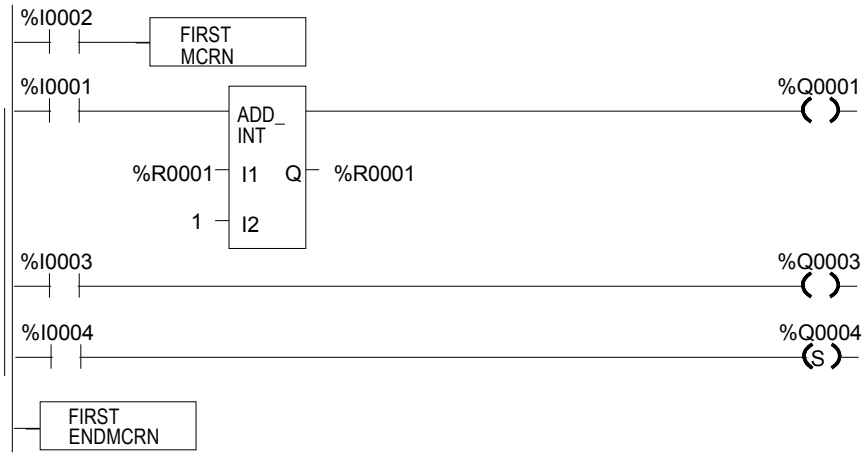


Функции управления принудительное исполнение (MCR)/ окончание принудительного исполнения (End MCR)

Пример функций принудительного исполнения (MCR) и окончания принудительного исполнения (ENDMCRN)

В этом примере при включении контакта %I0002, включается функция принудительного исполнения (MCR). Когда функция принудительного исполнения (MCR) включена - даже если контакт %I0001 включен - функциональный блок сложения выполняется без подачи питания (т.е., он не прибавляет 1 к %R0001), и катушка %Q0001 выключена.

Если контакты %I0003 и %I0004 включены, катушка %Q0003 выключена, а катушка %Q0004 остается включенной.



Функции управления
Переход (Jump), Метка (Label)

Вложенная команда перехода (Jump) позволяет пропускать часть логики программы. Выполнение программы продолжается с указанной метки (Label). Когда функция переход (Jump) активна, все катушки в области ее действия остаются в исходном состоянии. Это относится, в том числе, и к катушкам, связанным с таймерами, счетчиками, триггерами и реле.

Вложенная команда перехода (Jump) имеет вид----->>Метка 01, где Метка 01 является именем соответствующей вложенной инструкции Метка (Label).

Вложенную команду переход (Jump) можно поместить в любом месте программы.

Несколько вложенных команд переход (Jump) могут соответствовать одной вложенной команде Метка (Label). Вложенные команды Переход (Jump) могут совершать переходы вперед или назад.

В звене после команды Переход (Jump) не может ничего находиться. Питание переходит напрямую от команды к звену, помеченному меткой.

Предостережение

Для избежания бесконечного цикла с функциями перехода (Jump) назад и вперед, функция перехода назад должна иметь возможность становиться условной.

Метка (Label)

Команда Метка (Label) является целью команды Переход (Jump). Используйте команду Метка (Label) для возобновления нормального выполнения программы. В программе каждая метка (Label) должна иметь индивидуальное имя.

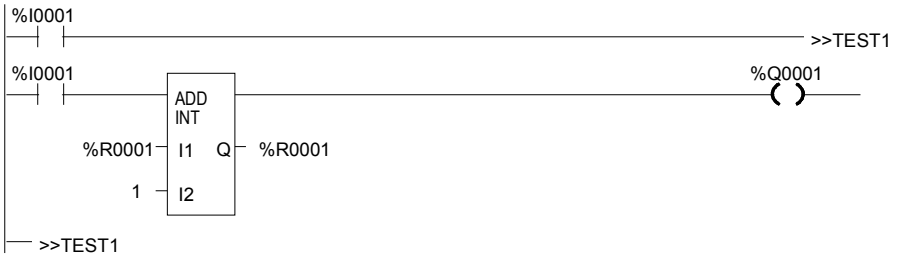
Команда Метка (Label) не имеет входов и выходов, и в звене не может находиться ничего ни перед, ни после команды Метка (Label).

Функции управления
Переход (Jump), Метка (Label)

Пример команд Метка (Label) и Переход (Jump)

В этом примере когда команда переход (Jump) TEST1 активна, питание передается на метку (Label) TEST1.

При использовании команды перехода (Jump) все функциональные блоки между командами перехода (Jump) и командой метки (Label) *не выполняются, и катушки остаются в исходном состоянии*. В этом примере когда вход %I0002 включен, происходит переход (Jump). Т.к. логика между командами перехода (Jump) и метки (Label) не выполняется, %Q0001 остается в исходном положении (если была включена, то и остается включенной; если была выключена, то и остается выключенной).



Функции управления
Комментарий (Comment)

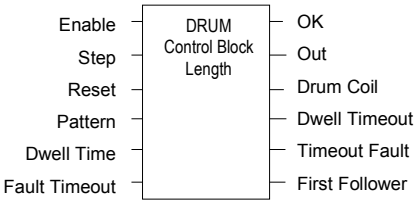
Функция Комментарий (Comment) используется для ввода в программу комментария (пояснения звена). Комментарий может содержать до 2048 символов текста. Более длинный текст можно включать в распечатки с помощью текстового файла аннотации.

Комментарий отображается в релейно-контактной логике как показано ниже:

```
(* COMMENT *)
```

Функции управления
Имитатор Командоаппарата (Drum Sequencer)

Функция Имитатор Командоаппарата (Drum Sequencer) является элементом программы с принципом действия как у механического командоаппарата. Имитатор командоаппарата (Drum Sequencer) пошагово перебирает набор потенциальных выходных битовых шаблонов и выбирает один, исходя из своих входных данных. Выбранное значение копируется в группу из 16 дискретных выходных ячеек.



При подаче питания на вход Enable имитатор командоапарата (Drum Sequencer) копирует содержимое выбранной ячейки в выходную ячейку OUT.

При подаче питания на вход Reset или на вход Step происходит выбор ячейки для копирования.

Вход Control Block является начальной ячейкой блока параметров функции имитатора командоаппарата (Drum Sequencer) и содержит информацию, используемую функцией.

Функции управления
Имитатор Командоаппарата (Drum Sequencer)

Параметры функции Имитатора Командоаппарата (Drum Sequencer)

Вход/ выход	Варианты	Описание
enable	питание	Вход Enable управляет выполнением функции.
Step	питание	Вход Step можно использовать для перехода вперед на один шаг. При подаче питания на вход Enable и при переходе входа Step из 0 в 1, имитатор командоаппарата (Drum Sequencer) перемещается на 1 шаг вперед. Когда вход Reset активен, функция игнорирует вход Step.
RESET (сброс)	питание	Вход Reset можно использовать для выбора определенного шага. Когда входы Enable и Reset получают питание, функция копирует значение начального положения (Preset Step) из управляющего блока в ячейку текущего шага (Active Step), также находящуюся в управляющем блоке. Затем функциональный блок копирует значение, хранящееся по адресу, на который указывает ячейка предустановленного шага (Preset Step), в ячейку Out. Когда вход Reset активен, функция игнорирует вход Step.
Pattern	R, AI, AQ	Начальный адрес массива слов, каждое из которых представляет собой одну ступень имитатора комаоаппарата (Drum Sequencer). Значение каждого слова представляет собой желаемую комбинацию выходов для определенного значения Active Step. Количество элементов массива равно длине входа.
Dwell Time	R, AI, AQ, отсутствие параметра	Этот необязательный входной массив слов содержит один элемент для каждого элемента в массиве шаблонов. Каждое значение в массиве представляет время задержки для соответствующего шага имитатора командоаппарата (Drum Sequencer). Когда время задержки для данного шага истекает, устанавливается бит тайм-аута задержки шага (Dwell Timeout). Если время задержки указано, имитатор командоаппарата не может перейти к следующему шагу до тех пор, пока не истечет время задержки.
Fault Timeout	R, AI, AQ, отсутствие параметра	Этот необязательный входной массив слов содержит один элемент для каждого элемента в массиве шаблонов. Каждое значение в массиве представляет собой ошибку тайм-аута для соответствующего шага имитатора командоаппарата (Drum Sequencer). При истечении тайм-аута ошибки устанавливается бит тайм-аута ошибки (Fault Timeout bit)
Control Block	R	Адрес начальной ячейки блока параметров функции. Длина управляющего блока (Control Block) пять слов. Более подробное описание содержания блока приведено ниже.
Длина	CONST	Значение от 1 до 128, указывающее количество шагов.
ok	питание, отсутствие питания	На выход OK подается питание, если вход Enable включен и не зафиксировано никаких ошибок. Если вход Enable выключен, выход OK будет тоже выключен.
OUT	I, Q, M, T, G, R, AI, AQ	Слово в памяти, содержащее элемент массива шаблонов, соответствующий текущему шагу (Active Step).
Drum Coil	I, Q, M, T, G, отсутствие питания	Эта необязательная битовая ячейка устанавливается, когда на функциональный блок подано питание, и текущий шаг не равен начальному положению.
Dwell Timeout	I, Q, M, T, G, отсутствие питания	Эта необязательная битовая ячейка устанавливается, если превышено время задержки данного шага.

Функции управления
Имитатор Командоаппарата (Drum Sequencer)

Параметры функции Имитатора Командоаппарата (Drum Sequencer) *(продолжение)*

Timeout Fault	I, Q, M, T, G, отсутствие питания	Эта необязательная битовая ячейка устанавливается, если имитатор командоаппарата находится на определенном шаге дольше, чем это разрешено уставкой тайм-аута.
First Follower	I, Q, M, T, G, отутствие питания	Этот необязательный битовый массив содержит один элемент для каждого шага имитатора командоаппарата (Drum Sequencer). В любой момент времени в массиве не может быть больше одного бита, установленного в 1, и этот бит соответствует значению данного шага.

Функции управления
Имитатор Командоаппарата (Drum Sequencer)

Блок параметров функции Имитатора Командоаппарата (Drum Sequencer)

Блок параметров (управляющий блок) функции имитатора командоаппарата (Drum Sequencer) содержит информацию, нужную для работы функции.

Адрес	Текущий шаг (Active Step)
адрес + 1	Начальное значение (Preset Step)
адрес + 2	Управление переходом (Step Control)
адрес + 3	Управление таймером (Timer Control)

Текущий шаг (Active Step) Значение текущего шага указывает на элемент в массиве шаблонов, выбранный для копирования в выходную ячейку памяти Out. Он используется в качестве указателя в массивах шаблона (Pattern), длины шага (Dwell Time), ошибки тайм-аута (Fault Timeout), и синхронизации (First Follower).

Начальное положение (Preset Step) Входное слово, которое копируется в выход Active Step, когда вход Reset включен.

Управление переходом (Step Control) Слово, используемое для обнаружения перехода входов Step и Enable из 0 в 1. Слово управления переходом (Step Control) зарезервировано для использования функциональным блоком; пользователь не должен вносить в него изменения.

Управление таймером (Timer Control) Два слова данных, которые содержат значения для работы таймера. Эти значения зарезервированы для использования функциональным блоком; пользователь не должен вносить в него изменения.

Примечания по использованию функции имитатора командоаппарата (Drum Sequencer)

1. Выходной бит тайм-аута задержки очищается, как только командоаппарат переходит на новый шаг. Это выполняется:
   - При переходе командоаппарата на новый шаг по изменению параметра текущего шага (Active Step) или в результате использования входа Step.

- Независимо от значения массива времени задержки, связанного с шагом (даже если оно равно 0).
- При инициализации параметра текущего шага (Active Step) во время первого цикла.

2. Для того, чтобы имитатор командоаппарата (Drum Sequencer) мог работать или пропускать через себя питание, должны быть инициализированы параметры текущего шага (Active Step) и начального положения (Preset Step) управляющего блока имитатора командоаппарата (Drum Sequencer). Даже если параметр текущего шага (Active Step) находится в допустимом диапазоне (между 1 и значением длины массива шаблона (Pattern)), а параметр начального положения (Preset Step) не используется, командоаппарат не будет работать, если значение параметра начального положения не находится в допустимом диапазоне.

Функции Пересылки Данных

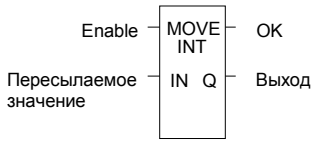
Функции пересылки данных системы команд обеспечивают основные операции пересылки данных.

- Пересылка данных (Move Data). Эта функция копирует данные как отдельные биты, поэтому новое место не обязательно должно иметь тот же тип данных.
- Пересылка блока (Block Move). Эта функция помещает константы в 7 указанных ячеек памяти.
- Очистка блока (Block Clear). Эта функция заполняет область памяти нулями.
- Сдвиг регистра (Shift Register). Эта функция сдвигает одно или несколько слов данных или бит данных с их первоначального адреса в указанную область памяти. Данные, уже находящиеся в данной области, выдвигаются из нее.
- Запрос связи (COMMREQ). Эта важная функция позволяет ЦПУ связываться с интеллектуальными модулями системы, например, с коммуникационными модулями. Основной формат функции запроса связи (COMMREQ) приведен в этой главе. Параметры, необходимые для программирования специфических коммуникационных задач, подробно рассмотрены в документации на каждый модуль.

Функции Пересылки Данных
Пересылка данных (Move Data)

Функция пересылки (Move) копирует данные как отдельные биты с одного места в другое. Так как данные копируются в битовом формате, новое место не обязательно должно быть того же типа данных, что и первоначальное.

Когда на функцию пересылки (Move) подается питание, она копирует данные из входного параметра IN в выходной параметр Q как биты. Если данные пересылаются из одной области дискретной памяти в другую (например, из памяти %I в память %T), информация о переходе, связанная с элементами дискретной памяти, обновляется, чтобы отобразить, вызвала ли операция пересылки (Move) изменение состояния элементов дискретной памяти. Входные данные не изменяются, если нет наложения областей источника и назначения.

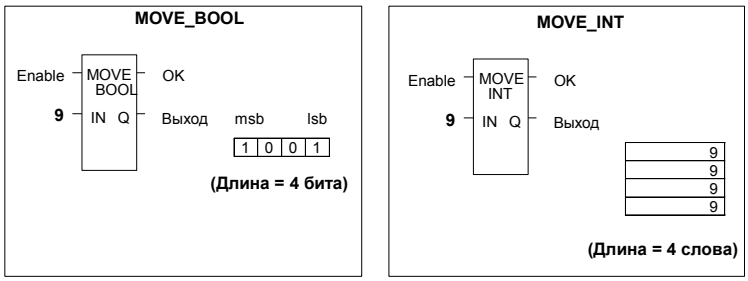


Обратите внимание, что если битовый массив данных, указанный в параметре Q, не включает всех битов в байте, биты перехода, связанные с этим байтом (не входящие в массив) очищаются, когда на функцию пересылки (Move) подается питание.

Вход IN может быть ячейкой пересылаемых данных или константой. Если указана константа, тогда значение константы помещается в область памяти, определенную выходным адресом. Например, если для входа IN указано значение константы 4, то 4 помещается в ячейку памяти, указанную в Q. Если длина больше 1и указана константа, то костанта помещается в ячейку памяти, указанную в Q и в ячейки памяти следующие за ней по всей указанной длине. Наложение параметров IN и Q не допускается.

Результат пересылки зависит от типа данных, выбранных для функции, как показано ниже. Например, если для IN указано значение константы 9 и длина 4, то 9 помещается в ячейку памяти битов, указанную Q, и три последующие ячейки:

Функции Пересылки Данных
Пересылка данных (Move Data)



Функция пропускает питание направо всякий раз, когда она его получает.

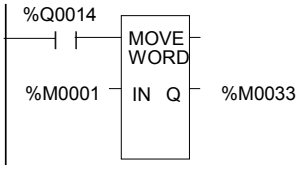
Функции Пересылки Данных
Пересылка данных (Move Data)

Параметры функции пересылки данных (Move Data)

Вход/ выход	Варианты	Описание
enable	питание	Пересылка выполняется, когда подано питание.
Длина		Количество бит, слов или двойных слов копируемых данных. Эта длина входного параметра IN. Длина может быть от 1 до 256 для всех типов, кроме BIT. Если IN - константа и Q битового типа, длина должна быть от 1 до 16. Если IN битового типа, длина должна быть от 1 до 256 бит.
IN	I, Q, M, T, G, R, AI, AQ, константы Только для данных типа бит или слово: S Для данных с плавающей точкой: R, AI, AQ	IN содержит пересылаемое значение. Для MOVE_BOOL могут использоваться любые дискретные ячейки, выравнивание по байтам не требуется. Однако, 16 бит, начинающиеся с указанной ячейки памяти, отображаются в режиме online.
ok	питание, отсутствие питания	Выход OK включен, когда на вход enable подано питание.
Q	I, Q, M, T, G, R, AI, AQ, Для данных бит/ слово: SA, SB, SC Для данных с плавающей точкой: R, AI, AQ	При выполнении пересылки значение из IN записывается в Q. Для MOVE_BOOL могут использоваться любые дискретные ячейки; выравнивание по байтам не требуется. Однако, 16 битов, начинающиеся с указанной ячейки памяти, отображаются в режиме online.

Пример 1

Когда вход разрешения %Q0014 включен, 48 бит пересылаются из области памяти, начинающейся с ячейки %M0001, в область памяти, начинающуюся с ячейки %M0033. (%M0001 и %M0003 определены как тип слово (WORD) при длине 3.)



Хотя происходит наложение 16 битов области назначения и исходной области, пересылка выполняется корректно.

Функции Пересылки Данных
Пересылка данных (Move Data)

Перед использованием функции пересылки (Move):

ВХОД (от %M0001 до %M0048)

%M0016	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0
%M0032	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
%M0048	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

После использования функции пересылки (Move):

ВХОД (от %M0033 до %M0080)

	1															33
%M0048	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0
%M0064	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
%M0080	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Функции Пересылки Данных
Пересылка блока (Block Move)

Функция пересылки блока (Block Move) копирует блок из семи констант в указанную область памяти. Когда на функцию пересылки блока (Block Move) подается питание, она копирует значения констант в последовательные ячейки памяти, начиная с указанной в выходе Q. Функция пропускает питание направо всякий раз, когда она его получает.

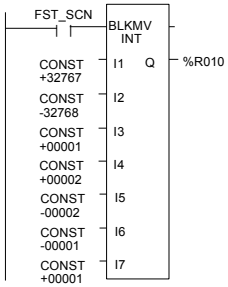
Параметры функции пересылки блока (Block Move)

Вход/ выход	Варианты	Описание
enable	питание	Пересылка блока выполняется, когда подано питание.
от I1 до I7	константа	Входы I1 - I7 содержат значения семи констант.
ok	питание, отсутствие питания	Выход OK включен, когда на вход enable подано питание.
Q	I, Q, M, T, G, R, AI, AQ, Для слов данных: SA, SB, SC Для данных с плавающей точкой: R, AI, AQ	Выход Q содержит первый элемент пересылаемого массива. Вход I1 пересылается в Q.

Функции Пересылки Данных
Пересылка блока (Block Move)

Пример

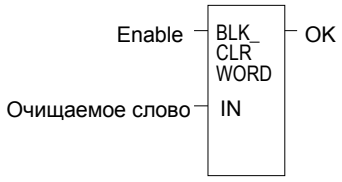
В примере всякий раз, когда вход разрешения FST_SCN устанавливается в 1, функция пересылки блока (Block Move) копирует входные константы в область памяти %R0010–16.



Функции Пересылки Данных
Очистка блока (Block Clear)

Функция очистки блока (Block Clear) заполняет указанный блок данных нулями. Когда на функцию подано питание, она записывает нули в область памяти, начинающуюся с ячейки, указанной в IN. При очистке данных дискретной памяти (%I, %Q, %M, %G или %T), информация о переходах, связанная с ячейками, также очищается.

Функция пропускает питание направо всякий раз, когда она его получает.

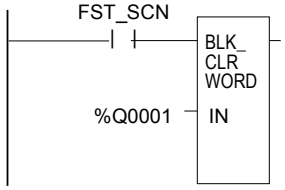


Параметры функции очистки блока (Block Clear)

Вход/ выход	Варианты	Описание
enable	питание	Массив очищается, когда подано питание.
IN	I, Q, M, T, SA, SB, SC, G, R, AI, AQ	Вход IN содержит первое слово очищаемого массива. Длина IN должна быть от 1 до 256 слов.
Длина		Количество очищаемых слов. Это длина IN.
ok	питание, отсутствие питания	Выход OK включен, когда на вход enable подано питание.

Пример

В этом примере при включении контроллера, 32 слова в памяти %Q (512 точек), начиная с %Q0001, заполняются нулями. %Q определяется как данные типа WORD при длине 32.



Функции Пересылки Данных
Регистр сдвига (Shift Register)

Функция регистра сдвига сдвигает одно или несколько слов данных или бит данных с исходного места в указанную область памяти. Например, одно слово может быть сдвинуто в область памяти с заданной длиной 5 слов. В результате этого сдвига, другое слова данных будет выдвинуто за пределы участка памяти.

Вход сброса (R) имеет приоритет перед входом разрешения (enable) функции. Когда вход сброса (R) включен, все ячейки регистра сдвига, начиная с ячейки, указанной входом ST, и по всей длине, указанной LEN, заполняются нулями.

Если на функцию подано питание и вход R выключен, каждый бит или слово регистра сдвига пересылаются в соседнюю старшую ячейку. Последний элемент регистра сдвига сдвигается в Q. Самая старшая ячейка IN сдвигается в освободившееся место, начинающееся с ST. Содержимое регистра сдвига доступно во время выполнения программы, т. к. оно находится в ячейках адресуемой памяти.

Функции Пересылки Данных
Регистр сдвига (Shift Register)

Параметры функции регистра сдвига

Вход/ выход	Варианты	Описание
enable	питание	Сдвиг выполняется, когда на вход разрешения (enable) питание подано, а на вход R нет.
Длина	от 1 до 256 битов или слов.	Длина регистра сдвига в битах или словах. Длина определяется как длина IN.
R	питание	Когда на вход R подано питание, регистр сдвига начинающийся с ST заполняется нулями.
IN	I, Q, M, T, S, G, R, AI, AQ, константы	IN содержит значение, сдвигаемое в первый бит или первое слово регистра сдвига. Для SHFR_BIT могут использоваться любые дискретные ячейки; их не требуется выравнивать по байтам.
ST	I, Q, M, T, SA, SB, SC, G, R, AI, AQ	ST содержит первый бит или первое слово регистра сдвига. Для SHFR_BIT могут использоваться любые дискретные ячейки; их не требуется выравнивать по байтам.
ok	питание, отсутствие питания	Выход OK включается, когда вход разрешения включен, а вход R выключен.
Q	I, Q, M, T, SA, SB, SC, G, R, AI, AQ	Выход Q содержит бит или слово, выдвигаемые из регистра сдвига. Для SHFR_BIT могут использоваться любые дискретные ячейки; их не требуется выравнивать по байтам.

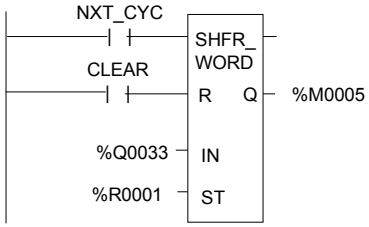
ПРЕДОСТЕРЕЖЕНИЕ: не рекомендуется допускать перекрытие диапазонов адресов входных и выходных ячеек в функциях, обрабатывающих несколько слов; это может привести к непредсказуемым последствиям.

Функции Пересылки Данных
Регистр сдвига (Shift Register)

Пример 1:

В примере регистр сдвига работает с памятью %R0001 - %R0100. (Регистр %R0001 определен как данные типа Word при длине 100). Когда ячейка сброса CLEAR включена, слова регистра сдвига заполняются нулями.

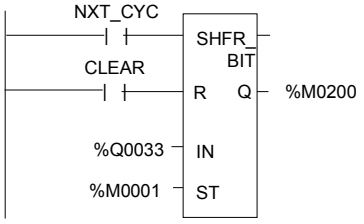
Когда включена ячейка NXT_CYC, а ячейка CLEAR выключена, слово %Q0033 из таблицы состояния выходов сдвигается в ячейку %R0001 регистра сдвига. Слово, сдвигаемое из ячейки %R0100 регистра сдвига, сохраняется в выходе %M0005.



Пример 2:

В этом примере регистр сдвига работает с памятью %M0001 - %M0100. (Ячейка %M0001 определена как данные типа Boolean при длине 100). Когда ячейка сброса CLEAR включена, функция регистра сдвига заполняет нулями ячейки с %M0001 по %M0100.

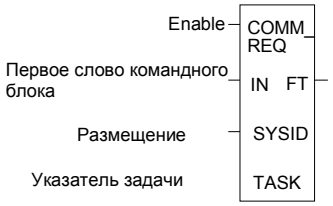
Когда включена ячейка NXT_CYC, а ячейка CLEAR выключена, функция регистра сдвига сдвигает данные ячеек %M0001 - %M0100 вниз на один бит. Бит %Q0033 сдвигается в ячейку %M0001, а бит, сдвигаемый из ячейки %M0100, записывается в ячейку %M0200.



Функции Пересылки Данных
Запрос связи (COMMREQ)

Функция запроса связи (COMMREQ) связывается с интеллектуальным модулем. Существует несколько типов запросов связи (COMMREQ). Информация, приведенная ниже, описывает только основной формат функции.

Когда функция получает питание, командный блок данных пересылается в указанный модуль. После посылки запроса связи (COMMREQ) программа может приостановить выполнение и ожидать ответа в течение максимального периода ожидания, указанного командой, или немедленно продолжить выполнение.



Параметры функции COMMREQ

Вход/ выход	Варианты	Описание
enable	питание	Запрос связи выполняется, когда функция включена.
IN	R, AI, AQ	IN содержит первое слово командного блока.
SYSID	I, Q, M, T, G, R, AI, AQ, константы	SYSID содержит номер крейта (старший значащий байт) и номер слота (младший значащий байт) устройства.
TASK	R AI, AQ, константа	TASK содержит идентификационный номер задачи процесса приемного устройства.
FT	питание, отсутствие питания	Выход FT включается, если при выполнении COMMREQ обнаружены ошибки: 1. Отсутствует указанный целевой адрес (SYSID). 2. Указанная задача не применима для устройства (TASK). 3. Длина данных равна 0. 4. Адрес указателя состояния устройства (в командном блоке) не существует.

Функции Пересылки Данных
Запрос связи (COMMREQ)

Командный блок для функции COMMREQ

Командный блок начинается с адреса, указанного в параметре IN функции COMMREQ. Размер командного блока зависит от количества данных, передаваемых устройству.

Командный блок содержит данные обмена с другим устройством и информацию, касающуюся выполнения функции запроса связи COMMREQ. Командный блок имеет следующую структуру:

адрес	Длина (в словах)
адрес + 1	Флаг Wait/No Wait (ждать/не ждать)
адрес + 2	Область памяти статуса
адрес + 3	Смещение области статуса
адрес + 4	Значение тайм-аута простоя
адрес + 5	Максимальное время связи
адрес + 6 - адрес + 133	Блок данных

Пример

В примере всякий раз, когда вход разрешения %M0020 включен, командный блок, начинающийся с адреса %R0016, выполняет задачу связи с устройством, расположенным в крейте 1, слоте 2 ПЛК. При возникновении ошибки в процессе выполнения запроса связи (COMMREQ) катушка %Q0100 устанавливается в 1.

Функции Преобразования Типов Данных

Функции преобразования типов данных используются для изменения типа элемента данных. Многие инструкции программы, такие как математические функции, должны использоваться с данными одного типа.

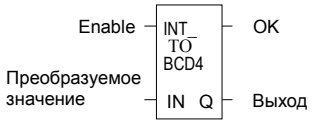
- Преобразовать данные в формат BCD-4
- Преобразовать данные в формат целого числа со знаком (INT)
- Преобразовать данные в формат целого числа двойной точности (DINT)
- Преобразовать данные в формат вещественного числа (REAL)
- Преобразовать данные в формат слова (WORD)
- Округление вещественного числа (TRUN)

Функции Преобразования Типов Данных
Преобразовать целые данные со знаком в формат BCD-4

Функция преобразования в формат BCD-4 выводит 4-значный двоично-десятичный (BCD) эквивалент целого числа со знаком. Исходные данные этой функцией не изменяются.

Данные могут преобразовываться в формат BCD для управления двоично-десятичными светодиодными дисплеями или для предустановки внешних устройств, таких как высокоскоростные счетчики.

Когда функция получает питание, она выполняет преобразование, выводя результат на выход Q. Функция пропускает питание всякий раз, когда она его получает, если значение результата указанного преобразования не выходит из диапазона 0 - 9999.



Параметры функции преобразования в формат BCD-4

Вход/ выход	Варианты	Описание
enable	питание	Преобразование выполняется, когда подано питание.
IN	I, Q, M, T, G, R, AI, AQ, константы	IN содержит ячейку с целым числом, преобразуемым в BCD-4.
OK	питание, отсутствие питания	Выход OK включается, когда функция выполняется без ошибок.
Q	I, Q, M, T, G, R, AI, AQ	Выход Q содержит BCD-4 форму исходного значения IN.

Функции Преобразования Типов Данных
Преобразовать целые данные со знаком в формат BCD-4

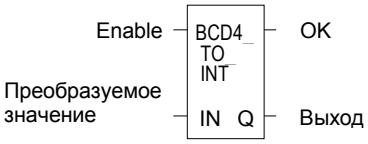
Пример

В примере всякий раз, когда вход %I0002 устанавливается в 1 и отсутствуют ошибки, целое число, находящееся в ячейках с %I0017 по %I0032, преобразуется в четыре двоично-десятичные (BCD) цифры, и результат сохраняется в ячейках с %Q0033 по %Q0048. Катушка %Q1432 используется для контроля успешности преобразования.

Функции Преобразования Типов Данных
Преобразовать в целое число со знаком (INT)

Функция преобразования в целое число со знаком выводит целочисленный эквивалент данных в формате BCD-4 или вещественных данных. Исходные данные этой функцией не меняются.

Когда функция получает питание, она выполняет преобразование, выводя результат на выход Q. Функция пропускает питание всякий раз, когда она его получает, если данные не выходят за пределы допустимого диапазона.

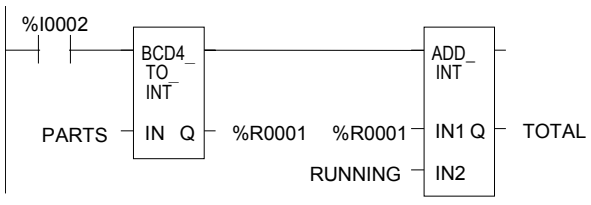


Параметры функции преобразования в целое число со знаком

Вход/ выход	Варианты	Описание
enable	питание	Преобразование выполняется, когда подано питание.
IN	Для BCD-4: I, Q, M, T, G, R, AI, AQ, константы Для вещественных чисел: R, AI, AQ	IN содержит значение в формате BCD-4, вещественном, или значение константы, преобразуемое в целое число со знаком.
ok	питание, отсутствие питания	Выход OK включается всякий раз, когда функция получает питание, если данные не выходят из диапазона и являются числом.
Q	I, Q, M, T, G, R, AI, AQ	Выход Q содержит целочисленный эквивалент исходного значения IN.

Пример

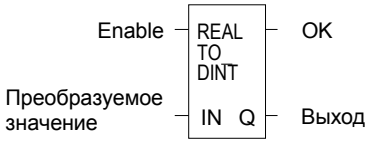
В примере всякий раз, когда вход %I0002 устанавливается в 1, число в формате BCD-4 из параметра PARTS преобразуется в целое число со знаком и передается в функцию сложения, где складывается с целым числом со знаком RUNNING. Сумма выводится функцией сложения в ячейку TOTAL.



Функции Преобразования Типов Данных
Преобразование в целое число двойной точности (DINT)

Функция преобразования в целое число двойной точности со знаком выводит целочисленный эквивалент вещественных данных двойной точности. Исходные данные этой функцией не изменяются.

Когда функция получает питание, она выполняет преобразование, выводя результат на выход Q. Функция пропускает питание всякий раз, когда она его получает, если вещественные данные не выходят за пределы допустимого диапазона.



Обратите внимание, что при преобразовании вещественных данных в целое число двойной точности может произойти потеря точности, т. к. вещественные данные имеют 24 значащих бита.

Параметры функции преобразования в целое число двойной точности со знаком

Вход/ выход	Варианты	Описание
enable	питание	Преобразование выполняется, когда подано питание.
IN	I, Q, M, T, G, R, AI, AQ, константы	Константа или ячейка с преобразуемым значением.
ok	питание, отсутствие питания	Выход OK включается всякий раз, когда функция получает питание, если данные не выходят из диапазона.
Q	R, AI, AQ	Ячейка, содержащая целочисленный эквивалент со знаком исходного значения двойной точности.

Функции Преобразования Типов Данных
Преобразование в целое число двойной точности (DINT)

Пример

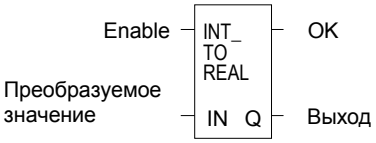
В примере всякий раз, когда вход %I0002 устанавливается в 1, целое значение из ячейки %I0017 преобразуется в целое значение двойной точности со знаком, и результат помещается в ячейку %R0001. Выход %Q1001 устанавливается в 1 всякий раз, когда функция выполняется успешно.

Функции Преобразования Типов Данных
Преобразование в вещественные данные (REAL)

Функция преобразования в вещественные данные выводит вещественный эквивалент входных данных. Исходные данные этой функцией не изменяются.

Когда функция получает питание, она выполняет преобразование, выводя результат на выход Q. Функция пропускает питание всякий раз, когда она его получает, если значение результата указанного преобразования не выходит за пределы допустимого диапазона.

Обратите внимание, что при преобразовании целого числа с двойной точностью в вещественные данные может произойти потеря точности, т. к. количество значащих бит уменьшается до 24.

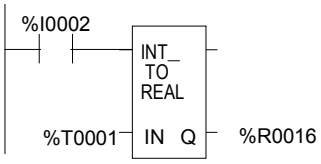


Параметры функции преобразования в формат вещественного числа

Вход/ выход	Варианты	Описание
enable	питание	Преобразование выполняется, когда подано питание.
IN	R AI, AQ, константа Только для INT: I, Q, M, T, G	IN содержит ячейку с целым числом, преобразуемым в вещественное.
ok	питание, отсутствие питания	OK включен, если функция выполнена без ошибок.
Q	R, AI, AQ	Вещественный эквивалент исходного значения в IN.

Пример

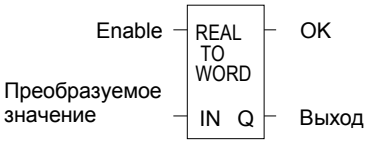
В примере целочисленное значение входа IN равно 678. Результат, помещенный в %T0016, равен 678.000.



Функции Преобразования Типов Данных
Преобразование вещественных данных в форму слов (WORD)

Функция преобразования в данные слова выводит эквивалент вещественных данных в формате слова. Исходные данные этой функцией не изменяются.

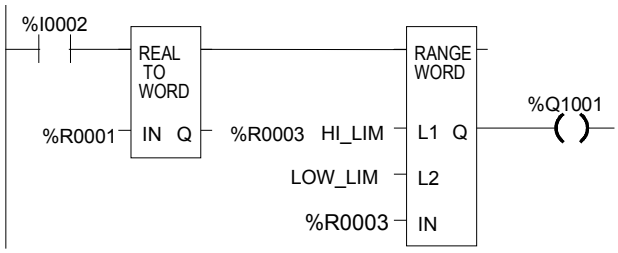
Когда функция получает питание, она выполняет преобразование, выводя результат на выход Q. Функция пропускает питание всякий раз, когда она его получает, если значение результата указанного преобразования не выходит из диапазона 0 - FFFFh.



Параметры функции преобразования в данные слова

Вход/ выход	Варианты	Описание
enable	питание	Преобразование выполняется, когда подано питание.
IN	R AI, AQ, константа	IN содержит ячейку со значением, преобразуемым в формат слова.
ok	питание, отсутствие питания	OK включен, если функция выполняется без ошибок.
Q	I, Q, M, T, G, R, AI, AQ	Содержит исходное значение IN в формате слова.

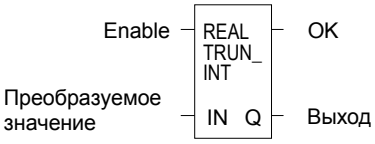
Пример



Функции Преобразования Типов Данных
Округление вещественного числа (TRUN)

Функция округления копирует вещественное число и округляет скопированное число в сторону уменьшения до целого или целого с двойной точностью. Исходные данные этой функцией не изменяются.

Когда функция получает питание, она выполняет преобразование, выводя результат на выход Q. Функция пропускает питание всякий раз, когда она его получает, если значение результата указанного преобразования не выходит за пределы допустимого диапазона и значение IN является числом.

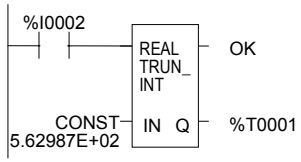


Параметры функции округления (TRUN)

Вход/ выход	Варианты	Описание
enable	питание	Преобразование выполняется, когда подано питание.
IN	R AI, AQ, константа	IN содержит ячейку с округляемым вещественным значением.
ok	питание, отсутствие питания	Выход OK включается всякий раз, когда функция получает питание, если данные не выходят за пределы допустимого диапазона и являются числом.
Q	R, AI, AQ Только для целых чисел: I, Q, M, T, G	Q содержит округленное целое или целое с двойной точностью значение исходного значения IN.

Пример

В примере показанная константа округляется и получившееся целое число 562 помещается в ячейку %T0001.



Математические и Арифметические Функции

Этот раздел описывает математические и арифметические функции системы команд:

- Стандартные математические функции: Сложение (ADD), Вычитание (SUB), Умножение (MUL), Деление (DIV)
- Деление по модулю
- Функция масштабирования
- Квадратный корень
- Тригонометрические функции
- Логарифмические/Экспоненциальные функции
- Преобразование в градусы
- Преобразование в радианы

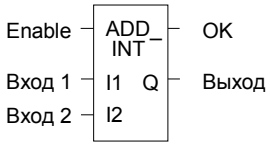
Преобразование данных для математических и числовых функций

Может потребоваться включить в программу логику для преобразования данных в другой тип перед использованием математической или числовой функции. Описание каждой функции включает информацию о подходящих типах данных. Как преобразовать данные в другой тип, объясняется в разделе Функции Преобразования типов Данных.

Математические и Арифметические Функции
Сложение (ADD), Вычитание (SUB), Умножение (MUL), Деление (DIV)

Стандартными математическими функциями являются сложение (ADD), вычитание (SUB), умножение (MUL), деление (DIV). Функция деления (DIV) округляет в меньшую сторону; она не округляет до ближайшего целого. (Например, 24 DIV 5 = 4.)

Когда математическая функция получает питание, выполняется действие с входными параметрами I1 и I2. Параметры I1, I2 и выход Q должны быть данными одного типа.



Математические функции пропускают питание, если нет математического переполнения. В случае переполнения результатом будет самое большое значение с соответствующим знаком и отсутствие питания на выходе.

Параметры стандартных математических функций

Вход/ выход	Варианты	Описание
enable	питание	Операция выполняется, когда подано питание.
I1	Все типы данных: R AI, AQ, константа Только данные типа INT: I, Q, M, T, G	I1 содержит константу или ячейку с первым значением, используемым в действии. (I1 находится на левой стороне математического выражения, как в I1 + I2). Диапазон констант при действиях с целыми числами двойной точности со знаком составляет минимум/максимум значения DINT.
I2	Все типы данных: R AI, AQ, константа Только данные типа INT: I, Q, M, T, G	I2 содержит константу или ячейку со вторым значением, используемым в действии. (I2 находится на правой стороне математического выражения, как в I1 + I2). Диапазон констант при действиях с целыми числами двойной точности со знаком составляет минимум/максимум значения DINT.
ok	питание, отсутствие питания	Выход OK включен, когда функция выполняется без переполнения, если не происходит недопустимой операции.
Q	Все типы данных: R, AI, AQ Только данные типа INT: I, Q, M, T, G	Выход Q содержит результат операции.

Математические и Арифметические Функции Сложение (ADD), Вычитание (SUB), Умножение (MUL), Деление (DIV)

Типы данных для стандартных математических функций

Стандартные математические функции работают со следующими типами данных:

INT	Целое число со знаком
DINT	Целое число со знаком двойной точности
REAL	Число с плавающей точкой

Типы данных входных и выходных параметров должны быть одинаковыми (16 бит или 32 бита).

Математические и Арифметические Функции
Сложение (ADD), Вычитание (SUB), Умножение (MUL), Деление (DIV)

Избежание переполнения

Будьте внимательны, чтобы избежать переполнения при использовании функций умножения (ADD) и деления (DIV).

Если вы должны преобразовать целое значение в целое с двойной точностью, помните, что ЦПУ использует стандартное двоичное представление со знаком, расположенным в старшем бите второго слова. Вы должны проверить знак младшего 16-битового слова и перенести его во второе 16-битовое слово. Если старший бит в 16-битовом слове типа INT - 0 (плюс), перенесите 0 во второе слово. Если старший бит в 16битовом слове – 1 (минус), перенесите 1 или шестнадцатиричное значение 0FFFFh во второе слово.

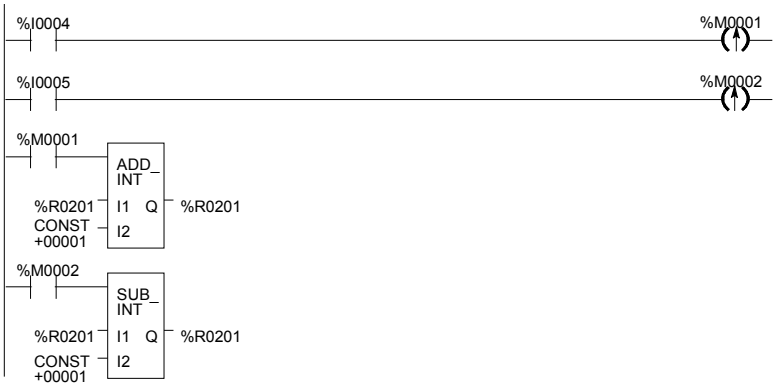
Преобразование из целого с двойной точностью в целое проще, т. к. младшее 16-битовое слово (первый регистр) является целой частью 32-битового слова в формате целого значения с двойной точностью (DINT). Старшие 16 битов должны быть 0 (плюс) или 1 (минус), в противном случае целое число с двойной точностью будет слишком большим для преобразования в 16 бит.

Математические и Арифметические Функции
Сложение (ADD), Вычитание (SUB), Умножение (MUL), Деление (DIV)

Пример

В этом примере используются функции сложения и вычитания для учета количества деталей в зоне временного хранения. Каждый раз, когда деталь поступает в зону хранения, питание через реле %I0004 поступает на катушку положительного перехода с адресом %M0001. Реле %M0001 подает питание на функцию сложения, прибавляя значение 1 (константу) к текущему общему значению, находящемуся в ячейке %R0201.

Каждый раз, когда деталь покидает в зону хранения, питание через реле %I0005 поступает на катушку положительного перехода с адресом %M0002. Реле %M0002 подает питание на функцию вычитания, вычитая значение 1 (константу) из текущего общего значения, находящегося в ячейке %R0201.



Математические и Арифметические Функции
Деление по модулю (MOD)

Функция деления по модулю делит одно значение на другое (имеющее тот же тип данных), для получения остатка. Знак остатка всегда совпадает со знаком входного параметра I1. Функция деления по модулю работает со следующими типами данных:

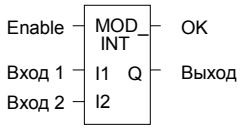
INT	Целое число со знаком
DINT	Целое число со знаком двойной точности

Когда функция получает питание, она делит вход I1 на вход I2. Эти параметры должны иметь одинаковый тип данных. Выход Q считается по формуле:

Q = I1-((I1 DIV I2) * I2)

Деление (DIV) дает целое число. Тип данных выхода Q совпадает с типом данных входов I1 и I2.

При получении функцией питания выход OK всегда включен, если не выполняется деление на 0. В этом случае он выключен.



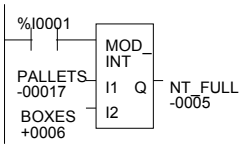
Параметры функции деления по модулю (MOD)

Вход/ выход	Варианты	Описание
enable	питание	Операция выполняется, когда подано питание.
I1	Все типы данных: R AI, AQ, константа Только данные типа INT: I, Q, M, T, G	I1 содержит константу или ячейку со значением, которое будет делиться на I2. Диапазон констант при действиях с целыми числами двойной точности со знаком составляет минимум/максимум значения DINT.
I2	Все типы данных: R AI, AQ, константа Только данные типа INT: I, Q, M, T, G	I2 содержит константу или ячейку со значением, на которое будет делиться I1. Диапазон констант при действиях с целыми числами двойной точности со знаком составляет минимум/максимум значения DINT.
ok	питание, отсутствие питания	Выход OK включается, когда функция выполняется без переполнения.
Q	Все типы данных: R, AI, AQ Только данные типа INT: I, Q, M, T, G	Выход Q содержит остаток от деления I1 на I2.

Математические и Арифметические Функции
Деление по модулю (MOD)

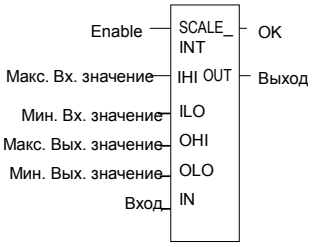
Пример

В примере остаток от деления значения PALLETS на значение BOXES помещается в NT_FULL, всякий раз, когда контакт %I0001 включен.



Математические и Арифметические Функции
Масштабирование (SCALE)

Функция масштабирования масштабирует входной параметр и помещает результат в выходную ячейку. Для данных целого типа (INT) все параметры должны быть целыми (со знаком). Для данных типа слово (WORD) все параметры должны быть типа слово (без знака).



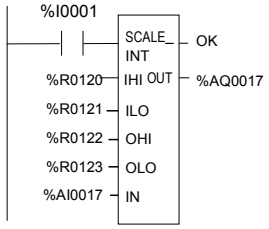
Параметры функции масштабирования (SCALE)

Вход/ выход	Варианты	Описание
enable	питание	Операция выполняется, когда подано питание.
IHI ILO	R AI, AQ, константа	Параметры IHI и ILO содержат константы или адреса ячеек для верхнего и нижнего пределов не масштабированных данных. Эти пределы, совместно со значениями параметров OHI и OLO, используются для вычисления коэффициента масштабирования, который будет применяться к входному значению IN.
OHI OLO	R AI, AQ, константа	Параметры OHI и OLO содержат константы или адреса ячеек для верхнего и нижнего пределов масштабированных данных.
IN	R AI, AQ, константа	IN содержит константу или адрес ячейки фактического масштабируемого значения.
ok	питание, отсутствие питания	Выход OK включается, когда функция выполняется без переполнения.
OUT	R, AI, AQ	Выход OUT содержит масштабированный эквивалент входного значения.

Математические и Арифметические Функции
Масштабирование (SCALE)

Пример

В примере регистры с %R0120 по %R0123 используются для хранения верхних и нижних значений масштабирования. Входным масштабируемым значением является аналоговый вход %AI0017. Масштабированные выходные данные используются для управления аналоговым выходом %AQ0017. Масштабирование выполняется всякий раз, когда контакт %I0001 включен.



Математические и Арифметические Функции
Квадратный корень (SQROOT)

Функция квадратного корня (SQROOT) находит квадратный корень значения. Когда функция получает питание, выход Q устанавливается в значение целой части квадратного корня значения входа IN. Тип данных выхода Q должен совпадать с типом данных IN.

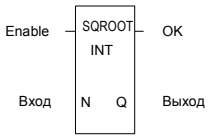
Функция квадратного корня (SQROOT) работает со следующими типами данных:

INT	Целое число со знаком
DINT	Целое число со знаком двойной точности
REAL	Число с плавающей точкой

Выход OK включен, если функция выполняется без переполнения, пока не произойдет одна из следующих недопустимых вещественных операций:

- IN < 0
- IN не является числом (NaN)

В этом случае выход OK выключен.



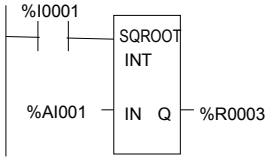
Параметры функции квадратного корня (SQROOT)

Вход/ выход	Варианты	Описание
enable	питание	Операция выполняется, когда подано питание.
IN	Все типы данных: R AI, AQ, константа Только данные типа INT: I, Q, M, T, G	Константа или адрес значения, квадратный корень которого будет вычисляться. Если IN меньше 0, функция не пропустит питания. Диапазон констант при действиях с целыми числами двойной точности со знаком составляет минимум/максимум значения DINT.
ok	питание, отсутствие питания	Выход OK включен, когда функция выполняется без переполнения, если не происходит недопустимой операции.
Q	Все типы данных: R, AI, AQ Только данные типа INT: I, Q, M, T, G	Выход Q содержит квадратный корень IN.

Математические и Арифметические Функции
Квадратный корень (SQROOT)

Пример

В примере, квадратный корень целого числа, находящегося в ячейке %AI001, помещается в ячейку %R0003 всякий раз, когда контакт %I0001 включен.

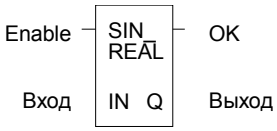


Математические и Арифметические Функции
Тригонометрические функции (Trigonometric Functions)

Существует шесть тригонометрических функций: синус (SIN), косинус (COS), тангенс (TAN), арксинус (ASIN), арккосинус (ACOS) и арктангенс (ATAN).

Синус (SIN), косинус (COS) и тангенс (TAN)

Когда функция синус (SIN), косинус (COS) или тангенс (TAN) получает питание, она выполняет действие над значением IN, выраженным в радианах, и сохраняет результат в выходе Q. И IN, и Q являются значениями с плавающей точкой.



Функции синус (SIN), косинус (COS) и тангенс (TAN) принимают широкий диапазон значений, а именно

$$-2^{63} < IN < +2^{63}, (2^{63} = 9.22 \times 10^{18})$$

Арксинус (ASIN), арккосинус (ACOS) и арктангенс (ATAN)

Когда функция арксинус (ASIN), арккосинус (ACOS) или арктангенс (ATAN) получает питание, она выполняет действие над значением IN, и сохраняет результат, выраженный в радианах, в выходе Q. И IN, и Q являются значениями с плавающей точкой.

Функции арксинус (ASIN) и арккосинус (ACOS) принимают узкий диапазон входных значений, а именно

$$-1 \leq IN \leq 1.$$

Получив допустимое значение параметра IN, арксинус выдает результат Q, как:

$$ASIN(IN) = \frac{\pi}{2} \leq Q \leq \frac{\pi}{2}$$

Арккосинус выдает результат Q, как:

$$ACOS(IN) = 0 \leq Q \leq \pi$$

Математические и Арифметические Функции
Тригонометрические функции (Trigonometric Functions)

Арктангенс принимает широкий диапазон значений, а именно

$$-\infty \leq IN \leq +\infty.$$

Получив допустимое значение параметра IN, арктангенс выдает результат Q, как:

$$ATAN(IN) = \frac{\pi}{2} \leq Q \leq \frac{\pi}{2}$$

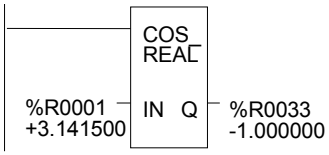
Математические и Арифметические Функции
Тригонометрические функции

Параметры тригонометрических функций

Вход/ выход	Варианты	Описание
enable	питание	Операция выполняется, когда подано питание.
IN	R AI, AQ, константа	IN содержит константу или адрес вещественного значения, над которым выполняется действие.
ok	питание, отсутствие питания	Выход OK включен, когда функция выполняется без переполнения, если не происходит недопустимая операция и/или параметр IN не окажется не числом (NaN).
Q	R, AI, AQ	Выход Q содержит тригонометрическое значение параметра IN.

Пример

В примере косинус (COS) значения ячейки %R0001 помещается в ячейку %R0033.



Математические и Арифметические Функции Логарифмические/Экспоненциальные функции (Logarithmic / Exponential Functions)

Когда логарифмическая или экспоненциальная функция получает питание, она выполняет соответствующее логарифмическое/экспоненциальное действие над вещественным значением входа IN и помещает результат в выход Q.

- Для функции десятичного логарифма (LOG), десятичный логарифм значения IN помещается в Q.
- Для функции натурального логарифма (LN), натуральный логарифм значения IN помещается в Q.
- Для функции возведения ***e*** в степень (EXP), ***e*** возводится в степень, указанную в IN, и результат помещается в Q.
- Для функции возведения X в степень (EXPT), значение входа I1 возводится в степень, указанную значением I2, и результат помещается в Q. (Функция EXPT имеет три входных параметра и два выходных параметра.)

Выход OK будет включен, пока вход не будет отрицательным или не окажется не числом.

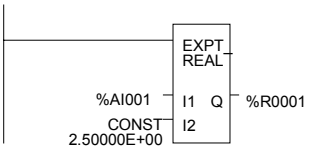
Параметры логарифмических/экспоненциальных функций

Вход/ выход	Варианты	Описание
enable	питание	Операция выполняется, когда подано питание.
IN или I1, I2	R AI, AQ, константа	Для EXP, LOG и LN, IN содержит вещественное значение, над которым выполняется действие. Функция EXPT имеет два входа, I1 и I2. Для EXPT, I1 - значение основания, а I2 - экспонента.
ok	питание, отсутствие питания	Выход OK включен, когда функция выполняется без переполнения, если не происходит недопустимая операция и/или параметр IN не окажется не числом (NaN) или отрицательным.
Q	R, AI, AQ	Выход Q содержит значение логарифмирования/возведения в степень IN.

Математические и Арифметические Функции Логарифмические/Экспоненциальные функции (Logarithmic / Exponential Functions)

Пример функции возведения в степень (EXPT)

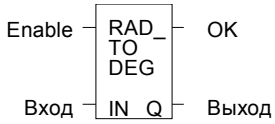
В примере значение ячейки %AI001 возводится в степень 2.5, и результат помещается в ячейку %R0001.



Математические и Арифметические Функции
Функции преобразования радиан

Когда функция преобразования градусов/радиан получает питание, выполняется соответствующее преобразование (радианы в градусы (RAD_TO DEG) или градусы в радианы (DEG_TO RAD)) вещественного значения входа IN, и результат помещается в выход Q.

Выход OK будет получать питание, если параметр IN не окажется не числом (NaN).

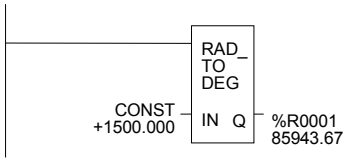


Параметры функции преобразования радиан

Вход/ выход	Варианты	Описание
enable	питание	Операция выполняется, когда подано питание.
IN	R AI, AQ, константа	IN содержит вещественное значение, над которым выполняется действие.
ok	питание, отсутствие питания	Выход OK включен, когда функция выполняется без переполнения, пока IN не окажется не числом (NaN).
Q	R, AI, AQ	Выход Q содержит преобразованное значение параметра IN.

Пример

В примере +1500 преобразуется в градусы (DEG) и помещается в %R0001.

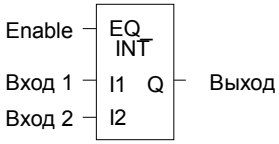


Функции сравнения

Функции сравнения используются для сравнения двух чисел и определения, находится ли число в указанном диапазоне.

- Равно (EQ) — Проверяет два числа на равенство
- Не равно (NE) — Проверяет два числа на неравенство
- Больше (GT) — Проверяет, больше одно число другого, или нет
- Больше или равно (GE) — Проверяет, больше или равно одно число другому, или нет
- Меньше (LT) — Проверяет, меньше одно число другого, или нет
- Меньше или равно (LE) — Проверяет, меньше или равно одно число другому, или нет
- Диапазон (RANGE) — Проверяет, находится ли одно число между двумя другими числами

Когда функция получает питание, она сравнивает вход I1 и вход I2. Эти параметры должны иметь одинаковый тип данных.



Если входы IN1 и IN2 удовлетворяют указанным условиям сравнения, выход Q получает питание и устанавливается в 1; в противном случае он устанавливается в 0.

Типы данных для функций сравнения

Функции сравнения работают со следующими типами данных:

INT	Целое число со знаком
DINT	Целое число со знаком двойной точности
REAL	Число с плавающей точкой

Бит %S0020 устанавливается в 1, когда функция сравнения, использующая вещественные данные, выполняется успешно. Он устанавливается в 0, когда хотя бы один вход не является числом (NaN).

Функции сравнения
Равно (EQ), Не равно (NE), Меньше (LT), Меньше или равно (LE), Больше (GT), Больше или равно (GE)

Параметры функций сравнения

Вход/ выход	Варианты	Описание
enable	питание	Операция выполняется, когда подано питание.
IN1	R AI, AQ, константа Только для данных типа INT: I, Q, M, T, G	IN1 содержит константу или адрес первого сравниваемого значения. Значение IN1 должно быть допустимым. Константы должны быть целыми для действий с целыми числами двойной точности со знаком. IN1 находится в выражении сравнения слева, как в IN1 < IN2.
IN2	R AI, AQ, константа Только для данных типа INT: I, Q, M, T, G	IN2 содержит константу или адрес второго сравниваемого значения. Значение IN2 должно быть допустимым. Константы должны быть целыми для действий с целыми числами двойной точности со знаком. IN2 находится в выражении сравнения справа, как в IN1 < IN2.
Q	питание, отсутствие питания	Выход Q включается, когда IN1 и IN2 удовлетворяют указанному сравнению.

Пример

В примере два целых числа двойной точности со знаком проверяются на равенство. Когда контакт %I0001 пропускает питание на функцию Меньше или равно (LE), значение с именем PWR_MDE сравнивается со значением с именем BIN_FUL. Если значение PWR_MDE меньше или равно значению BIN_FUL, включается катушка %Q0002.

Функции сравнения
Диапазон (RANGE)

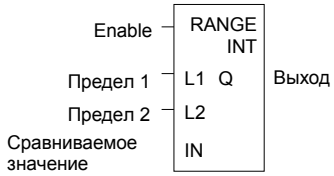
Функция Диапазон (RANGE) определяет, находится ли значение в диапазоне между двумя числами.

Типы данных для функции Диапазон (RANGE)

Функция Диапазон (RANGE) работает со следующими типами данных:

INT	Целое со знаком (по умолчанию).
DINT	Целое число со знаком двойной точности
WORD	Данные типа слово.

Когда на функцию Диапазон (RANGE) подается питание, она сравнивает значение входа IN с диапазоном, указанным пределами L1 и L2. И L1, и L2 могут быть как верхним, так и нижним пределом. Когда значение находится внутри диапазона, указанного пределами L1 и L2, включительно, выходной параметр Q устанавливается в 1, в противном случае Q устанавливается в 0.



Функции сравнения
Диапазон (RANGE)

Параметры функции Диапазон (RANGE)

Вход/ выход	Варианты	Описание
enable	питание	Операция выполняется, когда подано питание.
L1	R AI, AQ, константа Данные только типа INT и WORD: I, Q, M, T, G	L1 содержит начальную точку диапазона. Константы должны быть целыми для действий с целыми числами двойной точности со знаком.
L2	R AI, AQ, константа Данные только типа INT и WORD: I, Q, M, T, G	L2 содержит конечную точку диапазона. Константы должны быть целыми для действий с целыми числами двойной точности со знаком.
IN	R, AI, AQ Данные только типа INT и WORD: I, Q, M, T, G	IN содержит значение, сравниваемое с диапазоном, указанным L1 и L2.
Q	питание, отсутствие питания	Выход Q включается, когда значение IN находится внутри диапазона, указанного пределами L1 и L2, включительно.

Функции сравнения
Диапазон (RANGE)

Пример

В этом примере когда функция Диапазон (RANGE) получает питание через контакт %I0001, функция определяет, находится ли значение %AI001 внутри диапазона от 0 до 100.

%R0001 содержит значение 100. %R2 содержит значение 0.

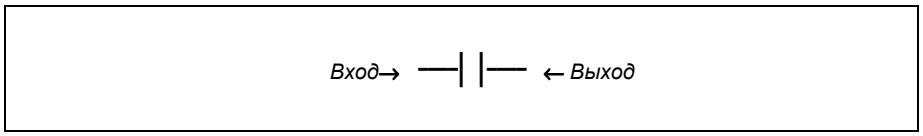
Выходная катушка %Q0001 включена, только если значение %AI0001 находится внутри диапазона от 0 до 100.

Значение IN %AI001	Состояние Q %Q0001
< 0	Выключен
0 - 100	Включен
> 100	Выключен

Функции реле

- Нормально разомкнутый контакт –| |–
- Нормально замкнутый контакт –|/|–
- Обычная катушка –( )–
- Записывающая катушка SET –(SM)–
- Записывающая катушка RESET –(RM)–
- Записывающая катушка с инверсией –(/M)–
- Катушка с инверсией –(/)–
- Записывающая катушка –(M)–
- Катушка фиксации (SET) –(S)–
- Катушка сброса (RESET) –(R)–
- Катушка положительного перехода –(↑)–
- Катушка отрицательного перехода –(↓)–
- Вертикальная связь vert |
- Горизонтальная связь horz –
- Катушка продолжения ——<+>
- Контакт продолжения <+>——

Каждый контакт и катушка реле имеют один вход и один выход. Вместе они обеспечивают прохождение логики через контакт или катушку.



Функции реле
Нормально разомкнутый, нормально замкнутый контакты и контакт продолжения

Контакт используется для контроля состояния ячейки. Будет ли контакт пропускать питание, зависит от состояния или статуса контролируемой ячейки и типа контакта. Ячейка включена, если она находится в состоянии 1; она выключена, если она находится в состоянии 0.

Тип контакта	Изображение	Контакт пропускает питание направо:
Нормально разомкнутый	-| |-	Когда ячейка включена.
Нормально замкнутый	-|/|-	Когда ячейка выключена.
Контакт продолжения	<+>---	Если предшествующая катушка продолжения включена.

Нормально разомкнутый контакт -| |-

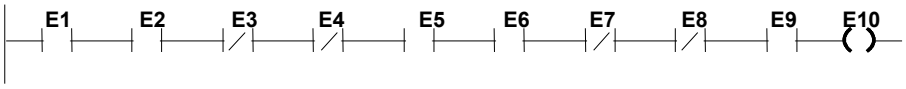
Нормально разомкнутый контакт работает как выключатель, пропускающий питание, если включена связанная с ним ячейка.

Нормально замкнутый контакт -|/|-

Нормально замкнутый контакт работает как выключатель, пропускающий питание, если выключена связанная с ним ячейка.

Пример

В примере показано звено из 10 элементов, имеющих имена от E1 до E10. Катушка E10 включена, если ячейки E1, E2, E5, E6 и E9 включены, а ячейки E3, E4, E7 и E8 выключены.



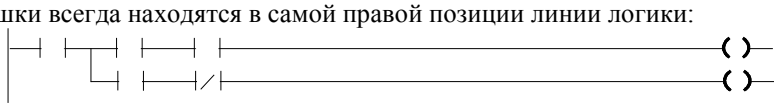
Функции реле
Нормально разомкнутый, нормально замкнутый контакты и контакт продолжения

Катушки и контакты продолжения

Катушки продолжения и контакты продолжения используются для продолжения звена релейно-контактной логики после последнего столбца. Состояние последней выполненной катушки является состоянием питания, используемого контактом продолжения, выполняемым следующим. Если логическое питание не подается на катушку продолжения, находящуюся перед выполняемым контактом продолжения, то на контакт питание не подается. В звене может быть только одна катушка продолжения и контакт продолжения; контакт продолжения должен находиться в столбце 1, а катушка продолжения должна быть в последнем столбце.

Функции реле
Катушки

Катушки используются для управления дискретными ячейками. Для управления питанием катушки должна использоваться условная логика. Катушки вызывают непосредственное действие; они не передают питание направо. Если дополнительная логика в программе должна выполняться в зависимости от состояния катушки, можно использовать внутренние ячейки для катушки или комбинацию катушка продолжения/контакт продолжения. Катушки всегда находятся в самой правой позиции линии логики:



Ячейки и проверка катушек

Когда уровень проверки катушек установлен на “однократное использование” (single), вы можете использовать указанную ячейку %M или %Q только с одной катушкой, однако, вы можете одновременно использовать ее с одной катушкой фиксации (Set) и одной катушкой сброса (Reset). Когда уровень проверки катушек установлен на “предупреждать о многократном использовании” (warn multiple) или “многократное использование” (multiple), каждая катушка может использоваться с несколькими обычными катушками, катушками фиксации (Set) и катушками сброса (Reset). При многократном использовании ячейки могут быть включены катушкой фиксации (Set) или обычной катушкой, и могут быть выключены катушкой сброса (Reset) или обычной катушкой.

Питание и записываемость

В следующей таблице показано, как питание различных типов катушек влияет на их ячейки. Состояние записывающих катушек сохраняется при пропадании питания ПЛК и при переходе ПЛК из режима Stop в режим Run. Не записывающие катушки устанавливаются в 0 при пропадании питания ПЛК и при переходе ПЛК из режима Stop в режим Run.

Тип катушки	Символ	Питание катушки	Результат
Обычная	-( )-	ON OFF	Устанавливает ячейку в 1, не записываемая. Устанавливает ячейку в 0, не записываемая.
Инвертирующая	-(/)-	ON OFF	Устанавливает ячейку в 0, не записываемая. Устанавливает ячейку в 1, не записываемая.
Записывающая	-(M)-	ON OFF	Устанавливает ячейку в 1, записываемая. Устанавливает ячейку в 0, записываемая.
Записывающая инвертирующая	-(/M)-	ON OFF	Устанавливает ячейку в 0, записываемая. Устанавливает ячейку в 1, записываемая.
Положительный переход	-(P)-	OFF→ON	Если питание на катушку в предыдущем цикле не подавалось, а в текущем цикле подается, устанавливает катушку в 1.

Функции реле
Катушки

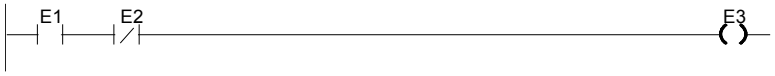
Отрицательный переход	-(N)-	ON→OFF	Если питание на катушку в предыдущем цикле подавалось, а в текущем цикле не подается, устанавливает катушку в 1.
SET (фиксация)	-(S)-	ON OFF	Устанавливает ячейку в 1, пока не будет установлена в 0 катушкой (R), не записываемая. Не изменяет состояние катушки, не записываемая.
RESET (сброс)	-(R)-	ON OFF	Устанавливает ячейку в 0, пока не будет установлена в 1 катушкой (S), не записываемая. Не изменяет состояние катушки, не записываемая.
Фиксация (SET) с записью	-(SM)-	ON OFF	Устанавливает ячейку в 1, пока не будет установлена в 0 катушкой (RM), записываемая. Не изменяет состояние катушки.
Сброс (RESET) с записью	-(RM)-	ON OFF	Устанавливает ячейку в 0, пока не будет установлена в 1 катушкой (SM), записываемая. Не изменяет состояние катушки.
Катушка продолжения	----<+>	ON OFF	Устанавливает следующий контакт продолжения в 1. Устанавливает следующий контакт продолжения в 0.

Функции реле
Катушки

При получении питания обычная катушка устанавливает дискретную ячейку в 1. Она не записываемая, следовательно, она не может использоваться с ячейками состояния системы (%SA, %SB, %SC, или %G).

Пример

В примере, катушка E3 включена, когда ячейка E1 включена, а ячейка E2 выключена.

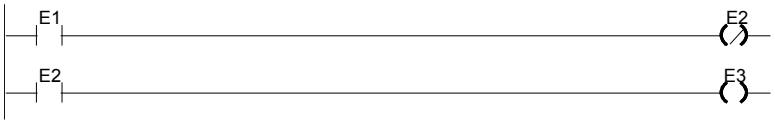


Инвертирующая катушка

Инвертирующая катушка устанавливает дискретную ячейку в 1, когда она не получает питание. Она не записываемая, следовательно, она не может использоваться с ячейками состояния системы (%SA, %SB, %SC, или %G).

Пример

В примере, катушка E2 включена, когда ячейка E1 выключена.



Записывающая катушка

Как и нормально разомкнутая катушка, записывающая катушка устанавливает дискретную ячейку в 1, когда она получает питание. Состояние записывающей катушки сохраняется при пропадании питания. Следовательно, она не может использоваться с ячейками строго не записываемой памяти (%T).

Записывающая инвертирующая катушка

Записывающая инвертирующая катушка устанавливает дискретную ячейку в 1, когда она не получает питание. Состояние записывающей катушки сохраняется при пропадании питания. Следовательно, она не может использоваться с ячейками строго не записываемой памяти (%T).

Функции реле
Катушки

Катушка положительного перехода

Если ячейка, связанная с катушкой положительного перехода, была выключена, то при получении катушкой питания она устанавливается в 1 до следующего выполнения катушки. (Если звено, содержащее катушку, пропускается в последующих циклах, она остается включенной.) Эта катушка может использоваться как импульсная.

Не записывайте данные из внешних устройств (таких, как модуль PCM, программатор, устройство ADS, и т. д.) в ячейки, используемые для катушек положительного перехода, т. к. это разрушит импульсный характер этих катушек.

Катушки перехода могут использоваться с ячейками записываемой и не записываемой памяти (%Q, %M, %T, %G, %SA, %SB или %SC).

Катушка отрицательного перехода

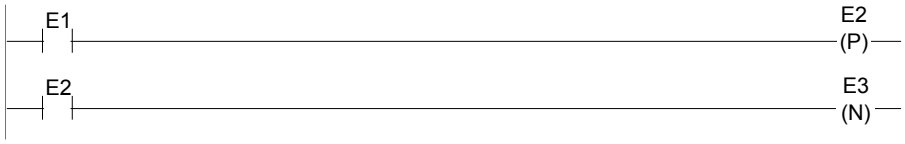
Если ячейка, связанная с катушкой, выключена, когда катушка перестает получать питание, ячейка устанавливается в 1 до следующего выполнения катушки.

Не записывайте данные из внешних устройств в ячейки, используемые для катушек положительного перехода, т. к. это разрушит импульсный характер этих катушек.

Катушки перехода могут использоваться с ячейками записываемой и не записываемой памяти (%Q, %M, %T, %G, %SA, %SB или %SC).

Пример

В примере, когда ячейка E1 переходит из 0 в 1, катушки E2 и E3 получают питание, включая E2 на один цикл. Когда E2 переходит из 1 в 0, с E2 и E3 снимается питание, включая катушку E3 на один цикл.



Функции реле
Катушки

Катушка фиксации (SET)

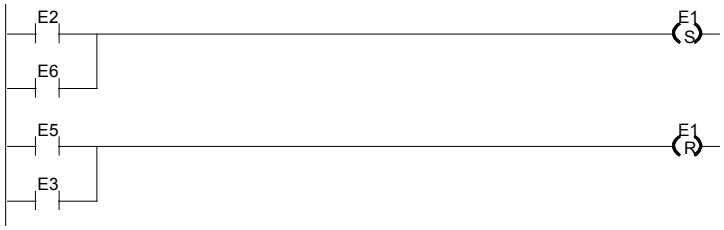
Катушки фиксации (SET) и сброса (RESET) являются незаписывающими катушками, которые используются для фиксации состояния ячейки в 1 или в 0. Когда катушка фиксации (SET) получает питание, ее ячейка остается включенной (независимо от того, получает сама катушка питание или нет) пока ячейка не будет сброшена другой катушкой.

Катушка сброса (RESET)

Катушка сброса (RESET) устанавливает дискретную ячейку в 0, если катушка получает питание. Ячейка остается выключенной, пока она не будет включена другой катушкой. Приоритет в паре имеет катушка, установленная последней (фиксации (SET) или сброса (RESET)).

Пример

В примере, катушка E1 включается при каждом включении ячеек E2 или E6. Катушка E1 выключается при каждом включении ячеек E5 или E3.



Записывающая катушка фиксации (SET)

Записывающие катушки фиксации (SET) и сброса (RESET) подобны катушкам фиксации (SET) и сброса (RESET), но они сохраняют свое значение при пропадании питания и при переходе ПЛК из режима Stop в режим Run. Записывающая катушка фиксации включает дискретную ячейку, если катушка получает питание. Ячейка остается включенной, пока не будет сброшена записывающей катушкой сброса (RESET).

Записывающая катушка сброса (RESET)

При получении питания эта катушка выключает дискретную ячейку. Ячейка остается выключенной, пока не будет установлена записывающей катушкой фиксации (SET). Состояние этой катушки сохраняется при пропадании питания и при переходе ПЛК из режима Stop в режим Run.

Табличные функции

Табличные функции используются для:

- Копирования массива данных: ARRAY MOVE
- Поиска значений в массиве

Максимальная длина, допустимая для этих функций, составляет 32767 для любого типа.

Типы данных для табличных функций

Табличные функции работают со следующими типами данных:

INT	Целое число со знаком
DINT	Целое число со знаком двойной точности
BOOL *	Данные типа бит
BYTE	Данные типа байт
WORD	Данные типа слово.

* Используется только с функцией пересылки массива (Array Move).

Табличные функции
Пересылка массива (Array Move)

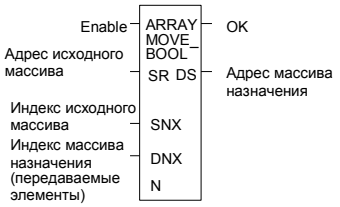
Функция пересылки массива (Array Move) копирует указанное количество элементов исходного массива в массив назначения. Когда функция получает питание, она копирует указанное количество элементов исходного массива, начинающегося с ячейки, указанной индексом. Затем функция записывает скопированные элементы в выходной массив, начинающийся с ячейки, указанной индексом.

Для битовых данных, когда для параметров начальных адресов исходного массива и/или массива назначения выбрана память слов, младший бит указанного слова является первым битом массива.

Отсчет индексов инструкции пересылки массива (Array Move) начинается с 1. При использовании функции пересылки массива (Array Move) не могут использоваться ячейки, находящиеся вне исходного массива или массива назначения (указанных их начальными адресами и длиной).

Выход OK получает питание, если не имеет место следующее:

- Вход Enable выключен.
- (N + SNX – 1) больше, чем длина.
- (N + DNX – 1) больше, чем длина.



Табличные функции
Пересылка массива (Array Move)

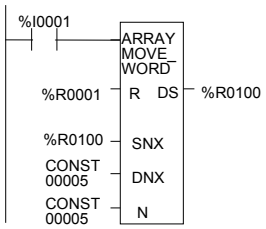
Параметры функции пересылки массива (Array Move)

Вход/ выход	Варианты	Описание
enable	питание	Операция выполняется, когда подано питание.
SR	Для всех: R, AI, AQ Для INT, BIT, BYTE, WORD: I, Q, M, T, G, Для BIT, BYTE, WORD: SA, SB, SC	SR содержит начальный адрес исходного массива. Для ARRAY_ MOVE_BOOL могут использоваться любые ячейки; выравнивание по байтам не требуется.
SNX	I, Q, M, T, G, R, AI, AQ, константы	SNX содержит индекс исходного массива.
DNX	I, Q, M, T, G, R, AI, AQ, константы	DNX содержит индекс массива назначения.
N	I, Q, M, T, G, R, AI, AQ, константы	N обеспечивает индикатор количества.
ok	питание, отсутствие питания	Выход OK включается, когда на вход enable подано питание.
DS	Для всех: SA, SB, SC, R, AI, AQ Для INT, BIT, BYTE, WORD: I, Q, M, T, G	Начальный адрес массива назначения. Для ARRAY_ MOVE_BOOL могут использоваться любые ячейки; выравнивание по байтам не требуется.
Длина		Количество элементов, начинающихся с SR и DS, которые формируют каждый массив. Он определяется длиной SR+DS.

Табличные функции
Пересылка массива (Array Move)

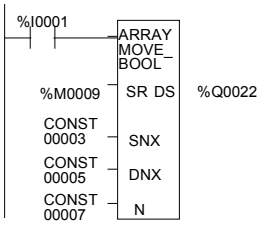
Пример 1:

В этом примере если %R100=3, тогда читаются ячейки %R0003 %R0007 массива %R0001 - %R0016 и записываются в ячейки %R0104 - %R0108 массива %R0100 - %R0115. (%R001 и %R0100 объявлены типом WORD при длине 16.)



Пример 2:

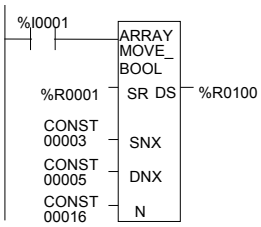
Для SR и DS используется память битов, читаются ячейки %M0011 - %M0017 массива %M0009 - %M0024 и записываются в ячейки %Q0026 - %Q0032 массива %Q0022 - %Q0037. (%M009 и %Q0022 объявлены типом BOOL при длине 16).



Табличные функции
Пересылка массива (Array Move)

Пример 3:

Для SR и DS используется память слов, читаются биты, начиная с третьего бита регистра %R0001 по второй бит регистра %R0002 массива, содержащего все 16 битов регистра %R0001 и четыре бита регистра %R0002 и затем записываются в биты, начиная с пятого бита регистра %R0100 по четвертый бит регистра %R0101 массива, содержащего все 16 битов регистра %R0100 и четыре бита регистра %R0101. %R0001 и %R0100 объявлены типом BOOL при длине 20.

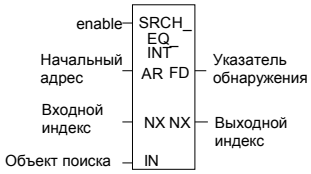


Табличные функции
Поиск значений в массиве

Используйте функции поиска, перечисленные ниже, для поиска значений в массиве.

▪ Поиск равных (Search Equal)	▪ Равенство указанному значению.
▪ Поиск не равных (Search Not Equal)	▪ Неравенство указанному значению.
▪ Поиск больших (Search Greater Than)	▪ Большее, чем указанное значение.
▪ Поиск больших или равных (Search Greater Than or Equal)	▪ Большее или равное указанному значению.
▪ Поиск меньших (Search Less Than)	▪ Меньшее, чем указанное значение.
▪ Поиск меньших или равных (Search Less Than or Equal)	▪ Меньшее или равное указанному значению.

Когда функция поиска (Search) получает питание, она просматривает указанный массив. Поиск начинается с начального адреса (AR) плюс значение индекса (NX).



Поиск продолжается, пока не будет найден элемент массива просматриваемого объекта (IN) или пока не будет достигнут конец массива. Если элемент массива найден, указатель обнаружения (FD) устанавливается в 1, а индекс выхода (выход NX) устанавливается в относительную позицию этого элемента в массиве. Если элемент массива не найден до достижения конца массива, указатель обнаружения (FD) и индекс выхода (выход NX) устанавливаются в 0.

Допустимые значения входа NX от 0 до (длина)- 1. NX следует установить в 0 для начала поиска с первого элемента. Это значение увеличивается на 1 во время выполнения. Следовательно, выход NX может принимать значения от 1 до (длина). Если значение входа NX выходит за диапазон, (< 0 или $\geq$ длина), его значение устанавливается в 0 по умолчанию.

Табличные функции
Поиск значений в массиве

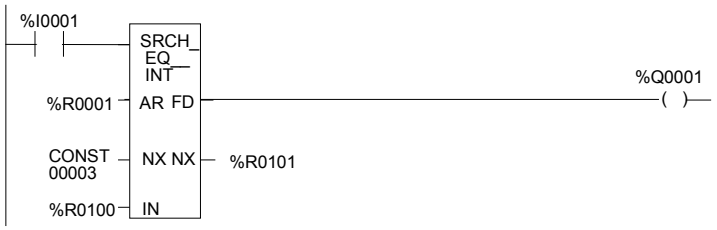
Параметры функций поиска

Вход/ выход	Варианты	Описание
enable	питание	Поиск выполняется, когда подано питание.
AR	Для всех: R, AI, AQ Для INT, BYTE, WORD: I, Q, M, T, G, Для BYTE, WORD: S	Содержит начальный адрес массива.
Вход NX	I, Q, M, T, G, R, AI, AQ, константы	Содержит индекс массива, отсчитываемый от 0, с которого начнется поиск.
IN	Для всех: R, AI, AQ Для INT, BYTE, WORD: I, Q, M, T, G, Для BYTE, WORD: S	IN содержит объект поиска.
Выход NX	I, Q, M, T, G, R, AI, AQ	Содержит позицию найденного элемента в массиве, отсчитываемую от 1.
FD	питание, отсутствие питания	FD указывает, что элемент массива найден, и функция успешно выполнена.
Длина	от 1 до 32,767 байтов или слов.	Количество просматриваемых элементов, начинающихся с AR, которые формируют массив.

Табличные функции
Поиск значений в массиве

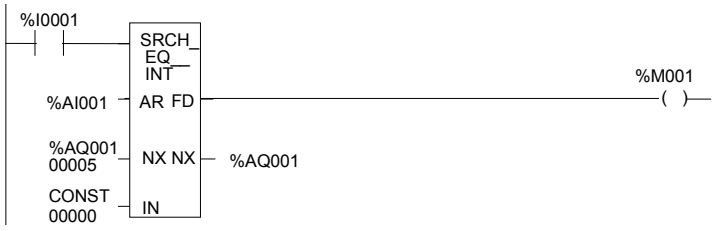
Пример 1:

Массив AR определен как адреса памяти %R0001 - %R0005. Когда EN включен, просматривается часть массива между %R0004 и %R0005 в поисках элемента, значение которого равно значению IN. Если %R0001 = 7, %R0002 = 9, %R0003 = 6, %R0004 = 7, %R0005 = 7 и %R0100 = 7, то поиск начнется с %R0004 и закончится в %R0004, при этом FD устанавливается в 1, и 4 записывается в %R0101.



Пример 2:

Массив AR определен как адреса памяти %AI001 - %AI016. Значения элементов массива - 100, 20, 0, 5, 90, 200, 0, 79, 102, 80, 24, 34, 987, 8, 0 и 500. В исходном состоянии, %AQ001 равен 5. Когда EN включен, каждый цикл будет просматриваться массив в поисках значения, совпадающего со значением IN - 0. В первом цикле поиск начнется с %AI006, и совпадение будет найдено в %AI007, FD включится, а %AQ001 станет равным 7. Во втором цикле поиск начнется с %AI008, и совпадение будет найдено в %AI015, FD останется включенным, а %AQ001 станет равным 15. В следующем цикле поиск начнется с %AI016. Т. к. совпадения не найдены до конца массива, FD выключится, а %AQ001 станет равным 0. В следующем цикле поиск начнется с начала массива.



Функции таймера и счетчика

Этот раздел описывает функции времени и счета системы команд. Данные, связанные с этими функциями, сохраняются при пропадании питания.

- Таймер задержки по включению со сбросом
- Таймер задержки по выключению
- Таймер задержки по включению
- Инкрементный счетчик
- Декрементный счетчик

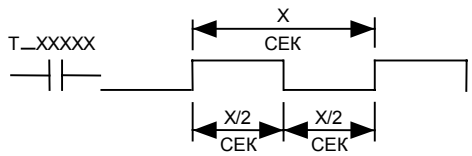
Периодические контакты времени

В дополнение к функциям таймера системы команд, ПЛК VersaMax имеет четыре периодических контакта времени. Их можно использовать для передачи другим функциям программы периодических импульсов. Четыре периодических контакта времени имеют периодичность: 0.01 секунды, 0.1 секунды, 1.0 секунда, и 1 минута.

Состояние этих контактов не изменяется во время выполнения цикла. Длительность импульса для этих контактов равна длительности паузы.

Ячейки контактов: T_10MS (0.01 секунды), T_100MS (0.1 секунды), T_SEC (1.0 секунда), и T_MIN (1 минута).

Приведенная ниже диаграмма показывает длительность импульсов и пауз для этих контактов.



Периодические контакты времени отображают определенные ячейки в памяти %S.

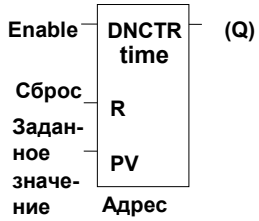
Функции таймера и счетчика

Функциональный блок данных, требуемый для таймеров и счетчиков

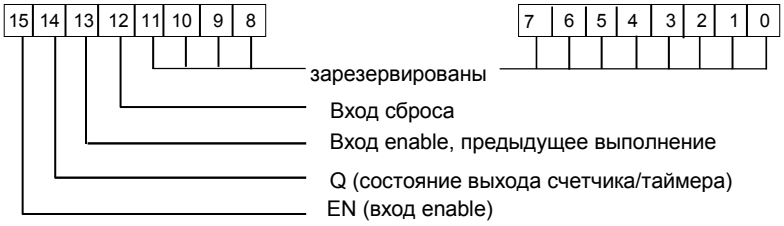
Каждый блок данных использует три слова (регистра) памяти %R для хранения следующей информации:

текущее значение (CV)	слово 1
заданное значение (PV)	слово 2
управляющее слово	слово 3

Когда вы вводите таймер или счетчик, вы должны ввести начальный адрес для этих трех слов (регистров). Не используйте смежные регистры для блоков таймера/счетчика из 3 слов. Таймеры и счетчики не будут работать, если вы поместите текущее значение блока поверх заданного значения предыдущего блока.



Управляющее слово сохраняет состояние логических входов и выходов связанного с ним функционального блока, в следующем формате:



Биты с 0 по 11 используются для обеспечения точности таймера; для счетчиков они не используются.

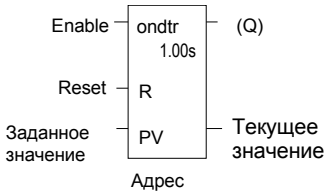
Если заданное значение (PV) не является константой, PV обычно размещается в ячейке, отличной от второго слова. Некоторые приложения используют адрес второго слова для PV, например %R0102, если блок данных начинается с %R0101. Это позволяет изменять заданное значение во время работы

таймера или счетчика. Первое (CV) и третье (Control) слова могут быть прочитаны, но не могут быть записаны, в противном случае функция не будет работать.

Функции таймера и счетчика
Таймер задержки по включению со сбросом

Записывающий таймер задержки по включению со сбросом (ONDTR) накапливает время, когда он получает питание, и сохраняет свое значение, когда питание прерывается. Время может отсчитываться в десятых (0.1), сотых (0.01), или тысячных (0.001) секунды. Диапазон единиц времени составляет от 0 до +32,767. Состояние таймера сохраняется при прерывании питания; при подаче питания автоматическая инициализация не происходит.

Как только эта функция получает питание, она начинает накапливать время (текущее значение). Когда релейно-контактная логика доходит до этого таймера, его текущее значение обновляется.



Когда текущее значение равно заданному значению PV или превышает его, включается выход Q. В течение всего времени, когда таймер получает питание, он продолжает накапливать время до тех пор, не будет достигнуто максимальное значение. Когда максимальное значение достигнуто, оно удерживается, и выход Q остается включенным независимо от состояния входа enable.

Если в течение цикла ЦПУ выполняются несколько одинаковых таймеров с одинаковыми адресами, текущие значения таймеров будут одинаковыми.

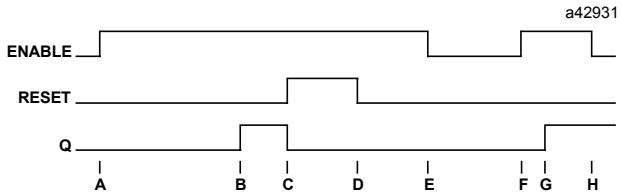
Функции таймера и счетчика
Таймер задержки по включению со сбросом

Параметры функции таймера задержки по включению со сбросом

Вход/ выход	Варианты	Описание
адрес	R	Функция использует три последовательных слова (регистра) памяти %R для хранения следующих данных: • Текущее значение (CV) =слово 1. • Заданное значение (PV) = слово 2. • Управляющее слово = слово 3. Не используйте этот адрес с другими инструкциями. **Осторожно:** Наложение ячеек вызовет неустойчивую работу таймера.
enable	питание	Текущее значение таймера увеличивается, когда подано питание.
R	питание	Когда на вход R подано питание, он устанавливает текущее значение в 0.
PV	I, Q, M, T, G, R, AI, AQ, константа, отсутствие значения	Заданное значение, используемое, когда таймер включен или сбрасывается.
Q	питание, отсутствие питания	Выход Q включается, когда текущее значение таймера больше или равно заданному значению.
время	десятые, сотые или тысячные доли секунды	Время увеличения для младшего бита заданного значения и текущего значения.

Функции таймера и счетчика
Таймер задержки по включению со сбросом

Работа функции таймера задержки по включению со сбросом



A. ENABLE переходит в 1; таймер начинает накапливать значение
B. Текущее значение достигает заданного значения PV; Q включается
C. RESET переходит в 1; Q выключается, накопленное время сбрасывается (CV=0)
D. RESET переходит в 0; таймер снова начинает накапливать значение
E. ENABLE переходит в 0; таймер прекращает накопление. Накопленное время остается тем же
F. ENABLE снова переходит в 1; таймер продолжает накапливать время
G. Текущее время становится равным заданному значению PV; Q включается. Таймер продолжает накапливать время, пока ENABLE не перейдет в 0, RESET не перейдет в 1 или текущее значение не станет равным максимальному времени
H. ENABLE переходит в 0; таймер прекращает накапливать время.

Когда прекращается подача питания на таймер, текущее значение перестает увеличиваться и удерживается. Выход Q, если включен, останется включенным. Когда функция получает питание снова, текущее значение опять увеличивается, начиная с удерживаемого значения. Когда вход R получает питание, текущее значение устанавливается в 0, а выход Q выключается, если PV не равно 0.

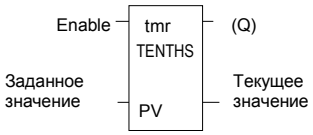
Функции таймера и счетчика
Таймер задержки по включению со сбросом

Пример

В примере записывающий таймер задержки по включению используется для формирования сигнала (%Q0011), который включается через 8.0 секунд после включения %Q0010, и выключается, когда %Q0010 выключается.

Функции таймера и счетчика
Таймер задержки по включению

Таймер задержки по включению (TMR) накапливает время, когда получает питание, и сбрасывается в 0, когда питание пропадает. Время может отсчитываться в десятых (настройка по умолчанию), сотых, или тысячных долях секунды. Диапазон единиц времени составляет от 0 до +32,767. Состояние таймера сохраняется при прерывании питания; при подаче питания автоматическая инициализация не происходит.



Адрес - 3 слова

Когда таймер задержки по включению получает питание, таймер начинает накапливать время (текущее значение). Текущее значение обновляется, когда логика доходит до таймера, для отображения общего прошедшего времени с момента последнего сброса таймера.

Если в течение цикла ЦПУ выполняются несколько одинаковых таймеров с одинаковыми адресами, текущие значения таймеров будут одинаковыми.

Обновление происходит в течение всего времени, пока разрешающая логика остается включенной. Когда текущее значение равно заданному значению PV или превышает его, функция начинает пропускать питание направо. Таймер продолжает накапливать время до тех пор, пока не будет достигнуто максимальное значение. Когда параметр разрешения переходит из 1 в 0, таймер прекращает накапливать время, и текущее значение сбрасывается 0.

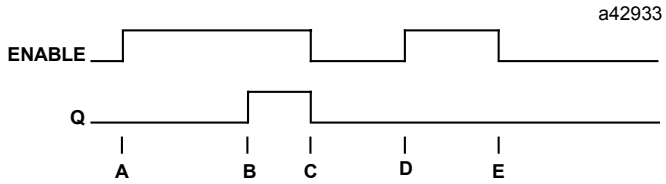
Функции таймера и счетчика
Таймер задержки по включению

Параметры функции таймера задержки по включению

Вход/ выход	Варианты	Описание
адрес	R	Функция использует три последовательных слова (регистра) памяти %R для хранения следующих данных: • Текущее значение (CV) =слово 1. • Заданное значение (PV) = слово 2. • Управляющее слово = слово 3. Не используйте этот адрес с другими инструкциями. **Осторожно:** Наложение ячеек вызовет неустойчивую работу таймера.
enable	питание	Текущее значение таймера увеличивается, когда подано питание. Когда на TMR питание не подается, текущее значение сбрасывается в 0 и выход Q выключается.
PV	I, Q, M, T, G, R, AI, AQ, константа, отсутствие значения	Значение PV копируется в заданое значение таймера, когда на таймер подается питание или он сбрасывается.
Q	питание, отсутствие питания	Выход Q включается, когда на таймер подано питание и текущее значение больше или равно заданному значению.
время	десятые (0.1), сотые (0.01), или тысячные (0.001) доли секунды	Время увеличения для младшего бита заданного значения и текущего значения.

Функции таймера и счетчика
Таймер задержки по включению

Работа функции таймера задержки по включению



A. ENABLE переходит в 1; таймер начинает накапливать время.

B. Текущее значение достигает заданного значения PV; Q включается, и таймер продолжает накапливать время.

C. ENABLE переходит в 0; Q выключается; таймер прекращает накапливать время и текущее время очищается.

D. ENABLE переходит в 1; таймер начинает накапливать время.

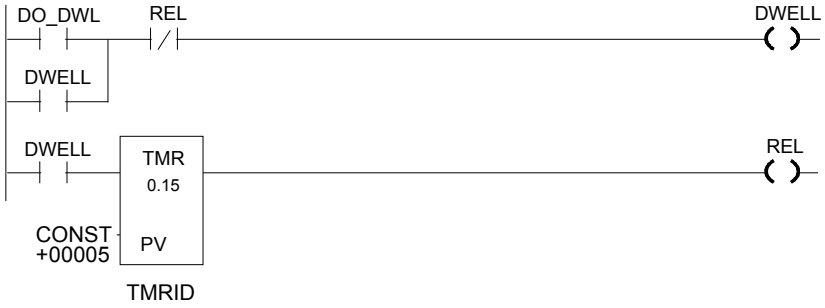
E. ENABLE переходит в 0 до достижения текущим значением заданного значения PV; Q остается выключенным; таймер прекращает накапливать время и текущее время очищается (CV=0).

Функции таймера и счетчика
Таймер задержки по включению

Пример

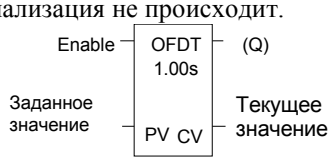
В примере таймер задержки (с адресом) TMRID используется для управления продолжительностью времени, в течение которого катушка включена. Этой катушке назначено имя DWELL . Когда нормально разомкнутый (импульсный) контакт с именем DO_DWL включается, катушка DWELL включается.

Контакт катушки DWELL держит катушку DWELL включенной (когда контакт DO_DWL разомкнут), а также запускает таймер TMRID. Когда TMRID достигает заданного ему значения (0.5 с), катушка REL включается, разрывая подхваченное состояние катушки DWELL. Контакт DWELL прерывает питание TMRID, сбрасывая текущее значение и выключая катушку REL. Теперь цепь готова к следующей импульсной активизации контакта DO_DWL.



Функции таймера и счетчика
Таймер задержки по выключению

Таймер задержки по выключению накапливает время, когда его питание выключено, и сбрасывается в 0, когда питание подается. Время может отсчитываться в десятых (0.1), сотых (0.01), или тысячных (0.001) долях секунды. Диапазон единиц времени составляет от 0 до +32,767. Состояние таймера сохраняется при прерывании питания; при подаче питания автоматическая инициализация не происходит.



Адрес - 3 слова

Когда таймер задержки по выключению получает питание, он пропускает питание направо и устанавливает текущее значение (CV) в 0. Функция использует слово 1 [регистр] для хранения значения CV. Выход остается включенным, пока функция получает питание. Если подача питания на функцию прерывается, она продолжает передавать питание направо, и таймер начинает накапливать время в текущем значении. Таймер задержки по выключению не пропускает питание, если заданное значение отрицательно или равно 0.

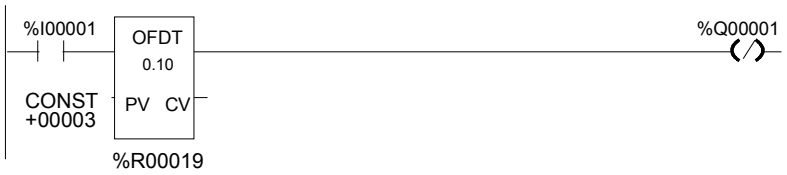
Если в течение цикла ЦПУ выполняются несколько одинаковых таймеров с одинаковыми адресами, текущие значения таймеров будут одинаковыми.

Каждый раз, когда функция вызывается логикой разрешения, установленной в 0, текущее значение обновляется для отображения времени, прошедшего с выключения таймера. Когда текущее значение (CV) равно заданному значению (PV), функция перестает пропускать питание направо, и таймер останавливает накопление. Когда функция снова получает питание, текущее значение устанавливается в 0. Когда таймер используется в программном блоке, вызываемом *не* в каждом цикле, таймер накапливает время между обращениями к программному блоку, пока не будет сброшен. Это означает, что он работает как таймер в программе с циклом, гораздо более медленным, чем таймер в главном программном блоке. Для программных блоков, не активных в течение длительного времени, таймер должен быть запрограммирован, чтобы он позволял наверстать отставание. Например, если таймер в программном блоке сбрасывается, и программный блок не вызывается в течение 4 минут, то при вызове программного блока будет накоплено 4 минуты. Это время используется таймером при наличии питания, до первого сброса таймера.

Функции таймера и счетчика
Таймер задержки по выключению

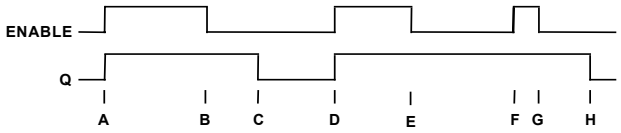
Пример

В примере таймер задержки по выключению используется для выключения выхода (%Q00001) всякий раз, когда включается вход (%I00001). Выход снова включается через три секунды после выключения входа.



Функции таймера и счетчика
Таймер задержки по выключению

Работа функции таймера задержки по выключению



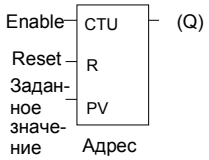
A. И ENABLE и Q переходят в 1; таймер сбрасывается (CV = 0).
B. ENABLE переходит в 0; таймер начинает накапливать время.
C. CV достигает PV; Q выключается, и таймер прекращает накапливать время.
D. ENABLE переходит в 1; таймер сбрасывается (CV = 0).
E. ENABLE переходит в 0; таймер начинает накапливать время.
F. ENABLE переходит в 1; таймер сбрасывается (CV = 0).
G. ENABLE переходит в 0; таймер начинает накапливать время.
H. CV достигает PV; Q выключается, и таймер прекращает накапливать время.

Параметры функции таймера задержки по выключению

Вход/ выход	Варианты	Описание
адрес	R	Функция использует три последовательных слова (регистра) памяти %R для хранения следующих данных: • Текущее значение (CV) =слово 1. • Заданное значение (PV) = слово 2. • Управляющее слово = слово 3. Не используйте этот адрес с другими инструкциями. **Осторожно:** Наложение ячеек вызовет неустойчивую работу таймера.
enable	питание	Текущее значение таймера увеличивается, когда подано питание.
PV	I, Q, M, T, G, R, AI, AQ, константа, отсутствие значения	Значение PV копируется в заданное значение таймера, когда на таймер подается питание или он сбрасывается. Для регистра (%R) ячейки PV, параметр PV указывается как второе слово адреса параметра. Например, при адресе параметра %R0001, для параметра PV будет использоваться %R0002.
Q	питание, отсутствие питания	Выход Q включается, когда текущее значение меньше заданного значения. Состояние Q сохраняется при пропадании питания; при подаче питания автоматическая инициализация не происходит.
время	десятые, сотые или тысячные доли секунды	Время увеличения для младшего бита заданного значения и текущего значения.

Функции таймера и счетчика
Инкрементный счетчик

Функция инкрементного счетчика (CTU) считает вверх до назначенного значения. Диапазон от –0 до +32,767 отсчетов. Когда вход сброса инкрементного счетчика (CTU) включен, текущее значение счетчика устанавливается в 0. При каждом переходе входа разрешения из 1 в 0 текущее значение увеличивается на 1. Текущее значение может быть больше заданного значения PV. Выход включен, когда текущее значение больше или равно заданному значению. Состояние инкрементного счетчика (CTU) сохраняется при пропадании питания; при подаче питания автоматическая инициализация не происходит.



Параметры функции инкрементного счетчика

Вход/ выход	Варианты	Описание
адрес	R	Функция использует три последовательных слова (регистра) памяти %R для хранения следующих данных: • Текущее значение (CV) =слово 1. • Заданное значение (PV) = слово 2. • Управляющее слово = слово 3. Не используйте этот адрес с другим инкрементным счетчиком, декрементным счетчиком или любыми другими инструкциями. **Осторожно:** Наложение ячеек вызовет неустойчивую работу счетчика.
enable	питание	При положительном переходе на входе enable текущий отсчет увеличивается на 1.
R	питание	Когда на вход R подано питание, он устанавливает текущее значение в 0.
PV	I, Q, M, T, G, R, AI, AQ, константа, отсутствие значения	Значение PV копируется в заданное значение счетчика, когда на счетчик подается питание или он сбрасывается.
Q	питание, отсутствие питания	Выход Q включается, когда текущее значение больше или равно заданному значению.

Функции таймера и счетчика
Инкрементный счетчик

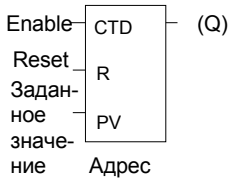
Пример функции инкрементного счетчика

В примере при каждом переходе входа %I0012 из 0 в 1, инкрементный счетчик PRT_CNT увеличивается на 1; внутренняя катушка %M0001 включается, когда будет отсчитано 100 переходов. Когда %M0001 включается, накопленное значение устанавливается в 0.

Функции таймера и счетчика
Декрементный счетчик

Функция декрементного счетчика (CTD) считает вниз от заданного значения. Минимальным заданным значением является 0; максимальным заданным значением является +32767 отсчетов. Минимальным текущим значением является –32768. При сбросе текущее значение счетчика устанавливается в заданное значение PV. Когда вход enable переходит из 0 в 1, текущее значение уменьшается на 1. Выход включается, когда текущее значение меньше или равно 0.

Текущее значение декрементного счетчика сохраняется при пропадании питания; при подаче питания автоматическая инициализация не происходит.



Параметры функции декрементного счетчика

Вход/ выход	Варианты	Описание
адрес	R	Функция использует три последовательных слова (регистра) памяти %R для хранения следующих данных: • Текущее значение (CV) =слово 1. • Заданное значение (PV) = слово 2. • Управляющее слово = слово 3. Не используйте этот адрес с другим инкрементным счетчиком, декрементным счетчиком или любыми другими инструкциями. **Осторожно:** Наложение ячеек вызовет неустойчивую работу счетчика.
enable	питание	При положительном переходе на входе enable текущее значение уменьшается на 1.
R	питание	Когда на вход R подано питание, он устанавливает текущее значение в заданное значение.
PV	I, Q, M, T, G, R, AI, AQ, константа, отсутствие значения	Значение PV копируется в заданное значение счетчика, когда на счетчик подается питание или он сбрасывается.
Q	питание, отсутствие питания	Выход Q включается, когда текущее значение меньше или равно 0.

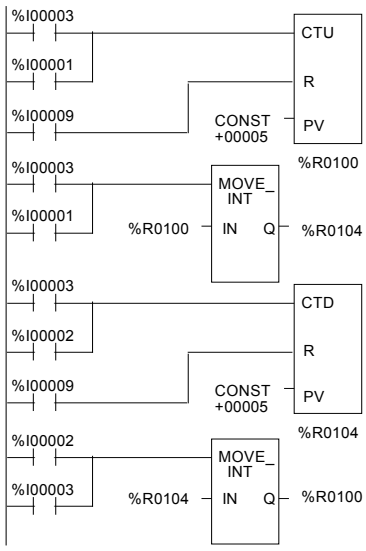
Функции таймера и счетчика
Декрементный счетчик

Пример 1:

В примере декрементный счетчик COUNTP отсчитывает 500 переходов перед включением выхода %Q0005.

Пример 2: Учет частей в зоне временного хранения

Следующий пример показывает, как ПЛК может учитывать количество частей в зоне временного хранения. Здесь используется пара инкрементного и декрементного счетчиков с общим регистром для хранения текущего значения. Когда части поступают в зону хранения, инкрементный счетчик увеличивает текущее значение количества хранящихся частей на 1. Когда части забираются из зоны хранения, декрементный счетчик уменьшается на 1, уменьшая значение хранимого количества на 1. Два счетчика используют различные адреса регистров. Когда регистр сосчитан, его текущее значение необходимо переслать в регистр текущего значения другого счетчика.



Пример использования функций сложения и вычитания для учета хранения приведен в разделе Математических функций.

Глава
11

Функция системного запроса

Эта глава объясняет функцию системного запроса SVCREQ, которая вызывает специальные службы ПЛК. Ниже описаны параметры функции SVCREQ для ЦПУ VersaMax®.

- Номера функций SVCREQ
- Формат функции SVCREQ
- SVCREQ 1: Изменить/прочитать таймер цикла с постоянным временем
- SVCREQ 2: Прочитать времена окон
- SVCREQ 3: Изменить режим и время окна связи с программатором
- SVCREQ 4: Изменить режим и время окна системных коммуникаций
- SVCREQ 6: Изменить/прочитать количество слов в контрольной сумме
- SVCREQ 7: Изменить/прочитать дату и время суток
- SVCREQ 8: Сбросить сторожевой таймер
- SVCREQ 9: Прочитать время с начала цикла
- SVCREQ 10: Прочитать имя папки
- SVCREQ 11: Прочитать идентификатор ПЛК
- SVCREQ 13: Выключить (остановить) ПЛК
- SVCREQ 14: Очистить таблицу ошибок
- SVCREQ 15: Прочитать последнюю запись в таблице ошибок
- SVCREQ 16: Прочитать время с момента включения
- SVCREQ 18: Прочитать статус принудительной установки каналов ввода/вывода
- SVCREQ 23: Прочитать контрольную сумму
- SVCREQ 26/30: Проверить модули ввода/вывода

Номера функций SVCREQ

Каждый системный запрос имеет свой собственный номер функции, приведенный в следующей ниже таблице

Функция #	Описание
1	Изменить/прочитать таймер цикла с постоянным временем
2	Прочитать времена окон
3	Изменить режим и время окна связи с программатором
4	Изменить режим и время окна системных коммуникаций
5	зарезервировано
6	Изменить/прочитать количество слов в контрольной сумме
7	Изменить/прочитать дату и время суток
8	Сбросить сторожевой таймер
9	Прочитать время с начала цикла
10	Прочитать имя папки
11	Прочитать идентификатор ПЛК
12	зарезервировано
13	Выключить ПЛК
14	Очистить таблицы ошибок
15	Прочитать последнюю запись в таблице ошибок
16	Прочитать время с момента включения
17	зарезервировано
18	Прочитать статус принудительной установки каналов ввода/вывода
19-22	зарезервировано
23	Прочитать контрольную сумму
26/30	Проверить модули ввода/вывода
27, 28	зарезервировано
29	Прочитать время нахождения в выключенном состоянии
31-255	зарезервированы

Формат функции SVCREQ

Функция SVCREQ имеет три входа и один выход.

Когда SVCREQ получает питание, ПЛК получает запрос на выполнение функции с указанным номером FNC. Параметры для функции располагаются, начиная с адреса, указанного в PARM. Это начало «блока параметров» функции. Количество 16-разрядных ячеек зависит от используемой функции SVCREQ.

Блоки параметров могут быть использованы как для входных параметров функции, так и для выходных данных, сформированных после выполнения функции. Поэтому, доступ к возвращаемым функцией данным осуществляется по тому же адресу, указанному в PARM.
Функция SVCREQ пропускает питание, если не указан неправильный номер функции, неправильные параметры или недопустимые адреса. Для отдельных функций SVCREQ возможны дополнительные причины неисправностей.

Параметры функции SVCREQ

Вход/Выход	Возможные значения	Описание
enable	питание	Системный запрос выполняется, когда на вход enable подано питание.
FNC	I, Q, M, T, G, R, AI, AQ, константа	Содержит константу или ячейку с номером запрашиваемой службы.
PARM	I, Q, M, T, G, R, AI, AQ	Содержит начальный адрес блока параметров запрашиваемой службы.
ok	питание, отсутствие питания	ОК включен, если функция выполнена без ошибок.

Пример функции SVCREQ

В этом примере, когда вход разрешения %I0001 включен, вызывается функция SVCREQ номер 7 с блоком параметров, начинающимся с %R0001. Выходная катушка %Q0001 включается, если операция выполнена успешно.

SVCREQ 1: Изменить/прочитать таймер цикла с постоянным временем

Используйте SVCREQ 1, чтобы включить или выключить режим цикла с постоянным временем, изменить время цикла, прочитать, включен ли в данный момент режим цикла с постоянным временем или прочитать время цикла.

Блок входных параметров для SCVREQ 1

Блок параметров для этой функции имеет длину 2 слова.

Выключить режим цикла с постоянным временем

Чтобы выключить режим цикла с постоянным временем, вызовите функцию SVCREQ # 1 со следующим блоком параметров:

адрес	0
адрес + 1	Игнорируется

Включить режим цикла с постоянным временем

Чтобы включить режим цикла с постоянным временем, вызовите функцию SVCREQ # 1 со следующим блоком параметров:

адрес	1
адрес + 1	0 или уставка таймера

Примечание: Если таймер должен использовать новую уставку, введите ее во втором слове. Если уставка таймера не должна быть изменена, введите 0 во втором слове. Если уставка таймера еще не существует, ввод 0 приведет к тому, что функция установит выход ОК в выключенное состояние (0).

Изменить время цикла

Чтобы изменить уставку таймера, **не изменяя** выбранный режим, вызовите функцию SVCREQ # 1 со следующим блоком параметров:

адрес	2
адрес + 1	новая уставка таймера

Прочитать текущее состояние цикла с постоянным временем и время

Чтобы прочитать текущий режим цикла и уставку таймера, ничего не изменяя, вызовите функцию SVCREQ # 1 со следующим блоком параметров:

адрес	3
адрес + 1	Игнорируется

Выполнение не удастся, если:

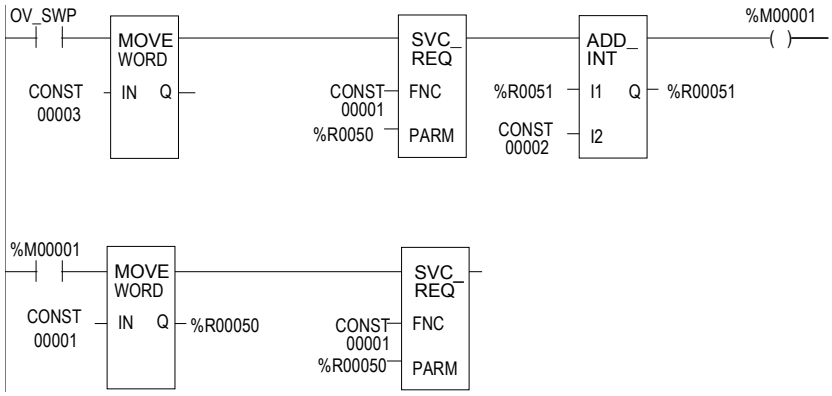
1. В качестве номера запрашиваемой операции введено число, отличное от 0, 1, 2 или 3:
2. Время цикла превышает 500 мс (0,5 с).
3. Режим цикла с постоянным временем включен, а уставка таймера не задана или ее старое значение равно 0.

После выполнения функция возвращает состояние и уставку таймера в те же самые ячейки блока параметров:

	0 = выключен
адрес	1 = включен
адрес + 1	текущая уставка таймера

Пример SVCREQ 1

В этом примере, если включен контакт OV_SWP, читается уставка таймера, она увеличивается на 2 миллисекунды, и новое значение уставки посылается обратно в ПЛК. Блок параметров находится в памяти по адресу %R0050. Поскольку для функций MOVE и ADD требуется три места по горизонтали, программа в примере использует дискретную внутреннюю катушку %M0001 в качестве промежуточного звена для хранения успешного результата выполнения первой строки. В каждом цикле, в котором OV_SVP не включен, %M0001 выключается.



SVCREQ 2: Прочитать времена окон

SVCREQ 2 используется для чтения времен окна связи с программатором и окна системных коммуникаций. Эти окна могут работать в ограниченном режиме или в режиме выполнения до завершения.

Режим	Значение	Описание
Ограниченный режим	0	Время выполнения окна ограничено 6 мс. Окно закрывается когда все задачи выполнены или 6 миллисекунд истекло.
Режим работы до завершения	2	Независимо от установленного определенному окну времени, оно действует до выполнения всех поставленных задач (до 400 миллисекунд).

Окно заблокировано, если значение времени равно 0.

Блок выходных параметров для SVCREQ 2

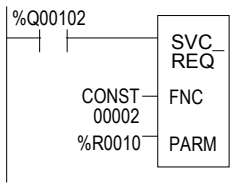
Блок параметров имеет длину 3 слова:

	Старший байт	Младший байт	
адрес	режим	Значение в мс	Окно программатора
адрес + 1	режим	Значение в мс	Окно системных коммуникаций
адрес + 2	должен быть 0	должен быть 0	зарезервирован

Все параметры являются выходными параметрами. Для программирования этой функции не требуется вводить значения в блок параметров .

Пример SVCREQ 2

В следующем примере, когда установлен разрешающий выход %Q00102, ЦПУ помещает значения текущего времени окон в блок параметров, начинающийся с ячейки %R0010.



SVCREQ 3: Изменить режим окна связи с программатором

Используйте SVCREQ 3 для изменения режима окна связи с программатором (ограниченный или выполнение до завершения). Изменение происходит во время следующего за вызовом функции цикла ЦПУ. Время окна не может быть изменено; оно всегда составляет 6мс.

SVCREQ 3 не пропускает питание направо, если режим отличается от 0 (ограниченный) или 2 (выполнение до завершения).

Блок параметров имеет длину 1 слово.

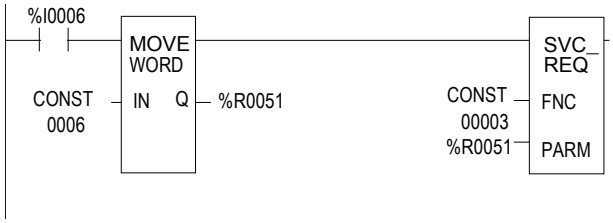
Изменение режима окна связи с программатором

Чтобы изменить окно связи с программатором, введите SVCREQ 3 с этим блоком параметров:

	Старший байт	Младший байт
адрес	режим	6

Пример SVCREQ 3

В следующем примере, когда вход разрешения %I006 переходит в состояние 1, окно связи с программатором разблокировано, и ему назначено значение времени 6 мс. Блок параметров находится в ячейке памяти %R0051.



SVCREQ 4: Изменить режим окна системных коммуникаций

Используйте SVCREQ 4 для изменения режима окна системных коммуникаций (ограниченный или выполнение до завершения). Изменение происходит во время следующего за вызовом функции цикла ЦПУ. Время окна не может быть изменено; оно всегда составляет 6мс.

SVCREQ 4 не пропускает питание направо, если выбран режим, отличный от 0 (ограниченный) или 2 (выполнение до завершения).

Блок параметров имеет длину 1 слово.

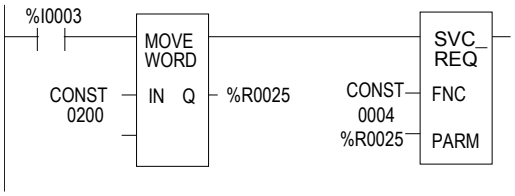
Изменение режима окна системных коммуникаций

Чтобы изменить окно связи с программатором, введите SVCREQ 4 с этим блоком параметров:

	Старший байт	Младший байт
адрес	режим	6

Пример SVCREQ 4

В следующем примере, когда вход разрешения %I0003 включен, режим окна системных коммуникаций меняется на режим выполнения до завершения. Блок параметров находится в ячейке памяти %R0025.



SVCREQ 6: Изменить/прочитать количество слов в контрольной сумме

Используйте SVCREQ 6, чтобы прочитать или изменить количество слов в контрольной сумме. Функция выполняется успешно, если в качестве запрашиваемой операции не введено число, отличное от 0 или 1.

Форматы блока параметров для SVCREQ 6

Блок параметров имеет длину 2 слова.

Чтобы прочитать количество слов, первое слово блока параметров должно содержать 0:

адрес	0 (чтение количества слов)
адрес + 1	игнорируется

Функция возвращает текущее количество слов во второе слово блока параметров.

адрес	0
адрес + 1	текущее количество слов

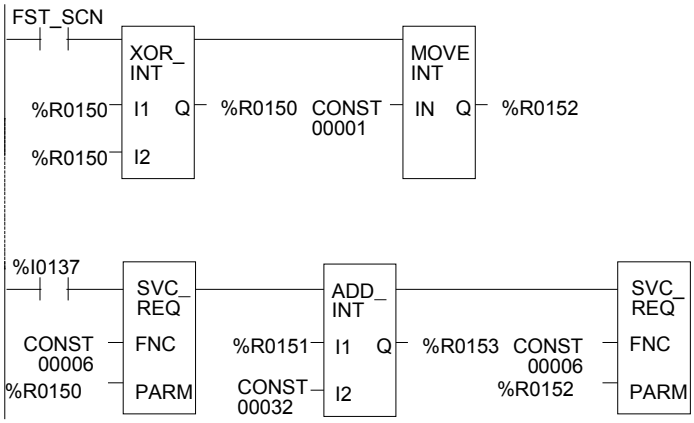
Чтобы изменить количество слов, первое слово блока параметров должно содержать 1:

адрес	1 (изменение количества слов)
адрес + 1	новое количество слов(0 или 32)

ПЛК изменит количество слов в контрольной сумме на новое значение.

Пример SVCREQ 6

В примере, когда разрешающий контакт FST_SCN замкнут, формируются блоки параметров для функции контрольной суммы. Позже в программе, если вход %I0137 включается, SVCREQ читает количество слов в контрольной сумме. Блок параметров для функции чтения находится в ячейках %R0150-151. Функция ADD добавляет 32 к текущему количеству слов в регистре %R0151 и помещает результат в ячейку %R0153. Блок параметров для функции изменения находится в ячейках %R00152-153. Второй SVCREQ изменяет количество слов на указанное в ячейке %R0153.



SVCREQ 7: Изменить/прочитать дату и время суток

Используйте SVCREQ 7, чтобы прочитать или изменить дату и время суток в ПЛК. Данные могут быть в двоично-десятичном формате или в формате ASCII. Запись года возможна двумя или четырьмя цифрами. Функция выполняется успешно, если в качестве запрашиваемой операции не введено число, отличное от 0 (чтение) или 1 (изменение), или не указан ошибочный формат данных, или данные не передаются в неизвестном формате.

Формат блока параметров для SVCREQ 7

Для функций даты/времени длина блока параметров зависит от формата данных. Блок данных может быть двоично-десятичным или ASCII. Двоично-десятичный формат требует 6 слов; сжатый ASCII требует 12 слов (13 слов для записи года 4 цифрами). Для обоих типов данных:

- Часы сохраняются в 24-часовом формате.
- День недели - цифровое значение от 1 (воскресенье) до 7 (суббота).

	Запись года 2 цифрами	**Запись года 4 цифрами**
адрес	0 = чтение времени и даты 1 = установка времени и даты	0 = чтение времени и даты 1 = установка времени и даты
адрес + 1	1 = двоично-десятичный формат 3 = сжатый ASCII формат	81 = двоично-десятичный формат 83 = сжатый ASCII формат
адрес + 2 до конца	данные	данные

Слова с 3 до конца блока параметров содержат выходные данные, возвращенные функцией чтения, или новые данные, обеспечиваемые функцией изменения. В обоих случаях, формат этих слов данных один и тот же. При чтении даты и времени, слова (адрес + 2) до конца блока параметров на входе игнорируются.

Содержание блока параметров SVCREQ 7: Двоично-десятичный формат

В двоично-десятичном формате каждая единица времени и даты занимает один байт, поэтому блок параметров состоит из шести слов.

Запись года 2 цифрами

Последний байт шестого слова не используется. При установке даты и времени этот байт игнорируется; при чтении даты и времени функция возвращает 00.

Формат блока параметров:
Старший байт Младший байт

1 = изменить или 0 = читать		адрес
1 двоично-десятичный формат		адрес + 1
месяц	год	адрес + 2
часы	день месяца	адрес + 3
секунды	минуты	адрес + 4
(ноль)	день недели	адрес + 5

Пример:
Чтение даты и времени в двоично-десятичном формате (Вс., 3 Июля, 1998, 14:45:30)

0 (читать)	
1 двоично-десятичный формат	
07 (Июль)	98 (год)
14 (часы)	03 (день)
30 (секунды)	45 (минуты)
00	06 (Пятница)

Запись года 4 цифрами

Блок параметров состоит из шести слов. Используются все байты.

Формат блока параметров:
Старший байт Младший байт

1 = изменить или 0 = читать		адрес
81h (двоично-десятичный формат, 4 цифры)		адрес + 1
год	год	адрес + 2
день месяца	месяц	адрес + 3
минуты	часы	адрес + 4
день недели	секунды	адрес + 5

Пример:
Чтение даты и времени в двоично-десятичном формате (Вс., 3 Июля, 1998, 14:45:30)

00	00 (читать)
00	81h (двоично-десятичный формат, 4 цифры)
19 (год)	98 (год)
03 (день)	07 (Июль)
45 (минуты)	14 (часы)
06 (Пятница)	30 (секунды)

Содержание блока параметров SVCREQ 7: Сжатый ASCII формат

В сжатом ASCII формате каждая цифра времени и даты занимает байт в формате ASCII. Интервалы и двоеточия встроены в дату для сохранения форматирования при печати или отображении на дисплее. ASCII формату требуются 12 слов в блоке параметров (13 слов при записи года 4 цифрами).

Запись года 2 цифрами

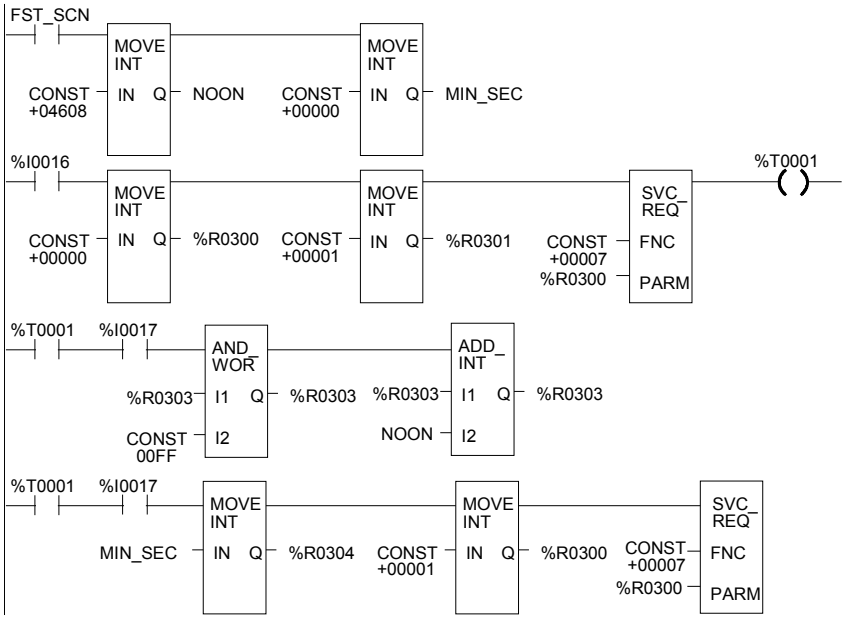
Формат блока параметров: Старший байт	**Младший байт**		**Пример: Чтение даты и времени в сжатом ASCII формате (Пн, 5 Окт. , 1998, 11:13:00)**	
1 = изменить или 0 = читать		адрес	0 (читать)	
3 (ASCII формат)		адрес + 1	3 (ASCII формат)	
год	год	адрес + 2	38 (8)	39 (9)
месяц	(пробел)	адрес + 3	31 (1)	20 (пробел)
(пробел)	месяц	адрес + 4	20 (пробел)	30 (0)
день месяца	день месяца	адрес + 5	35 (5)	30 (первый 0)
часы	(пробел)	адрес + 6	31 (1)	20 (пробел)
: (двоеточие)	часы	адрес + 7	3A (:)	31 (1)
минуты	минуты	адрес + 8	33 (3)	31 (1)
секунды	:	адрес + 9	30 (0)	3A (:)
(пробел)	секунды	адрес + 10	20 (пробел)	30 (0)
день недели	день недели	адрес + 11	32 (2: Пн.)	30 (первый 0)

Запись года 4 цифрами

Формат блока параметров: Старший байт	**Младший байт**		**Пример: Чтение даты и времени в сжатом ASCII формате (Пн, Окт. 5, 1998, 11:13:00)**	
1 = изменить или 0 = читать		адрес	0 (читать)	
83h (ASCII формат, 4 цифры)		адрес + 1	83h (ASCII формат, 4 цифры)	
год (сотни)	год (тысячи)	адрес + 2	39 (9)	31 (1)
год (единицы)	год (десятки)	адрес + 3	38 (8)	39 (9)
месяцы (десятки)	(пробел)	адрес + 4	31 (1)	20 (пробел)
(пробел)	месяцы (единицы)	адрес + 5	20 (пробел)	30 (0)
день месяца (единицы)	день месяца (десятки)	адрес + 6	35 (5)	30 (первый 0)
часы (десятки)	(пробел)	адрес + 7	31 (1)	20 (пробел)
: (двоеточие)	часы (единицы)	адрес + 8	3A (:)	31 (1)
минуты (единицы)	минуты (десятки)	адрес + 9	33 (3)	31 (1)
секунды (десятки)	: (двоеточие)	адрес + 10	30 (0)	3A (:)
(пробел)	секунды (единицы)	адрес + 11	20 (пробел)	30 (0)
день недели (единицы)	день недели (десятки)	адрес + 12	32 (2: Пн.)	30 (первый 0)

Пример SVCREQ 7

В примере блок параметров для даты и времени формируется предыдущей логикой. Запрашиваются текущие дата и время, затем часы устанавливаются на 12 часов дня с использованием двоично-десятичного формата. Блок параметров начинается с ячейки памяти %R0300. Массив NOON, содержащий значения 12, 0 и 0, сформирован в программе ранее. (Массив NOON также должен содержать данные из %R0300.) Двоично-десятичный формат требует шесть последовательных ячеек памяти для блока параметров.



SVCREQ 8: Сбросить сторожевой таймер

Используйте SVCREQ 8 для сброса сторожевого таймера во время цикла. Обычно, при наступлении тайм-аута сторожевого таймера, ПЛК выключается без предупреждения. SVCREQ 8 позволяет сторожевому таймеру не прерывать работу в течение задачи, занимающей много времени (например, при ожидании ответа от коммуникационной линии).

Предостережение

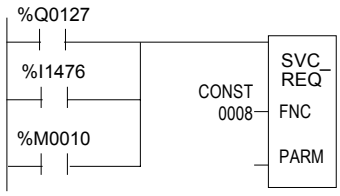
Убедитесь, что сброс сторожевого таймера не окажет неблагоприятного воздействия на управляемый процесс.

Формат блока параметров для SVCREQ 8

Эта функция не имеет блока параметров.

Пример SVCREQ 8

В этом примере подача питания через разрешающий выход %Q0027, или вход %I1476, или внутреннюю катушку %M00010 вызовет сброс сторожевого таймера.



SVCREQ 9: Прочитать время с начала цикла

Используйте SVCREQ 9, чтобы прочитать время в миллисекундах, прошедшее с начала цикла. Формат данных - 16-битовое целое число без знака.

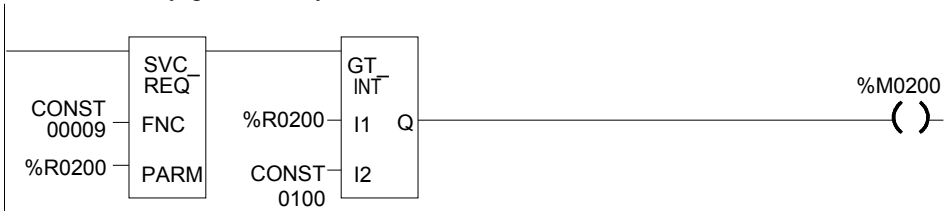
Формат блока выходных параметров для SVCREQ 9

Блок параметров является только выходным блоком; он имеет длину одно слово.

адрес	время, прошедшее с начала цикла

Пример SVCREQ 9

В следующем примере, читается время, прошедшее с начала цикла, и всегда помещается в ячейку памяти %R0200. Если оно больше, чем 100мс, включается внутренняя катушка %M0200.



SVCREQ 10: Прочитать имя папки

Используйте SVCREQ 10, чтобы прочитать имя текущей папки.

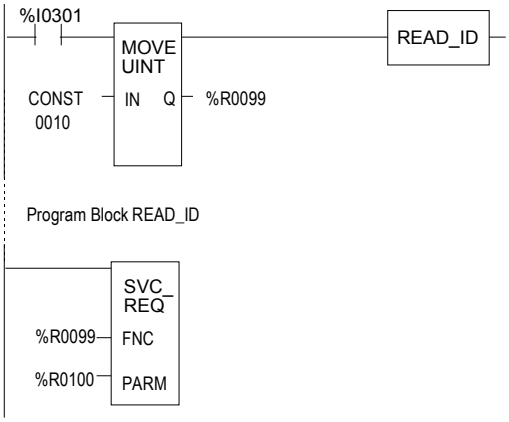
Формат блока выходных параметров для SVCREQ 10

Блок параметров имеет длину четыре слова. Он возвращает восемь символов ASCII; последний символ - нулевой (00h). Если имя программы имеет менее семи символов, к концу добавляются нулевые символы.

	Младший байт	Старший байт
адрес	символ 1	символ 2
адрес + 1	символ 3	символ 4
адрес + 2	символ 5	символ 6
адрес + 3	символ 7	00

Пример SVCREQ 10

В этом примере, когда разрешающий вход %I0301 включается, в ячейку памяти %R0099 загружается значение 10, которое является кодом функции чтения имени папки. Затем вызывается программный блок READ_ID для чтения имени папки. Блок параметров расположен по адресу %R0100.



SVCREQ 11: Прочитать идентификатор ПЛК

Используйте SVCREQ 11, чтобы прочитать имя ПЛК, выполняющего программу.

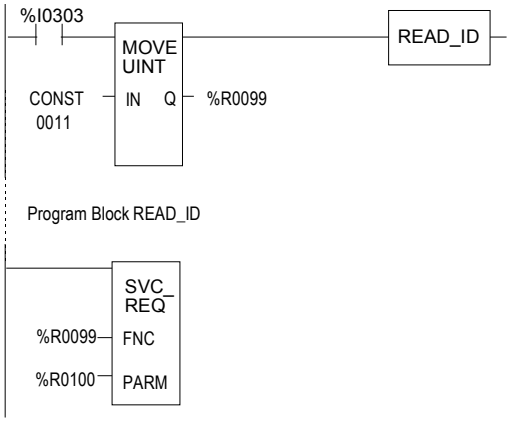
Формат блока выходных параметров для SVCREQ 11

Блок параметров имеет длину четыре слова. Он возвращает восемь символов ASCII; последний символ - нулевой (00h). Если идентификатор ПЛК имеет менее семи символов, к концу добавляются нулевые символы.

	Младший байт	**Старший байт**
адрес	символ 1	символ 2
адрес + 1	символ 3	символ 4
адрес + 2	символ 5	символ 6
адрес + 3	символ 7	00

Пример SVCREQ 11

В этом примере, когда разрешающий вход %I0302 включается, в ячейку памяти %R0099 загружается значение 11, которое является кодом функции чтения идентификатора ПЛК. Затем вызывается программный блок READ_ID для чтения идентификатора. Блок параметров расположен по адресу %R0100.



SVCREQ 13: Выключить (остановить) ПЛК

Используйте SVCREQ 13, чтобы остановить ПЛК *в конце следующего цикла*. Все выходы переходят в присвоенные им состояния по умолчанию в начале следующего цикла ПЛК. В таблицу ошибок ПЛК заносится информационная запись "Shut Down PLC" (Выключение ПЛК). Сканирование ввода-вывода продолжается, как было сконфигурировано.

Блок параметров для SVCREQ 13

Эта функция не имеет блока параметров.

Пример SVCREQ 13

В примере, SVCREQ 13 выполняется, когда возникает ошибка “Loss of I/O Module“ (потеря модуля ввода-вывода). Вход PARM не используется.

Этот пример использует оператор JUMP, чтобы перейти к концу программы для принудительного выключения при успешном выполнении функции выключения ПЛК. Операторы JUMP и LABEL необходимы, т. к. переход в режим Stop не произойдет до окончания цикла, в котором выполняется функция.

SVCREQ 14: Очистить таблицу ошибок

Используйте SVCREQ 14, чтобы очистить таблицу ошибок ПЛК или таблицу ошибок ввода-вывода. Выход SVCREQ включен, если в качестве запрашиваемой операции не введено число, отличное от 0 или 1.

Блок входных параметров для SVCREQ 14

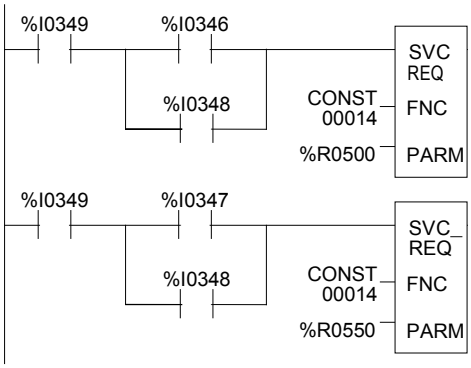
Блок параметров для этой функции имеет длину 1 слово. Это блок только входных параметров. Блок выходных параметров отсутствует.

0 = очистить таблицу ошибок ПЛК.
1 = очистить таблицу ошибок ввода-вывода.

Пример SVCREQ 14

В примере, таблица ошибок ПЛК очищается, когда входы %I0346 и %I0349 включены. Когда включены входы %I0347 и %I0349, очищается таблица ошибок ввода-вывода. Когда включены входы %I0348 и %I0349, очищаются обе таблицы.

Блок параметров для таблицы ошибок ПЛК расположен по адресу %R0500; для таблицы ошибок ввода-вывода блок параметров расположен по адресу %R0550. Оба блока параметров формируются в программе ранее.



SVCREQ 15: Прочитать последнюю запись в таблице ошибок

Используйте SVCREQ 15, чтобы прочитать последнюю запись в таблице ошибок ПЛК или в таблице ошибок ввода-вывода. Выход SVCREQ включен, если в качестве запрашиваемой операции не введено число, отличное от 0 или 1, и таблица ошибок не пуста.

Блок входных параметров для SVCREQ 15

Блок параметров для этой функции имеет длину 22 слов. Блок входных параметров имеет следующий формат:

	Запись года 2 цифрами	**Запись года 4 цифрами**
адрес	0 = прочитать таблицу ошибок ПЛК. 1 = прочитать таблицу ошибок ввода-вывода.	8 = прочитать таблицу ошибок ПЛК. 9 = прочитать таблицу ошибок ввода-вывода.

Формат блока выходных параметров зависит от того, читает ли функция данные из таблицы ошибок ПЛК или из таблицы ошибок ввода-вывода.

	Выходной формат таблицы ошибок ПЛК			Выходной формат таблицы ошибок ввода-вывода	
	Старший байт	**Младший байт**		**Старший байт**	**Младший байт**
	0			1	
	резерв	длинное/короткое	адрес + 1	тип памяти	длинное/короткое
	резерв	резерв	адрес + 2	Смещение	
	слот	Крейт	Адрес + 3	слот	Крейт
	задача		адрес + 4	модуль	шина
	действие ошибки	группа ошибки	адрес + 5	точка	
	код ошибки		адрес + 6	действие ошибки	группа ошибки
	специфические данные ошибки		адрес + 7	тип ошибки	категория ошибки
			адрес + 8	специфические данные ошибки	описание ошибки
			до		
			адрес + 18		
	минуты	секунды	адрес + 19	минуты	секунды
	день месяца	час	адрес + 20	день месяца	час
Запись года 2 цифрами	год	месяц	адрес + 21	год	месяц
или					
Запись года 4 цифрами	резерв	месяц	адрес + 21	резерв	месяц
Формат	год		адрес + 22	год	

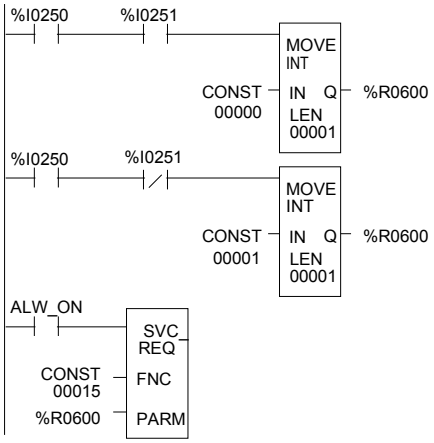
Значение параметра длинное/короткое

Первый байт слова адрес +1 содержит число, указывающее длину специфических данных ошибки в записи ошибки. Возможные значения этого параметра:

Таблица ошибок ПЛК (PLC Fault Table)	00 = 8 bytes (короткое) 01 = 24 bytes (длинное)
Таблица ошибок ввода-вывода	02 = 5 bytes (короткое) 03 = 21 bytes (длинное)

Пример SVCREQ 15

Когда оба входа %I0250 и %I0251 включены, первая функция Move помещает 0 (читать таблицу ошибок ПЛК) в блок параметров для SVCREQ 15. Когда вход %I0250 включен, а вход %I0251 выключен, инструкция Move помещает 1 (читать таблицу ошибок ввода-вывода) в блок параметров для SVCREQ. Блок параметров расположен по адресу %R0600.



SVCREQ 16: Прочитать время с момента включения

Используйте SVCREQ 16, чтобы прочитать время с момента включения системы. Время с момента включения измеряет время, прошедшее с момента подачи питания на ПЛК.

Блок выходных параметров для SVCREQ 16

Эта функция имеет только блок выходных параметров. Его длина составляет 3 слова.

адрес	секунды с момента включения питания (младший разряд)
адрес + 1	секунды с момента включения питания (старший разряд)
адрес + 2	100 микросекундные отсчеты

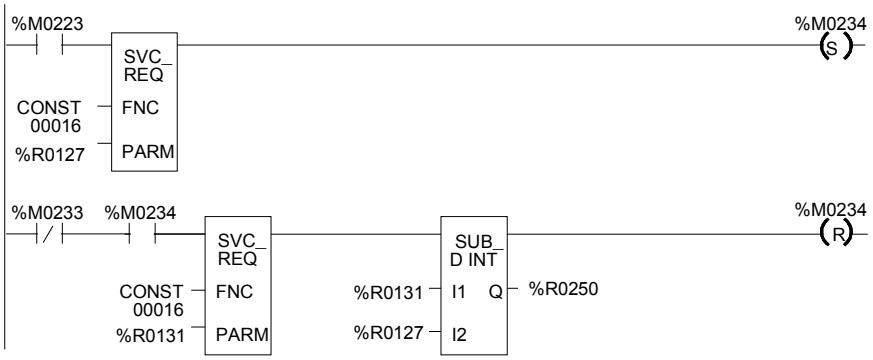
Первые два слова содержат прошедшее время в секундах. В последнем слове содержится количество 100 микросекундных отсчетов в текущей секунде.

Пример SVCREQ 16

В примере, когда включена внутренняя катушка %M0233, SVCREQ с блоком параметров, расположенным по адресу %R0127, читает время, прошедшее с момента включения системы и устанавливает внутреннюю катушку %M0234. Когда катушка %M0233 выключается, SVCREQ с блоком параметров, расположенным по адресу %R0131, читает прошедшее время снова.

Функция вычитания находит разность между первым и вторым значениями, сохраненными в блоках параметров SVCREQ. Вычитание игнорирует 100 микросекундные отсчеты.

Разность между двумя значениями помещается в ячейку памяти %R0250.



SVCREQ 18: Прочитать статус принудительной установки каналов ввода/вывода

Используйте SVCREQ 18, чтобы проверить наличие принудительной установки каких-либо ячеек памяти %I и %Q ЦПУ.

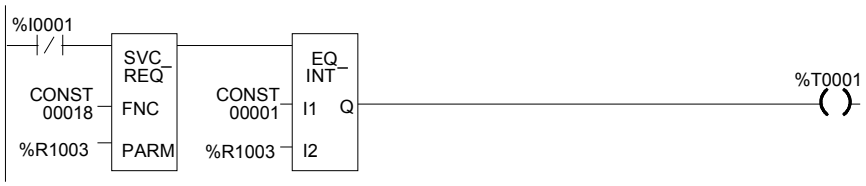
Блок выходных параметров для SVCREQ 18

Эта функция имеет только блок выходных параметров. Его длина составляет 1 слово.

адрес	0 = Принудительная установка отсутствует. 1 = Принудительная установка присутствует.

Пример SVCREQ 18

Следующий SVCREQ считывает статус принудительной установки каналов ввода/вывода в ячейку памяти %R1003. Функция проверки равенства сравнивает значение регистра %R1003 с 1 (константа). Если они равны, функция проверки равенства включает выход %T0001.



SVCREQ 23: Прочитать контрольную сумму

Используйте SVCREQ 23, чтобы прочитать контрольную сумму прикладной программы и конфигурации. Выход SVCREQ всегда включен, если на вход разрешения подано питание.

Блок выходных параметров для SVCREQ 23

Для этой функции блок выходных параметров имеет длину 12 слов в следующем формате:

Первые два параметра блока выходных параметров показывают достоверность контрольных сумм программы и конфигурации. (Контрольные суммы программы могут быть недостоверны во время сохранения в режиме Run.)

адрес	Достоверность контрольной суммы программы (0 = недостоверна, 1 = достоверна)
адрес + 1	Достоверность контрольной суммы конфигурации (0 = недостоверна, 1 = достоверна)
адрес + 2	Количество программных блоков (включая _MAIN)
адрес + 3	Размер пользовательской программы в байтах (данные типа DWORD)
адрес + 5	Аддитивная контрольная сумма программы
адрес + 6	Циклическая контрольная сумма программы (CRC) (данные типа DWORD)
адрес + 8	Размер конфигурационных данных в байтах
адрес + 9	Аддитивная контрольная сумма конфигурации
адрес + 10	Циклическая контрольная сумма конфигурации (CRC) (данные типа DWORD)

Пример SVCREQ 23

В примере, когда включен вход %I0251, информация о контрольной сумме помещается в блок параметров по адресу %R0050 и включается выходная катушка (%Q0001).

SVCREQ 26/30: Проверить модули ввода/вывода

Используйте SVCREQ 26 и 30, чтобы проверить, соответствуют ли установленные модули запрограммированной конфигурации. Если нет, эти SVCREQ помещают соответствующие ошибки добавления, потери и несоответствия модулей в таблицы ошибок ПЛК и/или ввода/вывода. SVCREQ 26 и 30 выполняют одну и туже функцию.

Чем больше ошибок конфигурации, тем дольше выполняются эти SVCREQ.

У этих SVCREQ нет блока параметров. Они всегда пропускают питание на выход.

Пример SVCREQ 26

В примере, когда включен выход %I0251, SVCREQ проверяет установленные модули и сравнивает их с запрограммированной конфигурацией. Выход %Q0001 включается после завершения SVCREQ.

SVCREQ 29: Прочитать время нахождения в выключенном состоянии

Используйте SVCREQ 29, чтобы прочитать время, прошедшее между последним выключением питания и последним включением питания. Если сторожевой таймер сработал до выключения питания, ПЛК не может подсчитать время нахождения в выключенном состоянии, поэтому значение времени устанавливается в 0.

Выход SVCREQ всегда включен.

Блок выходных параметров для SVCREQ 29

Эта функция имеет только блок выходных параметров. Блок параметров имеет длину 3 слова.

адрес	секунды с момента выключения питания (младший разряд)
адрес + 1	секунды с момента выключения питания (старший разряд)
адрес + 2	ноль

Первые два слова содержат время в секундах, прошедшее с момента выключения питания. Последнее слово всегда 0.

Пример SVCREQ 29

В примере, когда включен вход %I0251, время нахождения в выключенном состоянии помещается в блок параметров, начинающийся с ячейки %R0050. Выходная катушка (%Q0001) включена.

Глава
12

Протоколы Serial I/O / SNP / RTU

Эта глава описывает возможности Serial I/O ЦПУ VersaMax®, которые могут использоваться для управления функциями чтения/записи одного из портов ЦПУ, непосредственно из прикладной программы.

Эта глава содержит также инструкции по использованию функции COMMREQ для конфигурирования последовательных портов ЦПУ для работы по протоколам SNP, RTU или Serial I/O.

- Формат функции COMMREQ
- Конфигурирование последовательных портов с помощью функции COMMREQ
  - Работа RTU слэйв/SNP слэйв с подключенным программатором
  - Командный блок COMMREQ для конфигурирования протокола SNP
  - Блок данных COMMREQ для конфигурирования протокола RTU
  - Блок данных COMMREQ для конфигурирования протокола Serial I/O
- Команды COMMREQ для протокола Serial I/O
  - Инициализация порта
  - Настройка входного буфера
  - Очистка входного буфера
  - Чтение состояния порта
  - Управление портом
  - Отмена выполнения
  - Автонабор
  - Запись байт:
  - Чтение байт
  - Чтение потока данных

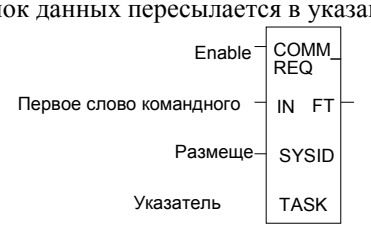
Подробная информация по протоколам RTU и SNP приведена в документе *Serial Communications User's Manual* (GFK-0582).

Формат функции запроса связи

Протокол Serial I/O реализуется с помощью функций запроса связи (COMMREQ). Функции протокола, такие как передача символа через последовательный порт или ожидание входного символа, реализуются с помощью функционального блока COMMREQ. В CPUE05 протокол Serial I/O не может использоваться портом 1 (Port 1), если этот порт сконфигурирован или переведен в режим работы монитора станции.

COMMREQ требует, чтобы все его управляющие данные были размещены в правильной последовательности (в командном блоке) в памяти ЦПУ до выполнения. COMMREQ должен быть вызван контактом импульсной катушки для предотвращения многократной передачи данных. Для пересылки слов и создания командного блока в таблице регистров следует использовать несколько команд Block Move (BLKMV).

Функция COMMREQ имеет три входа и один выход. Когда функция получает питание, командный блок данных пересылается в указанный модуль.



Параметры функции COMMREQ

Вход/ Выход	Варианты	Описание
enable	питание	Запрос связи выполняется, когда функция включена.
IN	R, AI, AQ	IN содержит первое слово командного блока.
SYSID	I, Q, M, T, G, R, AI, AQ, константы	SYSID содержит номер крейта (старший значащий байт) и номер слота (младший значащий байт) устройства. Для ЦПУ, SYSID должен определить крейт/слот - 0.
TASK	R AI, AQ, константа	TASK указывает порт, назначенный для выполнения функции: TASK 19 для порта 1 TASK 20 для порта 2
FT	питание, отсутствие питания	Выход FT включается, если при выполнении COMMREQ обнаружены ошибки: • Отсутствует указанный целевой адрес (SYSID). • Указанная задача не применима для устройства (TASK). • Длина данных равна 0. • Адрес указателя состояния устройства (в командном блоке) не существует.

Командный блок для функции COMMREQ

Командный блок начинается с адреса, указанного в параметре IN функции COMMREQ. Размер командного блока зависит от количества данных, передаваемых устройству.

Командный блок содержит данные обмена с другим устройством и информацию, касающуюся выполнения функции COMMREQ. Командный блок имеет следующую структуру:

Адрес	Длина (в словах)
адрес + 1	Флаг Wait/No Wait (ждать/не ждать)
адрес + 2	Область памяти статуса
адрес + 3	Смещение области статуса
адрес + 4	Значение тайм-аута простоя
адрес + 5	Максимальное время связи
адрес + 6 - адрес + 133	Блок данных

Пример функции COMMREQ

В примере, когда %M0021 находится в состоянии 1, командный блок, начинающийся с регистра %R0032, передается в порт 2 (TASK = 20) ЦПУ (крейт 0, слот 0). Если при обработке COMMREQ происходит ошибка, бит %Q0110 устанавливается в 1.

Конфигурирование последовательных портов с помощью функции COMMREQ

В приведенных ниже таблицах указаны значения командного блока, требуемые для настройки последовательного порта для работы по протоколам SNP, RTU и Serial I/O. Все значения приведены в шестнадцатиричном виде, если не указано иное. Команды BLKMV, используемые для формирования командного блока, описаны в примере.

Следует отметить, что в функции COMMREQ конфигурирования порта RTU и Serial I/O добавлены два параметра: задержка между приемом и передачей и задержка отсутствия сигнала RTS. Если эти параметры включены в COMMREQ, длина блока данных должна быть равна 12H. Если используется значение 10H, функция COMMREQ будет выполняться, однако задержки между приемом и передачей и отсутствия сигнала RTS не будут распознаны. Также важно заметить, что, если функция COMMREQ, содержащая задержку между приемом и передачей и задержку отсутствия сигнала RTS, передается в ЦПУ, не поддерживающее эти задержки, ЦПУ примет и выполнит COMMREQ, но будет игнорировать задержку между приемом и передачей, задержку отсутствия сигнала RTS и задержку переключения (т. е., в этом случае, задержка переключения будет игнорироваться только для протоколов RTU и Serial I/O).

Примечание: Возможно использование как старой (длина 10H), так и новой (длина 12H) формы COMMREQ. Новые параметры поддерживает только новая форма.

Синхронизация

Если COMMREQ конфигурирования порта передан в последовательный порт, к которому в данный момент подключен мастер SNP/SNPX (например, программатор), конфигурация последовательного порта, указанная COMMREQ, не вступит в силу до тех пор, пока ЦПУ не определит потерю мастера SNP/SNPX. Это происходит через сконфигурированный промежуток времени T3 после отключения мастера. Статусное слово COMMREQ конфигурирования порта обновляется сразу, как только ЦПУ проверит, допустима ли указанная конфигурация. Это означает, что значение успешного прохождения COMMREQ может быть возвращено COMMREQ конфигурирования порта до действительного вступления в силу указанной конфигурации.

Передача другого COMMREQ в тот же порт

После инсталляции нового протокола последовательного порта, прикладная программа должна ждать, по крайней мере, 2 секунды плюс

сконфигурированное время T3 перед посылкой порту каких-либо COMMREQ по этому протоколу. Это правило применяется к новому протоколу порта, инсталлированному путем сохранения новой конфигурации оборудования или через COMMREQ конфигурирования порта. Если порт сконфигурирован для использования протокола Serial I/O, этот период ожидания должен следовать также за каждым переходом ЦПУ из режима Stop в режим Run.

Недопустимые комбинации конфигурации порта

Конфигурации обоих портов должны быть совместимы. Один порт должен быть доступен для подключения программатора ПЛК.

Следующие комбинации недопустимы для ЦПУ:

Порт 1	Порт 2
Disabled	Disabled
Disabled	Serial I/O (Переключатель режимов ЦПУ Run/Stop не используется)
Serial I/O (Переключатель режимов ЦПУ Run/Stop не используется)	Disabled
Serial I/O (Переключатель режимов ЦПУ Run/Stop не используется)	Serial I/O (Переключатель режимов ЦПУ Run/Stop не используется)
Station Manager	Disabled
Station Manager	Serial I/O (Переключатель режимов ЦПУ Run/Stop не используется)

Работа в режиме RTU слэйв/SNP слэйв с подключенным программатором

Программатор (устройство SNP/SNPX) может быть подключен к порту 1 или порту 2, несмотря на то, что порт находится в режиме RTU слэйв. При многоточечных подключениях ЦПУ должен быть сконфигурирован так, чтобы использовать соответствующий ID ПЛК. Заметьте, что при многоточечном SNP подключении к порту, в данный момент сконфигурированному для работы в режиме RTU, SNP ID, связанный с настройками ЦПУ, должен соответствовать ID многоточечного подключения.

Чтобы быть распознанным, программатор должен использовать те же самые параметры последовательного соединения (скорость, паритет, количество стоповых битов и т. д.), что и протокол RTU слэйв, действующий в настоящий момент.

Когда ЦПУ опознает программатор, ЦПУ делает активным протокол SNP слэйв вместо RTU слэйв. Значения SNP ID, времени переключения модема и времени простоя по умолчанию для этого нового сеанса связи SNP слэйв берутся из настроек конфигурации ЦПУ, а не из конфигурации порта 1 или порта 2. Соединение должно быть установлено в течение 12 секунд. Если подключение программатора разрешено, может осуществляться обычная связь с программатором. (Невозможность установления связи программатором в течение 12 секунд считается потерей связи с программатором).

Программатор может переслать новый протокол через конфигурацию или через COMMREQ настройки последовательного порта. (COMMREQ, не поддерживаемые протоколом SNP слэйв, игнорируются). Если получен новый протокол, он не вступит в силу до тех пор, пока не будет отключен программатор.

После отключения программатора выполняется небольшая задержка (равная сконфигурированному SNP тайм-ауту T3) перед тем, как ЦПУ определит отсутствие программатора. В течение этого времени порт сообщения не обрабатывает. ЦПУ определяет отключение программатора по истечению тайм-аута протокола SNP слэйв. Таким образом, важно соблюдать осторожность при отключении тайм-аутов, используемых протоколом SNP слэйв.

Когда ЦПУ определяет отключение, оно восстанавливает протокол RTU слэйв, если не был получен новый протокол. В этом случае ЦПУ устанавливает новый протокол.

Пример

1. Порт 1 работает по протоколу RTU слэйв на скорости 9600 бод.
2. Программатор подключен к порту 1. Программатор использует скорость 9600 бод.

3. ЦПУ устанавливает для порта 1 протокол SNP слэйв и программатор связывается обычным образом.
4. Программатор записывает новую конфигурацию для порта 1. Новая конфигурация устанавливает порту протокол SNP слэйв на скорости 4800 бод (это не вступит в силу, пока не прервется связь между портом и программатором).
5. Когда связь между ЦПУ и программатором прервется, вступит в силу новая конфигурация.

Пример командного блока COMMREQ конфигурирования протокола SNP

	Значения	Содержание
Адрес	10H	Длина блока данных
Адрес + 1	0 = No Wait	Флаг WAIT/NOWAIT (ждать/не ждать)
Адрес + 2	0008 = %R, память регистров	Тип памяти слова состояния
Адрес + 3	Адрес слова состояния COMMREQ (отсчитывается от нуля, в отличие от регистров, отсчитываемых от 1. Например, значение 99 дает адрес слова состояния 100)	Смещение слова состояния
Адрес + 4	0 (Используется только в режиме Wait/No Wait)	Значение тайм-аута простоя
Адрес + 5	0 (Используется только в режиме Wait/No Wait)	Максимальное время связи
Адрес + 6	FFF0H	Управляющее слово (настройка последовательного порта)
Адрес + 7	0001	Протокол: 1=SNP
Адрес + 8	0000=слэйв	Режим порта
Адрес + 9	7=38400, 6=19200, 5=9600, 4=4800	Скорость обмена
Адрес + 10	0 = нет, 1 = нечетный, 2 = четный	Четность
Адрес + 11	1 = нет	Контроль связи
Адрес + 12	0 = нет, 1 = 10мс, 2 = 100мс, 3 = 500мс	Задержка переключения
Адрес + 13	0 = длинный, 1 = средний, 2 = короткий, 3 = нет	Тайм-аут
Адрес + 14	1 = 8 бит	Количество бит на символ
Адрес + 15	0 = 1 стоповый бит, 1 = 2 стоповых бита	Стоповые биты
Адрес + 16	не используется	Интерфейс
Адрес + 17	не используется	Дуплексный режим
Адрес + 18	устанавливается пользователем*	Байты 1 и 2 идентификации устройства
Адрес + 19	устанавливается пользователем*	Байты 3 и 4 идентификации устройства
Адрес + 20	устанавливается пользователем*	Байты 5 и 6 идентификации устройства
Адрес + 21	устанавливается пользователем*	Байты 7 и 8 идентификации устройства

* Идентификатор устройства портов SNP слэйв упакован в слова так, что левый символ находится в правом байте слова. Например, если первые два символа “A” и “B,” Адрес + 18 будет содержать шестнадцатиричное значение 4241.

Пример блока данных COMMREQ конфигурирования протокола RTU

	Значения	Содержание
Первые 6 слов		Зарезервированы для использования COMMREQ.
Адрес + 6	FFF0H	Команда
Адрес + 7	0003	Протокол: 0003=RTU
Адрес + 8	0000	Режим порта 0000=слэйв
Адрес + 9	2=1200, 3-2400, 4=4800, 5=9600, 6=19200,7=38400*, 8=57600** *только для моделей ЦПУ IC200CPU005 и CPUE05	Скорость обмена
Адрес + 10	0 = нет, 1 = нечетный, 2 = четный	Четность
Адрес + 11	0 = аппаратный, 1 = нет	Контроль связи
Адрес + 12	0-255 (в десятках миллисекунд, т. е. 10=100мс)	Задержка переключения
Адрес + 13	не используется	Тайм-аут
Адрес + 14	не используется	Количество бит на символ
Адрес + 15	не используется	Стоповые биты
Адрес + 16	не используется	Интерфейс
Адрес + 17	0 = 2-проводной, 1 = 4-проводной	Дуплексный режим
Адрес + 18	Адрес станции (1-247)	Идентификатор устройства
Адрес + 19—21	не используется	Идентификатор устройства
Адрес + 22	0-255 (в десятках миллисекунд, т. е. 10=100мс)	Задержка между приемом и передачей
Адрес + 22	0-255 (в десятках миллисекунд, т. е. 10=100мс)	Задержка отсутствия сигнала RTS

Примечание

Длина блока данных (Адрес + 0) для COMMREQ, включающего задержку между приемом и передачей и задержку отсутствия сигнала RTS, должна равняться 12H, а не 10H. Поддерживаются оба варианта (длина 10H и 12H)

Если порт RTU сконфигурирован на скорость 115.2Кбод, в слово состояния COMMREQ заносятся старший код ошибки 12 (0cH) и младший код ошибки 2 (02H). Это произойдет при любой неподдерживаемой скорости обмена.

Пример блока данных COMMREQ конфигурирования протокола Serial I/O

	Значения	Содержание
Первые 6 слов		Зарезервированы для использования COMMREQ.
Адрес + 6	FFF0H	Команда
Адрес + 7	0005	Протокол: 0005=Serial IO
Адрес + 8	0=слэйв	Режим порта
Адрес + 9	4=, 4800, 5=9600, 6=19200, 7=38400*, 8=57600** *только для моделей ЦПУ IC200CPU005 и CPUE05	Скорость обмена
Адрес + 10	0 = нет, 1 = нечетный, 2 = четный	Четность
Адрес + 11	0 = аппаратный, 1 = нет	Контроль связи
Адрес + 12	0-255 (в десятках миллисекунд, т. е. 10=100мс)	Задержка переключения
Адрес + 13	0 = длинный	Тайм-аут
Адрес + 14	0=7 бит, 1=8 бит	Количество бит на символ
Адрес + 15	0 = 1 стоповый бит, 1 = 2 стоповых бита	Стоповые биты
Адрес + 16	не используется	Интерфейс
Адрес + 17	0 = 2-проводной, 1 = 4-проводной	Дуплексный режим
Адрес + 18—21	не используется	Идентификатор устройства
Адрес + 22	0-255 (в десятках миллисекунд, т. е. 10=100мс)	Задержка между приемом и передачей
Адрес + 22*	0-255 (в десятках миллисекунд, т. е. 10=100мс)	Задержка отсутствия сигнала RTS

Примечание

Длина блока данных (Адрес + 0) для COMMREQ, включающего задержку между приемом и передачей и задержку отсутствия сигнала RTS, должна равняться 12H, а не 10H. Поддерживаются оба варианта (длина 10H и 12H)

Если порт Serial I/O сконфигурирован на скорость 115.2Кбод, в слово состояния COMMREQ заносятся старший код ошибки 12 (0cH) и младший код ошибки 2 (02H). Это произойдет при любой неподдерживаемой скорости обмена.

Обращение к COMMREQ Serial I/O в цикле ПЛК

Работа по последовательному протоколу, использующему COMMREQ Serial I/O, может быть ограничена временем цикла ПЛК. Например, если протокол требует, чтобы ответ на определенную посылку от удаленного устройства был инициирован в течение 5 мС с момента получения посылки, такой способ может не сработать, если цикл ПЛК составляет 5 мс или более, т. к. своевременный отклик не гарантируется.

Поскольку Serial I/O полностью управляется прикладной программой, порт, сконфигурированный как Serial I/O, в режиме STOP автоматически переходит в режим SNP слэйв для обеспечения связи с программатором. Таким образом, протокол Serial I/O в режиме Stop не работает; он работает только, когда ПЛК находится в режиме Run.

Когда порт переключается в режим SNP слэйв, используются те же параметры связи (скорость обмена, паритет, стоповые биты …), что и при работе по протоколу Serial I/O. Следовательно, чтобы быть опознанным, программатор должен использовать те же параметры. Если хоть какие-нибудь значения параметров протокола Serial I/O не поддерживаются протоколом SNP слэйв, программатор не сможет связаться с ПЛК через этот порт.

Совместимость

Функциональные блоки COMMREQ, поддерживаемые протоколом Serial I/O, не поддерживаются другими существующими в настоящее время протоколами (такими, как SNP слэйв, SNP мастер и RTU слэйв). При попытке использовать их для порта, сконфигурированного под один из этих протоколов, выдается сообщение об ошибке.

Слово состояния COMMREQ Serial I/O

После успешного выполнения COMMREQ в слово состояния COMMREQ записывается значение 1. Любое другое значение является кодом ошибки, где младший байт - старший код ошибки, а старший байт - младший код ошибки.

Старший код ошибки	Описание	
1 (01h)	Успешное выполнение (это ожидаемое значение слова состояния COMMREQ).	
12 (0Ch)	Локальная ошибка — Ошибка выполнения местной команды. Младший код ошибки указывает конкретную ошибку.	
	1 (01h)	Команда типа Wait не разрешена. Используйте команду типа No-Wait.
	2 (02h)	Команда COMMREQ не поддерживается.
	5 (05h)	Ошибка записи слова состояния COMMREQ в память ПЛК.
	6 (06h)	Указан неверный тип памяти ПЛК.
	7 (07h)	Указано неверное смещение в памяти ПЛК.
	8 (08h)	Невозможен доступ к памяти ПЛК.
	9 (09h)	Превышена длина данных.
	12 (0Ch)	Длина блока данных COMMREQ слишком мала.
	14 (0Eh)	Неверные данные COMMREQ.
	15 (0Fh)	Невозможно выделить ресурсы системы для завершения выполнения COMMREQ.
13 (0Dh)	Удаленная ошибка — Ошибка выполнения удаленной команды. Младший код ошибки указывает ошибку.	
	2 (02h)	Количество байт, запрошенных для чтения, больше размера входного буфера или количество байт, запрошенных для записи, равно 0 или превышает 250 байт.
	3 (03h)	Длина блока данных COMMREQ слишком мала. Последовательность данных отсутствует или не завершена.
	4 (04h)	Тайм-аут ожидания данных при приеме.
	8 (08h)	Невозможен доступ к памяти ПЛК.
	12 (0Ch)	Длина блока данных COMMREQ слишком мала.
	48 (30h)	Тайм-аут передачи. Последовательный порт не смог передать последовательность данных. (Может быть вызвана отсутствием сигнала CTS, если последовательный порт сконфигурирован для использования аппаратного контроля связи.)
	50 (32h)	Тайм-аут COMMREQ. COMMREQ не завершен в течение 20 секунд.

14 (0Eh)	Ошибка автонабора — ошибка возникает при попытке пересылки последовательности команд на подключенный внешний модем. Младший код ошибки указывает конкретную ошибку.	
	1 (01h)	Не используется
	2 (02h)	Длина последовательности команд модема выходит за границу памяти указанного типа.
	3 (03h)	Длина блока данных COMMREQ слишком мала. Последовательность выходных управляющих команд отсутствует или не завершена.
	4 (04h)	Тайм-аут передачи. Последовательный порт не смог передать модему команду автонабора.
	5 (05h)	Не был получен ответ от модема. Проверьте модем и кабель.
	6 (06h)	Модем ответил сигналом BUSY (линия занята). Модем не может выполнить требуемое соединение. Удаленный модем уже используется; попробуйте передать запрос связи позже.
	7 (07h)	Модем ответил сигналом NO CARRIER (нет несущей). Модем не может выполнить требуемое соединение. Проверьте местный и удаленный модемы и телефонную линию.
	8 (08h)	Модем ответил сигналом NO DIALTONE (нет гудка в линии). Модем не может выполнить требуемое соединение. Проверьте подключения модема и телефонную линию.
	9 (09h)	Модем ответил сигналом ERROR (ошибка). Модем не может выполнить требуемое соединение. Проверьте командную строку модема и модем.
	10 (0Ah)	Модем ответил сигналом RING (входящий звонок), сообщая, что модем вызван другим модемом. Модем не может выполнить команду. Попробуйте передать команду модему позже.
	11 (0Bh)	От модема получен неизвестный ответ. Модем не может выполнить запрос. Проверьте командную строку модема и модем. Ответ должен быть CONNECT или OK.
	50 (32h)	Тайм-аут COMMREQ. COMMREQ не завершен в течение 20 секунд.

Команды COMMREQ для протокола Serial I/O

Следующие COMMREQ используются при реализации Serial I/O:

- Локальные COMMREQ - не принимают и не передают данные через последовательный порт.
    - Инициализация порта4300
    - Настройка входного буфера (4301)
    - Очистка входного Буфера (4302)
    - Чтение состояния Порта (4303)
    - Управление Портом (4304)
    - Отмена действия (4399)
- Удаленные COMMREQ - принимают и/или передают данные через последовательный порт.
    - Автонабор (4400)
    - Запись Байт (4401)
    - Чтение Байт (4402)
    - Чтение потока данных (4403)

Наложение COMMREQ

Выполнение некоторых COMMREQ Serial I/O должно быть закончено до выполнения следующего COMMREQ. Другие могут оставаться незаконченными во время выполнения следующих.

COMMREQ, которые должны завершить выполнение

- Автонабор (4400)
- Инициализация порта (4300)
- Настройка входного буфера (4301)
- Очистка входного Буфера (4302)
- Чтение состояния Порта (4303)
- Запись управления Портом (4304)
- Отмена действия (4399)
- Настройка последовательного порта (FFF0)

COMMREQ, которые могут быть незаконченными во время выполнения других

В приведенной ниже таблице показано, могут ли быть незаконченными COMMREQ Запись байт, Чтение байт и Чтение потока данных, когда выполняются другие COMMREQ.

	Новый COMMREQ										
COMMREQ, в настоящее время неоконченные	Автонабор (4400)	Запись Байт (4401)	Инициализация порта 4300	Настройка входного буфера (4301)	Очистка входного Буфера (4302)	Чтение состояния Порта (4303)	Запись управления Портом (4304)	Чтение Байт (4402)	Чтение потока данных (4403)	Отмена действия (4399)	Настройка последовательного порта (FFF0)
Запись Байт (4401)	Нет	Нет	Да	Да	Да	Да	Да	Да	Да	Да	Нет
Чтение Байт (4402)	Нет	Да	Да	Нет	Нет	Да	Да	Нет	Нет	Да	Нет
Чтение потока данных (4403)	Нет	Да	Да	Нет	Нет	Да	Да	Нет	Нет	Да	Нет

Функция инициализации порта (4300)

Эта функция отправляет в указанный порт команду сброса. Она также прерывает любой COMMREQ, выполняющийся в данный момент, и очищает внутренний входной буфер. Сигнал RTS устанавливается неактивным.

Пример командного блока функции инициализации порта

	Значение (Десятичное)	Значение (шестнадца-тиричное)	Содержание
Адрес	0001	0001	Длина блока данных
Адрес +1	0000	0000	Режим NOWAIT
Адрес +2	0008	0008	Тип памяти слова состояния (%R)
Адрес +3	0000	0000	Адрес слова состояния минус 1 (%R0001)
Адрес +4	0000	0000	не используется
Адрес +5	0000	0000	не используется
Адрес +6	4300	10CC	Команда инициализации порта

Особенности работы

Примечание: COMMREQ, прерванные из-за выполнения этой команды, не обновляют соответствующие слова состояния COMMREQ.

Предостережение: Если этот COMMREQ посылается, когда COMMREQ Запись байтов (4401) передает поток данных через последовательный порт, передача останавливается. Передача потока данных останавливается в неопределенной позиции. Кроме этого последний символ, посланный ЦПУ и полученный устройством, также будет неопределенным.

Функция настройки входного буфера (4301)

Эта функция может использоваться для изменения размера буфера внутренней памяти, в который помещаются данные при приеме. По умолчанию устанавливается максимальный размер буфера - 2Кбайт. Данные, полученные по последовательному порту, помещаются во входной буфер. Если буфер переполняется, последующие данные, полученные по последовательному порту, отбрасываются, и в слове состояния порта устанавливается бит ошибки переполнения (Overflow Error) (см. функцию чтение состояния порта).

Извлечение данных из буфера

Данные могут быть извлечены из буфера с помощью функций Чтение потока данных или Чтение байтов. Непосредственно из прикладной программы эти данные недоступны.

Если данные своевременно не извлечены из буфера, некоторые символы могут быть потеряны.

Пример командного блока функции настройки входного буфера

	Значение (десятичное)	Значение (шестнадца-тиричное)	Содержание
Адрес	0002	0002	Длина блока данных
Адрес +1	0000	0000	Режим NOWAIT
Адрес +2	0008	0008	Тип памяти слова состояния (%R)
Адрес +3	0000	0000	Адрес слова состояния минус 1 (%R0001)
Адрес +4	0000	0000	не используется
Адрес +5	0000	0000	не используется
Адрес +6	4301	10CD	Команда настройки входного буфера
Адрес +7	0064	0040	Длина буфера (в словах)

Особенности работы

Нельзя установить длину буфера равной 0. Если введена длина буфера, равная 0, по умолчанию будет установлен размер буфера 2 Кбайт.

Если указана длина буфера больше 2 Кбайт, выдается сообщение об ошибке.

Функция очистки входного Буфера (4302)

Эта операция очищает входной буфер от любых символов, полученных через последовательный порт, но не извлеченных командой чтения. Все эти символы будут утеряны.

Пример командного блока функции очистки входного буфера

	Значение (Десятичное)	Значение (шестнадца-тиричное)	Содержание
Адрес	0001	0001	Длина блока данных
Адрес +1	0000	0000	Режим NOWAIT
Адрес +2	0008	0008	Тип памяти слова состояния (%R)
Адрес +3	0000	0000	Адрес слова состояния минус 1 (%R0001)
Адрес +4	0000	0000	не используется
Адрес +5	0000	0000	не используется
Адрес +6	4302	10CE	Команда очистки входного буфера

Функция чтения Состояния Порта (4303)

Эта функция возвращает текущее состояние порта. Могут быть определены следующие события:

1. Запрос чтения был предварительно инициирован, и требуемое количество символов было получено или истекло указанное время тайм-аута.
2. Запрос записи был предварительно инициирован, и передача указанного количества символов завершилась или истекло указанное время тайм-аута.

Состояние, возвращенное функцией, указывает завершенные события или событие. Если предварительно были инициированы и запись, и чтение, могут возникнуть более одного события одновременно.

Пример командного блока функции чтения состояния порта

	Значение (Десятичное)	**Значение (шестнадца-тиричное)**	**Содержание**
Адрес	0003	0003	Длина блока данных
Адрес +1	0000	0000	Режим NOWAIT
Адрес +2	0008	0008	Тип памяти слова состояния (%R)
Адрес +3	0000	0000	Адрес слова состояния минус 1 (%R0001)
Адрес +4	0000	0000	не используется
Адрес +5	0000	0000	не используется
Адрес +6	4303	10CF	Команда чтения состояния порта
Адрес +7	0076	004C	Тип памяти состояния порта (%M)
Адрес +8	0101	0065	Адрес состояния порта в памяти (%M101)

Состояние порта

Состояние порта состоит из слова состояния и количества символов во входном буфере, которые не были извлечены приложением (полученные и доступные символы).

слово 1	Слово состояния порта (см. ниже)
слово 2	Доступные символы входного буфера

Слово состояния порта может быть:

Бит	Название	Описание	Содержание	
15	RI	Чтение выполняется	Установлен	Выполняется Чтение байтов или Чтение потока данных
			Сброшен	Произошел тайм-аут предыдущего Чтения байтов или потока данных, оно было прервано или закончено
14	RS	Успешное чтение	Установлен	Чтение байтов или Чтение потока данных успешно завершено
			Сброшен	Выполняется новое Чтение байтов или Чтение потока данных
13	RT	Тайм-аут чтения	Установлен	Произошел тайм-аут приема при Чтении байтов или Чтении потока данных
			Сброшен	Выполняется новое Чтение байтов или Чтение потока данных
12	WI	Запись выполняется	Установлен	Выполняется новая Запись байтов
			Сброшен	Произошел тайм-аут предыдущей Записи байтов, она была прервана или закончена
11	WS	Успешная запись	Установлен	Предыдущая Запись байтов успешно завершена
			Сброшен	Выполняется новая Запись байтов
10	WT	Тайм-аут записи	Установлен	Произошел тайм-аут передачи при Записи байтов
			Сброшен	Выполняется новая Запись байтов
9	CA	Символ доступен	Установлен	Непрочитанные символы в буфере
			Сброшен	Непрочитанных символов в буфере нет
8	OF	Ошибка переполнения	Установлен	Произошла ошибка переполнения последовательного порта или внутреннего буфера
			Сброшен	Выполняется Чтение состояния порта

7	FE	Ошибка кадровой синхронизации	Установлен	Произошла ошибка кадровой синхронизации последовательного порта
			Сброшен	Выполняется Чтение состояния порта
6	PE	Ошибка паритета	Установлен	Произошла ошибка четности последовательного порта
			Сброшен	Выполняется Чтение состояния порта
5	CT	Сигнал CTS активен	Установлен	Линия CTS последовательного порта активна или у последовательного порта нет линии CTS
			Сброшен	Линия CTS последовательного порта не активна
4 - 0	U	Не используются, должны быть в состоянии 0		

Функция Управления Портом (4304)

Эта функция принудительно устанавливает сигнал RTS для указанного порта:

Пример командного блока функции записи управления портом

	Значение (Десятичное)	Значение (шестнадца-теричное)	Содержание
Адрес	0002	0002	Длина блока данных
Адрес +1	0000	0000	Режим NOWAIT
Адрес +2	0008	0008	Тип памяти слова состояния (%R)
Адрес +3	0000	0000	Адрес слова состояния минус 1 (%R0001)
Адрес +4	0000	0000	не используется
Адрес +5	0000	0000	не используется
Адрес +6	4304	10D0	Команда управления портом
Адрес +7	xxxx	xxxx	Управляющее слово порта

Управляющее слово порта

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RTS	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U

Управляющее слово порта может быть:

15 **RTS** Управляемое состояние выхода **RTS**
1 = Активирует RTS
0 = Деактивирует RTS

0-14 **U** **Н**е используются (должны быть в состоянии 0)

Особенности работы

Для порта 2 (RS-485) ЦПУ, сигнал RTS управляется также драйвером передачи. Таким образом, управление RTS зависит от текущего состояния драйвера передачи. Если драйвер передачи не доступен, установка RTS с помощью COMMREQ управления портом не вызовет установки RTS в последовательной линии. Состояние драйвера передачи управляется протоколом и зависит от текущего дуплексного режима (Duplex Mode) порта. Для 2-проводного и 4-проводного дуплексных режимов, драйвер передачи доступен только во время передачи. Следовательно, RTS в последовательной линии будет активным на порту 2 (сконфигурированном для 2-проводного или 4-проводного дуплексного режима) только при передаче данных. Для дуплексного режима точка-точка драйвер передачи доступен всегда. Следовательно, в дуплексном режиме точка-точка RTS в последовательной

линии будет всегда отражать то, что выбрано с помощью COMMREQ управления портом.

Функция отмены запроса связи (4399)

Эта функция отменяет текущее выполняемое действие. Она используется для отмены и чтения, и записи.

Если выполняется операция чтения, и во входном буфере находятся необработанные символы, эти символы останутся во входном буфере и будут доступны для чтения в будущем. Последовательный порт не сбрасывается.

Пример командного блока функции отмены операции

	Значение (Десятичное)	Значение (шестнадца-тиричное)	Содержание
Адрес		0002	Длина блока данных (2)
Адрес +1	0000	0000	Режим NOWAIT
Адрес +2	0008	0008	Тип памяти слова состояния (%R)
Адрес +3	0000	0000	Адрес слова состояния минус 1 (%R0001)
Адрес +4	0000	0000	не используется
Адрес +5	0000	0000	не используется
Адрес +6	4399	112F	Команда отмены операции
Адрес +7	0001	0001	Тип отменяемой транзакции 1 Все операции 2 Операции чтения 3 Операции записи

Особенности работы

Эта функция не обновляет слова состояния отменяемых функций COMMREQ.

Предостережение: Если этот COMMREQ передается в режиме Отменить Все (Cancel All) или Отменить Запись (Cancel Write), когда COMMREQ Write Bytes (4401) передает поток данных через последовательный порт, передача прерывается. Передача потока данных останавливается в неопределенной позиции. Кроме этого последний символ, посланный ЦПУ и полученный устройством, также будет неопределенным.

Функция автонабора (Autodial) (4400)

Эта возможность позволяет ЦПУ VersaMax автоматически набирать номер с использованием модема и посылать указанный поток байтов.

Для реализации этой функции порт должен быть сконфигурирован как Serial I/O.

Например, вызов пейджера может быть выполнен тремя командами, для которых требуются три командных блока COMMREQ:

Автонабор: 04400 (1130h) Связывается с модемом.

Запись байт: 04401 (1131h) Указывает поток ASCII-символов длиной от 1 до 250 байт для передачи через последовательный порт.

Автонабор: 04400 (1130h) Разрыв телефонного соединения возлагается на прикладную программу ПЛК. Это выполняется повторной передачей команды автонабора и передачей команды разрыва соединения.

Командный блок автонабора

Команда автонабора автоматически передает Escape-последовательность, соответствующую соглашению Hayes. Если вы используете модем, не поддерживающий соглашение Hayes, вы можете использовать команду Запись Байтов (Write Bytes) для активизации модема.

Примеры обычно используемых команд для Hayes-совместимых модемов приведены ниже:

Команда	Длина	Функция
ATDP15035559999<CR>	16 (10h)	Импульсный набор номера 1-503-555-9999
ATDT15035559999<CR>	16 (10h)	Тоновый набор номера 1-503-555-9999
ATDT9,15035559999<CR>	18 (12h)	Тоновый набор с паузой для выхода на внешнюю линию
ATH0<CR>	5 (05h)	Повесить трубку
ATZ <CR>	4 (04h)	Восстановление конфигурации модема по внутренним сохраненным значениям

Пример командного блока автонабора

В этом примере командного блока COMMREQ набирается номер 234-5678 при использовании Hayes-совместимого модема.

Слово	Описание	Значения
1	0009h	Длина пользовательского блока данных (включая команду)
2	0000h	Режим NOWAIT
3	0008h	Тип памяти слова состояния (%R)
4	0000h	Адрес слова состояния минус 1 (Регистр 1)
5	0000h	не используется
6	0000h	не используется
7	04400 (1130h)	Номер команды автонабора
8	00030 (001Eh)	Тайм-аут отклика модема (30 секунд)
9	0012 (000Ch)	Количество байт в команде
10	5441h	A (41h), T (54h)
11	5444h	D (44h), T (54h)
12	3332h	Номер телефона: 2 (32h), 3 (33h)
13	3534h	4 (34h), 5 (35h)
14	3736h	6 (36h), 7 (37h)
15	0D38h	8 (38h) <CR> (0Dh)

Функция Записи Байт (Write Bytes) (4401)

Эту операцию можно использовать для передачи одного или нескольких символов удаленному устройству через указанный последовательный порт. Передаваемые символы должны находиться в памяти слов. Они не должны изменяться до окончания операции.

До 250 символов может быть передано за один вызов этой функции. Операция не получит статуса выполненной до тех пор, пока все символы не будут переданы или не произойдет тайм-аут (например, если используется аппаратный контроль связи и удаленное устройство никогда не позволяет передачу).

Пример командного блока для функции Записи Байт (Write Bytes)

	Значение (Десятичное)	Значение (шестнадца-тиричное)	Содержание
Адрес	0006	0006	Длина блока данных (включая передаваемые символы)
Адрес +1	0000	0000	Режим NOWAIT
Адрес +2	0008	0008	Тип памяти слова состояния (%R)
Адрес +3	0000	0000	Адрес слова состояния минус 1 (%R0001)
Адрес +4	0000	0000	не используется
Адрес +5	0000	0000	не используется
Адрес +6	4401	1131	Команда записи байтов
Адрес +7	0030	001E	Тайм-аут передачи (30 секунд). См. примечание ниже.
Адрес +8	0005	0005	Количество записываемых байтов
Адрес +9	25960	6568	‘h’ (68h), ‘e’ (65h)
Адрес +10	27756	6C6C	‘l’ (6Ch), ‘l’ (6Ch)
Адрес +11	0111	006F	‘o’ (6Fh)

Хотя в этом примере используются печатаемые значения символов ASCII, ограничения на коды символов, которые могут быть переданы, отсутствуют.

Особенности работы

Примечание: Ноль, указанный как тайм-аут передачи, устанавливает значение тайм-аута равным действительному времени, необходимому для передачи данных, плюс 4 секунды.

Предостережение: Если во время передачи этим COMMREQ потока данных через последовательный порт посылаются COMMREQ Инициализации Порта (4300) или COMMREQ Отмены Операции (4399) в режиме Отменить Все или

Отменить Запись, передача останавливается. Передача потока данных останавливается в неопределенной позиции. Кроме этого последний символ, посланный ЦПУ и полученный устройством, также будет неопределенным.

Функция Чтения Байт (Read Bytes) (4402)

Эта функция считывает один или несколько символов из указанного порта. Символы считываются из внутреннего входного буфера и помещаются в указанную область входных данных.

Функция возвращает и количество принятых символов, и количество необработанных символов, все еще находящихся во входном буфере. Если запрашивается нулевое количество входных символов, возвращается только количество необработанных символов во входном буфере.

Если символов недостаточно для выполнения запроса, и указано ненулевое количество считываемых символов, операция не будет завершена, пока не будет получено требуемое количество символов или не произойдет тайм-аут. По любому из этих условий состояние порта указывает причину завершения операции чтения. Слово состояния не обновляется до тех пор, пока операция чтения не закончится (по тайм-ауту или по получении всех данных).

Если установленный тайм-аут равен 0, COMMREQ останется незаконченным до тех пор, пока не будут получены все запрошенные данные, или он не будет отменен.

Если при выполнении этого COMMREQ происходит ошибка по любой причине, данные в буфер не возвращаются. Любые данные, уже находящиеся в буфере, сохраняются, и могут быть извлечены последующим запросом чтения.

Пример командного блока для функции Чтения Байт

	Значение (Десятичное)	**Значение (шестнадца-тиричное)**	**Содержание**
Адрес	0005	0005	Длина блока данных
Адрес +1	0000	0000	Режим NOWAIT
Адрес +2	0008	0008	Тип памяти слова состояния (%R)
Адрес +3	0000	0000	Адрес слова состояния минус 1 (%R0001)
Адрес +4	0000	0000	не используется
Адрес +5	0000	0000	не используется
Адрес +6	4402	1132	Команда чтения байтов
Адрес +7	0030	001E	Тайм-аут чтения (30 секунд).
Адрес +8	0005	0005	Количество считываемых байтов
Адрес +9	0008	0008	Тип памяти входных данных (%R).
Адрес +10	0100	0064	Адрес входных данных в памяти (%R0100)

Формат возвращаемых данных для функции Чтения Байт

Возвращаемые данные состоят из количества фактически прочитанных символов, количества символов, все еще доступных во входном буфере после завершения чтения (если есть), и фактических входных символов:

Адрес	Количество фактически прочитанных символов
Адрес + 1	Количество символов все еще доступных во входном буфере, если есть
Адрес + 2	Первые два символа (первый символ в младшем байте)
Адрес + 3	Третий и четвертый символы (третий символ в младшем байте)
Адрес + n	Последующие символы

Особенности работы

Если параметру типа памяти входных данных установлено значение памяти слов, и если фактически принято нечетное количество байт, тогда старший байт последнего слова, в которое записываются полученные данные, устанавливается в 0.

Поскольку данные получены по последовательному порту, они помещаются во внутренний входной буфер. Если буфер переполняется, тогда любые последующие данные, полученные по последовательному порту, отбрасываются, и в слове состояния порта устанавливается бит Ошибка Переполнения (См. Функция чтения состояния порта).

Функция Чтения Потока Данных (Read String) (4403)

Эта функция считывает символы из указанного порта до тех пор, пока не будет получен указанный завершающий символ. Символы считываются из внутреннего входного буфера и помещаются в указанную область входных данных.

Функция возвращает и количество принятых символов, и количество необработанных символов, все еще находящихся во входном буфере. Если запрашивается нулевое количество входных символов, возвращается только количество необработанных символов во входном буфере.

Если завершающего символа нет во входном буфере, операция не будет завершена до тех пор, пока не будет получен завершающий символ или не произойдет тайм-аут. По любому из этих условий состояние порта указывает причину завершения операции чтения.

Если установленный тайм-аут равен 0, COMMREQ останется незаконченным до тех пор, пока не будет получен весь запрошенный поток данных, завершенный указанным последним символом.

Если при выполнении этого COMMREQ происходит ошибка по любой причине, данные в буфер не возвращаются. Любые данные, уже находящиеся в буфере, сохраняются, и могут быть извлечены последующим запросом чтения.

Пример командного блока для функции Чтения Потока Данных

	Значение (Десятичное)	**Значение (шестнадца-тиричное)**	**Содержание**
Адрес	0005	0005	Длина блока данных
Адрес +1	0000	0000	Режим NOWAIT
Адрес +2	0008	0008	Тип памяти слова состояния (%R)
Адрес +3	0000	0000	Адрес слова состояния минус 1 (%R0001)
Адрес +4	0000	0000	не используется
Адрес +5	0000	0000	не используется
Адрес +6	4403	1133	Команда чтения потока данных
Адрес +7	0030	001E	Тайм-аут чтения (30 секунд).
Адрес +8	0013	000D	Завершающий символ (возврат каретки): должен иметь значение от 0 до 255 (0xFF), включительно
Адрес +9	0008	0008	Тип памяти входных данных (%R).
Адрес +10	0100	0064	Адрес входных данных в памяти (%R0100)

Формат возвращаемых данных для функции Чтения Потока Данных

Возвращаемые данные состоят из количества фактически прочитанных символов, количества символов, все еще доступных во входном буфере после завершения чтения (если есть), и фактических входных символов:

Адрес	Количество действительно прочитанных символов
Адрес + 1	Количество символов все еще доступных во входном буфере, если есть
Адрес + 2	Первые два символа (первый символ в младшем байте)
Адрес + 3	Третий и четвертый символы (третий символ в младшем байте)
Адрес + n	Последующие символы

Особенности работы

Если параметру типа памяти входных данных установлено значение памяти слов, и если нечетное количество байт фактически принято, тогда старший байт последнего слова, в которое записываются полученные данные, устанавливается в 0.

Поскольку данные получены по последовательному порту, они помещаются во внутренний входной буфер. Если буфер переполняется, тогда любые последующие данные, полученные через последовательный порт, отбрасываются, и в слове состояния порта устанавливается бит Ошибка Переполнения (См. Функция чтения состояния порта).

Глава
13

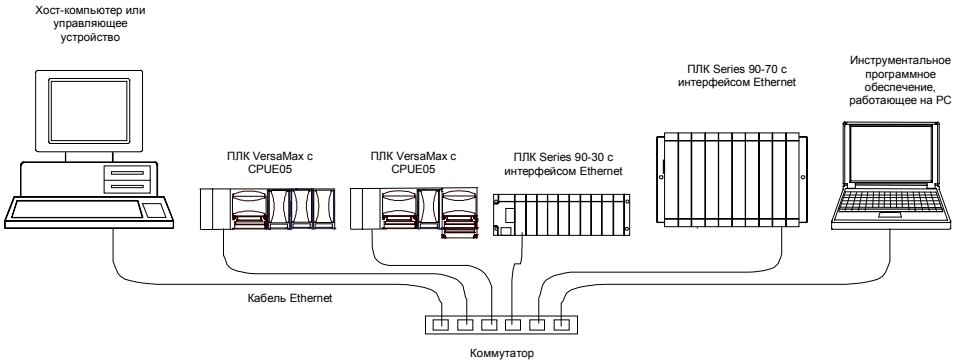
Связь по интерфейсу Ethernet

Эта глава описывает возможности связи по интерфейсу Ethernet модели ЦПУ VersaMax® IC200CPUE05.

- Обзор интерфейса Ethernet
- IP адресация
- Маршрутизаторы
- Ethernet Global Data
- Контроль состояния обмена Ethernet Global Data
- Средства диагностики
- Устранение основных неисправностей

Краткий обзор интерфейса Ethernet

ЦПУ VersaMax модели IC200CPUE05 имеет встроенный интерфейс Ethernet, позволяющий ей связываться по сети 10BaseT в полудуплексном или дуплексном режиме. Использование коммутаторов 10/100 позволяет CPUE05 работать в сети, содержащей устройства 100Мб.



Интерфейс Ethernet используется для:

- ***Передачи и приема Ethernet Global Data.*** Ethernet Global Data могут быть использованы для высокоэффективной передачи данных по сети.
- ***Доступ к данным CPUE05 с использованием хост-компьютера.*** Данные CPUE05 доступны для компьютерных приложений благодаря возможностям его SRTP сервера.
- ***Одновременная связь с несколькими устройствами.*** Возможности коммутации интерфейса Ethernet, наряду с большой производительностью сети Ethernet, позволяют CPUE05 связываться с несколькими устройствами одновременно.
- ***Подключение к другим сетям через IP маршрутизаторы третьих фирм.*** CPUE05 может связываться с другими ПЛК и другими узлами через IP маршрутизатор.
- ***Связь с удаленными компьютерами по последовательному линейному протоколу (SLIP) с использованием модемов и/или последовательных каналов.*** С помощью SLIP-устройств третьих фирм, удаленный компьютер может быть подключен к сети TCP/IP. После подсоединения, последовательное подключение может быть перенаправлено через интерфейс Ethernet CPUE05.
- ***Совместимость с другими устройствами.*** CPUE05 совместимо с интерфейсом Ethernet GE Fanuc Series 90-30, встроенным интерфейсом

Ethernet Series 90-30 CPU364, и интерфейсом Ethernet Series 90-70 (Тип 2). CPUE05 также совместимо с инструментальными программными средствами GE Fanuc, поддерживающими связь по Ethernet TCP/IP.

Ethernet Global Data

CPUE05 также поддерживает одновременно до 32 обменов Ethernet Global Data. Обмены Ethernet Global Data конфигурируются с помощью инструментального программного обеспечения для программирования ПЛК, затем сохраняются в ПЛК. Могут быть сконфигурированы как входящие, так и исходящие обмены. CPUE05 поддерживает до 1200 диапазонов данных для всех обменов Ethernet Global Data, и может быть сконфигурировано для выборочного приема обменов Ethernet Global Data.

Сервер SRTP

CPUE05 поддерживает до 8 одновременных подключений к серверу SRTP других устройств сети Ethernet, таких как программатор ПЛК, CIMPLICITY HMI, каналы SRTP для ПЛК Series 90 и приложений средств связи с хост-компьютером. Для работы сервера не требуется программирование ПЛК.

Каналы SRTP

Каналы SRTP могут использоваться для связи ПЛК Series 90-30 или Series 90-70 с CPUE05. CPUE05 не может инициировать каналы SRTP.

Подключение к сети Ethernet

Порт Ethernet использует кабель типа витая пара длиной до 100 метров между каждым узлом и коммутатором или репитером. Обычно коммутаторы или репитеры поддерживают от 4 до 12 узлов, подключенных по топологии “звезда”.

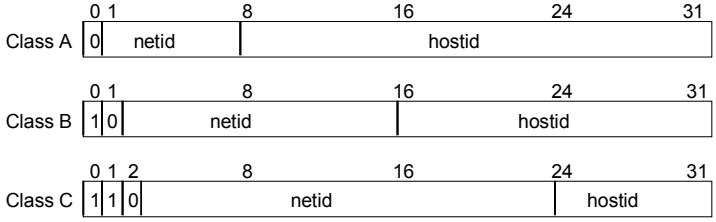
Программное обеспечение монитора станции

CPUE05 обеспечивает встроенную поддержку монитора станции. Он включает диагностику в режиме online и доступ супервизора или через порт монитора станции, или через Ethernet. Сервис монитора станции включает в себя:

- Интерактивный набор команд для опроса станции и управления ей.
- Неограниченный доступ к внутренней статистике, журналу ошибок и параметрам конфигурации.
- Защиту паролем для команд, изменяющих параметры станции или ее работу.
- Для доступа к монитору станции необходим компьютерный терминал или эмулятор терминала.

IP адресация

CPUE05 должно иметь уникальный IP адрес, идентифицирующий его в сети Ethernet. IP адрес назначается с помощью конфигурационного программного обеспечения, как описано в главе 6. Длина IP адреса - 32 бита, он включает в себя две части: netid и hostid. Формат IP адреса зависит от класса сети:



Каждый IP адрес сети имеет:

- Одинаковый класс. Существует три класса сетей: Класс A, Класс B или Класс C. Сеть Класса A может поддерживать 16777214 устройств, Класса B: 65534 устройств и Класса C: 254 устройства.
- Одинаковый netid, который обычно назначается администрацией Интернету
- Отличающийся hostid, дающий уникальный IP адрес. Параметр hostid обычно назначается администратором вашей локальной сети.

IP адрес записывается в десятичном формате как четыре десятичных целых числа (0-255), разделенных точками. Каждое целое число представляет один байт IP адреса. Например, 32-битовый IP адрес

00001010 00000000 00000000 00000001

записывается как

10.0.0.1

Класс IP адреса указывается первым десятичным целым числом:

Диапазон первого числа	Класс
0 – 127	A
128 – 191	B
192 – 223	C
224-239	D (Зарезервировано для использования в режиме Multicast)
240 – 255	E (Зарезервировано для экспериментального использования)

RFC 1918 резервирует IP адреса для частных сетей в следующих диапазонах.

10.0.0.0 – 10.255.255.255	(Класс A)
172.16.0.0 – 172.31.255.255	(Класс B)
192.168.0.0 – 192.168.255.255	(Класс C)

x.y.z.1 зарезервирован для шлюзов.

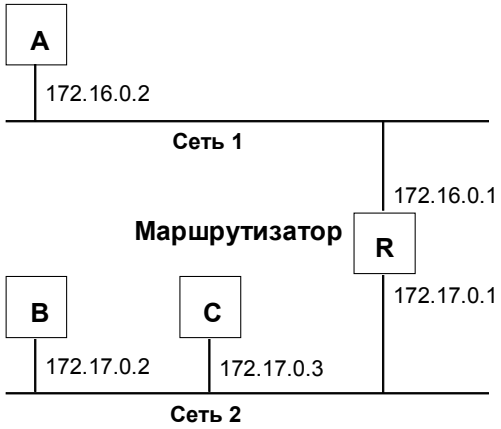
x.y.z.255 зарезервирован для режима широковещательной рассылки (broadcast) в подсети

Маршрутизаторы

Маршрутизаторы соединяют отдельные сети в систему сетей. Когда узлу в одной сети требуется связаться с узлом в другой сети, маршрутизатор передает данные между двумя сетями.

Пример: сети, соединенные маршрутизатором

На следующем рисунке показаны сети Сеть 1 и Сеть 2, соединенные маршрутизатором R.



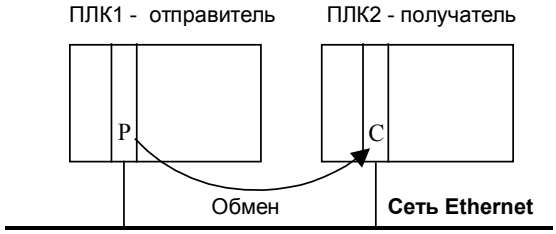
Узел B может связываться с узлом C напрямую, т. к. они работают в одной сети. Их IP адреса имеют одинаковый netid.

Однако, чтобы переслать данные узлу A, который находится в другой сети (у него другой netid, узел B должен переслать их через маршрутизатор. Маршрутизатор имеет два IP адреса (172.16.0.1 и 172.17.0.1). Первый используется узлами Сети 1, а второй узлами Сети 2. В этом примере IP адрес маршрутизатора в Сети 2 - 172.17.0.1. Этот адрес будет сконфигурирован в узле B, как его адрес шлюза (“gateway”) по умолчанию.

Ethernet Global Data

Ethernet Global Data – это данные, автоматически передающиеся от Ethernet-устройства одному или нескольким Ethernet-устройствам. Если Ethernet Global Data были сконфигурированы, данные передаются автоматически во время работы системы. Для передачи или приема Ethernet Global Data не требуется дополнительное программирование.

Устройство, посылающее Ethernet Global Data, называется *отправитель*. Каждое устройство, принимающее Ethernet Global Data называется *получатель*. Каждое отдельное сообщение Ethernet Global Data называется *обмен*.



Ethernet Global Data обеспечивает простой, регулярный обмен данными между устройствами. Не следует использовать Ethernet Global Data для сообщения о событии, если возможная потеря данных может оказаться значительной.

ЦПУ VersaMax IC200CPUE05 может быть сконфигурирован на обработку до 32 обменов Ethernet Global Data (всего входящих и исходящих). Каждый обмен Ethernet Global Data должен быть сконфигурирован отдельно для каждого ПЛК и состоять из одного или нескольких диапазонов данных. Информация о конфигурировании приведена в главе 6.

Частота посылок/приемов

Во время конфигурирования, для отправителя устанавливается период повтора каждого обмена Ethernet Global Data. Диапазон составляет от 10 миллисекунд до 1 часа с шагом 10 мс. Нет необходимости передавать и принимать данные чаще, чем требуется в приложении. Это уменьшает нагрузку на сеть и на устройства сети, освобождая ресурсы для других обменов.

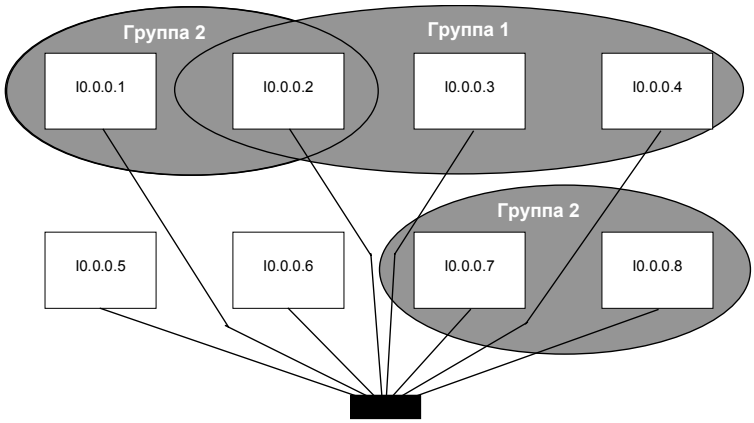
Тайм-аут периода обновления получателя

Может быть настроен тайм-аут периода обмена, являющийся частью конфигурации каждого приема. ЦПУ сообщает об ошибке, если первый или последующий пакет данных не был получен за указанное время. Тайм-аут может быть установлен в диапазоне от 10 до 3,600,000 мс, 0 означает отсутствие проверки тайм-аута. Тайм-аут получателя должен быть больше периода повтора отправителя. GE Fanuc рекомендует, чтобы тайм-аут получателя был больше периода отправления не менее, чем в два раза .

Группы Ethernet Global Data

Если более одного устройства в сети должно получить данные обмена по Ethernet Global Data, эти устройства могут быть определены как группа. Сеть может включать до 32 групп. Применение групп позволяет всем получателям в составе группы одновременно принимать данные от отправителя.

Устройства могут принадлежать более чем к одной группе, как показано ниже.



Каждое устройство в группе откликается на назначенный группе идентификационный номер ID. Для CPUE05 номера ID группы могут быть от 1 до 32.

Каждый ID группы соответствует IP адресам групповой рассылки Multicast (Class D), зарезервированным администрацией Интернета. Следующие IP адреса Multicast используются Ethernet Global Data по умолчанию:

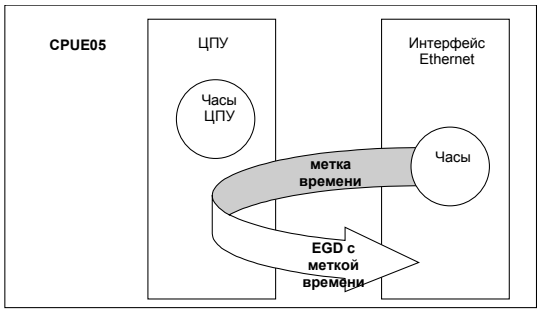
ID группы	IP адрес
1	224.0.7.1
2	224.0.7.2
.	.
.	.
.	.
32	224.0.7.32

IP адреса групп Multicast, используемые Ethernet Global Data не следует изменять, если адреса по умолчанию не вызывают конфликта в сети. В случае необходимости они могут быть изменены на резервные IP адреса Multicast (от 224.0.0.0 до 239.255.255.255). Изменение должно быть сделано с помощью файла Advanced User Parameter.

Метки времени обменов Ethernet Global Data

ЦПУ ПЛК добавляет метку времени к каждому сообщению Ethernet Global Data, передаваемому им. Метка времени указывает, когда данные были переданы от ЦПУ ПЛК к его интерфейсу Ethernet для передачи по сети.

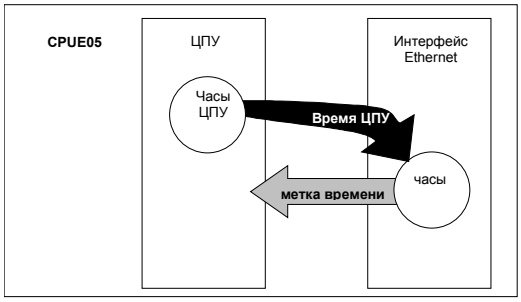
ЦПУ ПЛК получает данные метки времени от часов интерфейса Ethernet. ЦПУ использует эти метки времени только для обменов Ethernet Global Data. Метки времени интерфейса Ethernet не влияют на время внутренних часов ЦПУ.



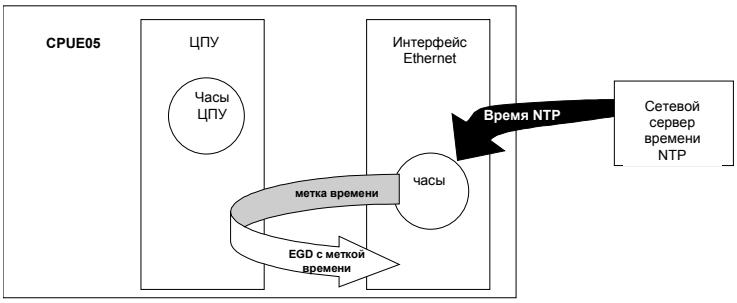
Синхронизация меток времени

Часы меток времени интерфейса Ethernet синхронизированы либо с часами ЦПУ, либо с часами внешнего сервера сетевого протокола времени (NTP - Network Time Protocol).

- **Часы ЦПУ:** Если не сконфигурирован ни один сервер NTP, встроенные часы интерфейса Ethernet синхронизируются с часами ЦПУ при включении питания или перезапуске. Т. к. часы других устройств в сети не синхронизированы с CPUE05, их метки времени не могут быть точно сравнены.



- **Часы сервера NTP:** Если серверы времени сконфигурированы и присутствуют в сети (подробности конфигурирования описаны в главе 6), встроенные часы интерфейса Ethernet периодически синхронизируются с часами серверов NTP (от одного до трех) сети. Интерфейс Ethernet периодически запрашивает время у серверов и использует время наиболее точного сервера (на основе номера уровня NTP).



У интерфейсов Ethernet, сконфигурированных для использования протокола сетевого времени (NTP), будут обновленные синхронизированные метки времени, т. к. все они контролируются часами сервера NTP. Поэтому могут быть выполнены точные сравнения времени обменов. Например, если несколько ПЛК передают данные аварийной сигнализации, такая синхронизация полезна для определения порядка появления сигналов.

Может быть использовано несколько серверов NTP для облегчения доступа к службе времени.

Если время берется с сервера NTP, интерфейс Ethernet поддерживает даты, начиная с 1 января 1970г.

Конфигурирование NTP для интерфейса Ethernet CPUE05

Чтобы реализовать протокол сетевого времени в интерфейсе Ethernet CPUE05, в конфигурации Ethernet ПЛК указываются IP адреса серверов NTP (от одного до трех) . Подробности приведены в главе 6 “Конфигурирование интерфейса Ethernet”. CPUE05 не поддерживает работу NTP в режиме групповой рассылки multicast; несколько серверов NTP могут быть определены по отдельности.

Интерфейс Ethernet CPUE05 всегда работает в режиме “клиент”. Он синхронизируется с сервером времени NTP, но не синхронизирует другие устройства в сети.

Для достижения максимальной точности, временная синхронизация использует несколько обменов. В случае установки значения времени опроса по умолчанию, NTP синхронизация должна произойти примерно через 2 минуты после установки сервера времени.

Содержание обмена Ethernet Global Data

Каждый обмен Ethernet Global Data состоит из одного или нескольких диапазонов данных, передаваемых, как последовательность от одного до 1400 байт данных. Содержание данных определяется как для отправителя, так и для получателя данных. В этом примере отправитель посылает 11-байтное сообщение, состоящее из содержимого регистров %R00100 - %R00104 и содержимого ячеек памяти с %I00257 по %I00264:

Адрес	Длина	Тип	Описание
%R00100	5	WORD	Конвейер 1 в ПЛК 1
%I00257	1	Байт	Концевой выключатель конвейера1 в ПЛК1

Такой же обмен может быть сконфигурирован для каждого получателя, в соответствии с требованиями его приложения. Величина обмена должна быть совместима со всеми узлами сети.

Типы данных для Ethernet Global Data

В приведенной ниже таблице перечислены типы памяти, которые могут быть сконфигурированы для передачи и/или приема Ethernet Global Data.

Тип	Описание	Отправитель, Получатель
%R	Регистровая память в представлении слов	О/П
%AI	Память аналоговых входов в представлении слов	О/П
%AQ	Память аналоговых выходов в представлении слов	О/П
%I	Память дискретных входов в представлении байт	О/П
%Q	Память дискретных выходов в представлении байт	О/П
%T	Дискретная временная память в представлении байт	О/П
%M	Дискретная память в представлении байт	О/П
%SA	Дискретная системная память группы A в представлении байт	О/П
%SB	Дискретная системная память группы B в представлении байт	О/П
%SC	Дискретная системная память группы C в представлении байт	О/П
%G	Таблица дискретных глобальных данных в представлении байт	О/П

Диапазоны данных при обмене Global Data

Параметры обменов определяются конфигурацией Ethernet Global Data при конфигурировании оборудования. Это могут быть:

- До 1200 диапазонов данных для всех обменов EGD одного CPUE05.
- До 100 диапазонов данных для одного обмена.
- Длина обмена от 1 байта до 1400 байт. Общий размер обмена равен сумме длин данных всех диапазонов, сконфигурированных для этого обмена.

Различные обмены могут совместно использовать часть или все диапазоны данных, даже если обмены передаются с разной скоростью. Получатель не обязан принимать все данные переданного обмена. Принимаемый обмен может быть сконфигурирован так, чтобы игнорировать указанные диапазоны данных. (См. “Выборочный прием” в главе 6.)

Влияние режимов ПЛК на Ethernet Global Data

Обычный режим ПЛК для работы с Ethernet Global Data - Run с разрешением ввода-вывода. В этом режиме Ethernet Global Data остаются сконфигурированными и выполняются обмены обоих типов - как исходящий, так и входящий. Если ПЛК находится в режиме Stop с запретом ввода-вывода, ID отправителя остается сконфигурированным, но прием и передача данных прекращаются. Когда ПЛК остановлен, Ethernet интерфейс продолжает обрабатывать данные принимаемых обменов. Последние принятые из сети данные будут доступны для приложения, когда ПЛК вернется в состояние разрешения ввода-вывода.

В следующей таблице описывается, что происходит с конфигурацией и работой Ethernet Global Data в различных режимах ПЛК.

	Продолжение обменов...	
Режим ПЛК	**Исходящих**	**Входящих**
RUN-вывод разрешен	Да	Да
STOP-ввод-вывод разрешен	Да	Да
STOP-ввод-вывод запрещен	Нет	Нет *

* Последние принятые из сети данные доступны для приложения, при переходе ПЛК из режима Stop в режим Run.

Синхронизация EGD

Ethernet Global Data стремится предоставлять наиболее свежие данные в соответствии со сконфигурированным графиком. Интерфейс Ethernet поддерживает таймер для каждого передаваемого обмена. Когда истекает заданное в таймере время, интерфейс Ethernet запрашивает в памяти данные для обмена во время формирования выходов в следующем цикле ЦПУ. После передачи данных в цикле ЦПУ, интерфейс Ethernet немедленно формирует посылку и передает ее в сеть. При получении посылки с принимаемым обменом, она передается в ЦПУ во время опроса входов в следующем цикле ЦПУ.

Такой способ построения графика обмена Ethernet Global Data вносит в периодичность отправляемых посылок непостоянство в пределах продолжительности одного цикла ЦПУ отправителя. Такая гибкость времени между посылками обеспечивает передачу наиболее свежих данных.

Обычно, нет необходимости устанавливать период передачи меньше времени цикла ЦПУ. Если значение периода передачи меньше времени цикла ЦПУ, интерфейс Ethernet будет передавать устаревшую посылку (посылку, содержащую те же данные, что и предыдущая) в сконфигурированном интервале. Когда ЦПУ выдаст свежие данные в конце цикла, интерфейс Ethernet немедленно передаст следующую посылку с обновленными данными. Таймер передаваемого обмена не сбрасывается при передаче посылки. Это может приводить к большему количеству посылок в сети за сконфигурированный период, чем ожидалось.

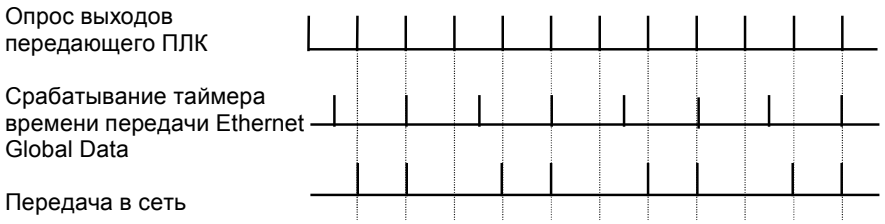
Примеры синхронизации

На следующей иллюстрации показана зависимость между временем опроса выходов ПЛК, таймером исходящего обмена и посылками данных в сети.

Пример 1

Только одна посылка передается в сеть за период передачи. Время между посылками может изменяться до времени цикла передающего ЦПУ.

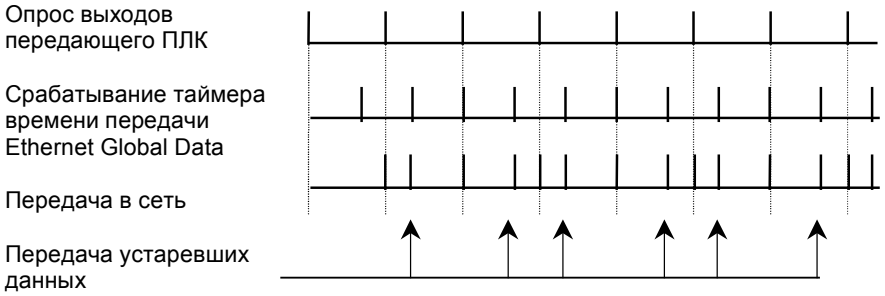
Период передачи = 1.5 времени цикла ЦПУ



Пример 2

Более одной посылки может быть передано за период передачи, и устаревшие посылки передаются в сеть.

Период передачи = 2/3 времени цикла ЦПУ



Средства диагностики

Существует несколько средств, позволяющих вам диагностировать проблемы, которые могут возникнуть при работе Ethernet и Ethernet Global Data.

- Проверьте светодиоды ***Ethernet***, как описано на следующих ниже страницах, для определения неисправности, возникшей при включении питания интерфейса Ethernet. Светодиоды обеспечивают немедленную визуальную информацию о состоянии интерфейса.
- Используйте таблицу ошибок ПЛК ***(PLC Fault Table)***, также описанную в этой главе. В таблице ошибок ПЛК записываются ощибки, зарегистрированные ПЛК, интерфейсом Ethernet, и другими модулями. Доступ к таблице ошибок ПЛК осуществляется с помощью инструментального программного обеспечения для программирования ПЛК.
- Прикладная программа может использовать специальные статусные данные для контроля работы Ethernet.
  - Адрес состояния интерфейса Ethernet, определенный при конфигурировании ПЛК. По этому адресу содержится информация о состоянии интерфейса Ethernet.
  - Слова состояния обмена (Exchange Status), определенные при конфигурировании Ethernet Global Data, содержат информацию о состоянии обмена.
- Используйте функцию монитора станции для определения проблем, возникших с интерфейсом Ethernet, с сетью, с обменом по внутренней шине ПЛК, или с вашим приложением. Команды монитора станции LOG, TALLY и STAT особенно полезны. За информацией по доступу и использованию монитора станции обратитесь к документу *Монитор Ethernet станции ПЛК VersaMax. Руководство (VersaMax PLC Ethernet Station Manager Manual)*.

Что делать, если вы не можете решить проблему

Если вы не можете решить вашу проблему, обратитесь в GE Fanuc Automation – NA, 1-800-GE FANUC. Пожалуйста, имейте во время звонка при себе следующую информацию.

- Название изделия и его номер по каталогу.
- Описание симптомов проблемы. *В зависимости от проблемы, у вас может быть запрошена следующая информация:*

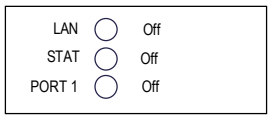
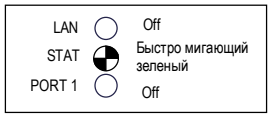
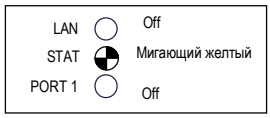
1. Прикладная программа релейной логики и время цикла ПЛК в момент возникновения проблемы.
2. Список параметров конфигурации интерфейса Ethernet, с которым возникла проблема.
3. Описание конфигурации сети. Оно должно включать количество ПЛК и компьютеров в сети, тип используемого сетевого кабеля (витая пара, оптоволокно и т. д.), длину сетевого кабеля, наименование производителя и количество используемых трансиверов, коммутаторов и сетевых переключателей.

Проверка светодиодов Ethernet

После конфигурирования интерфейса выполните следующие шаги для проверки правильности работы интерфейса Ethernet.

1. Выключите питание ПЛК на 3–5 секунд, затем снова включите питание. Начнется серия диагностических тестов. Во время выполнения диагностики при включении питания, после небольшой задержки, светодиод STAT модуля ЦПУ, находящийся со стороны интерфейса Ethernet, будет мигать. Светодиоды LAN и PORT1 выключены. Если обнаруживается фатальная ошибка, она индицируется миганием светодиода STAT желтого цвета, двузначным кодом.
2. После успешного прохождения диагностики, все три светодиода интерфейса Ethernet кратковременно включаются. Светодиоды STAT и LAN должны быть зелеными. Светодиод LAN мигает при наличии обмена по сети.
3. Если светодиод STAT светится желтым цветом, проверьте таблицу ошибок ПЛК (PLC Fault Table). В мониторе станции вы можете также использовать команду LOG, как описано в документе GFK-1876, *Монитор Ethernet станции ПЛК VersaMax. Руководство пользователя (The VersaMax PLC Ethernet Station Manager Manual)*.

Если проблема возникает при включении питания, интерфейс Ethernet может не начать работу. Проверьте светодиоды интерфейса Ethernet, как описано ниже.

Светодиоды Ethernet	Индикация	Действия
	Выключен	▪ Убедитесь, что питание ПЛК включено ▪ Проверьте записи в таблице ошибок ПЛК (PLC Fault Table) ▪ Проверьте конфигурацию ▪ Проверьте установку модуля ▪ Если проблема сохраняется, замените ЦПУ ПЛК
	Выполняется диагностика	Действия не требуются; Диагностика закончится в течение 3 - 10 секунд.
	Ошибка оборудования. STAT: мигает двузначным кодом: 1 – 2 неожиданное прерывание 1 – 3 ошибка таймера 1 – 4 ошибка ПДП 2 – 1 ошибка ОЗУ 2 – 2 ошибка стека 2 – 3 ошибка общей памяти интерфейса 2 – 4 ошибка контрольной суммы фирменного программного обеспечения 3 - 1 неопределенная инструкция или деление на 0 3 – 2 неожиданное прерывание SWI 3 – 3 ошибка прекращения упреждающего выбора 3 - 4 ошибка прерывания данных 3 – 5 неожиданный запрос IRQ 3 – 6 неожиданное прерывание FIQ 3 - 7 резерв 4 - 1 фатальная ошибка операционной системы при включении или повреждение EEPROM	▪ Заметьте код ошибки ▪ Повторно включите питание или перезапустите интерфейс Ethernet ▪ Если проблема сохраняется, замените оборудование ПЛК.

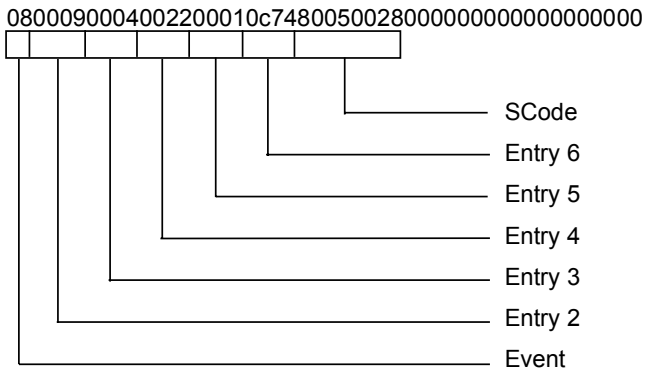
Светодиоды Ethernet	Индикация	Действия
5 EMBED Word.Picture.8 LAN Off STAT Медленно мигающий зеленый PORT 1 Off	Ожидание данных конфигурации Ethernet от ЦПУ. PORT 1: Порт 1 под управлением ЦПУ ПЛК.	▪ Используйте программатор ПЛК для обновления конфигурации, затем сохраните конфигурацию в ПЛК. ▪ Выключите и включите питание ПЛК. ▪ Сбросьте ошибки и нажмите кнопку Restart менее, чем на 5 секунд, для перезапуска интерфейса Ethernet.
LAN Зеленый/ мерцающий STAT Медленно мигающий зеленый PORT 1 Off	Ожидание IP адреса LAN: Интерфейс Ethernet в режиме online. Мигает при обмене. STAT: IP адрес не сконфигурирован. PORT 1: Порт 1 под управлением ЦПУ ПЛК.	IP адрес не сконфигурирован, или сконфигурирован как 0.0.0.0 ▪ Используйте программтор ПЛК для конфигурирования не нулевого IP адреса.
LAN Зеленый/ мерцающий STAT Медленно мигающий зеленый PORT 1 Желтый	Ожидание IP адреса LAN: Интерфейс Ethernet в режиме online. Мигает при обмене. STAT: IP адрес не сконфигурирован. PORT 1: Порт 1 доступен для использования монитором станции	
LAN Желтый STAT Медленно мигающий зеленый PORT 1 Желтый	Ожидание IP адреса LAN: Интерфейс Ethernet в режиме offline. Попытка восстановления. STAT: IP адрес не сконфигурирован. PORT 1: Порт 1 под управлением ЦПУ ПЛК.	
LAN Желтый STAT Медленно мигающий зеленый PORT 1 Желтый	Ожидание IP адреса LAN: Интерфейс Ethernet в режиме offline. Попытка восстановления. STAT: IP адрес не сконфигурирован. PORT 1: Порт 1 доступен для использования монитором станции	
LAN Зеленый/ мерцающий STAT Зеленый PORT 1 Off	Работа LAN: Интерфейс Ethernet в режиме online. Мигает при обмене. STAT: Коллизий не обнаружено PORT 1: Порт 1 под управлением ЦПУ ПЛК.	Возможные проблемы, если LAN не горит: ▪ Сетевой кабель не подключен либо к ПЛК, либо к коммутатору. ▪ Коммутатор отключен/поврежден. ▪ Сетевой кабель не согласован. Если STAT светится желтым светом - обнаружена коллизия.
LAN Зеленый/ мерцающий STAT Зеленый PORT 1 Желтый	Работа LAN: Интерфейс Ethernet в режиме online. Мигает при обмене. STAT: Коллизий не обнаружено PORT 1: Порт 1 назначен для использования монитором станции	
LAN Желтый STAT Зеленый PORT 1 Off	Работа LAN: Интерфейс Ethernet в режиме offline. Попытка восстановления. STAT: Коллизий не обнаружено PORT 1: Порт 1 под управлением ЦПУ ПЛК.	
LAN Желтый STAT Зеленый PORT 1 Желтый	Работа LAN: Интерфейс Ethernet в режиме offline. Попытка восстановления. STAT: Коллизий не обнаружено PORT 1: Порт 1 назначен для использования монитором станции	
LAN Медленно мигающий зеленый, STAT все светодиоды PORT 1 мигают одновременно	Загрузка программного обеспечения Загрузка нового фирменного программного обеспечения (через последовательный порт ЦПУ)	Действия не требуются; интерфейс Ethernet перезапустится автоматически после окончания загрузки

Использование таблицы ошибок ПЛК (PLC Fault Table)

Большинство неисправностей, относящихся к интерфейсу Ethernet, регистрируется в таблице ошибок ПЛК (PLC Fault table). В таблице далее приведены ошибки интерфейса Ethernet и действия по их исправлению.

Чтобы отобразить текст ошибки интерфейса Ethernet, войдите в таблицу ошибок ПЛК (PLC Fault Table) с помощью программатора. Для интерфейса Ethernet, крайние левые 14 разрядов дополнительных данных ошибки показывают соответствующие записи событий (Event) - 2 знака, вводы (Entry), по 4 знака каждый, и другие дополнительные данные.

В следующем примере приведено Event 8, Entry 2=9, Entry 3=4, Entry 4 = 22H, Entry 5 = 1, Entry 6=c74H, и SCode = 80050028H.



Эта информация может быть использована при обращении к детальному описанию ошибок, включенному в таблицу регистрации событий команды LOG в документе *Монитор станции ПЛК VersaMax (VersaMax PLC Station Manager Manual).*

Обратите внимание, пожалуйста, что некоторые внутренние системные ошибки выдают сообщения об ошибке, как ASCII-текст в дополнительных данных ошибки.

Описание таблицы ошибок ПЛК (PLC Fault Table)

Ошибка ПЛК	Действия пользователя
Ошибка связи по внутренней шине ПЛК; потерянный запрос	Проверьте, нормально ли работает ЦПУ ПЛК (обычно в режиме Run) * Убедитесь, что вы не посылаете запросы COMMREQ быстрее, чем интерфейс Ethernet может обработать их. *
Неверный запрос локального приложения; запрос отклонен	Проверьте правильность управляющего кода COMMREQ. *
Не верный запрос удаленного приложения; запрос отклонен	Проверьте правильность работы удаленного узла. *
Невозможно найти удаленный узел; запрос отклонен	Сообщается об ошибке, когда сообщение получено с неизвестного IP адреса. Ошибка может указывать, что удаленный узел не работает в сети. Проверьте работоспособность узла в сети и правильность его адреса.
Comm_req – запрограммирован неверный ID задачи	Сообщение ПЛК о неизвестной интерфейсу Ethernet задаче. Проверьте функциональный блок COMMREQ.
Comm_req – режим ожидания (Wait) не разрешен	Проверьте COMMREQ, что он передается в режиме без ожидания (no-wait).
Нехватка памяти сетевых данных; возобновление	У интерфейса Ethernet нет свободной памяти для обработки соединений. *
Емкость сетевого интерфейса превышена; запрос отклонен	Проверьте, что ограничение на количество подключений не превышено.
неисправность трансивера; отключен от сети до исправления	Интерфейс Ethernet не подключен должным образом к сети. Проверьте подключение к коммутатору или переключателю сети.
Ошибка сетевого системного обеспечения; возобновление прерванного соединения	Внутренняя системная ошибка. *
Ошибка сетевого системного обеспечения; перезапуск сетевого интерфейса	
Ошибка сетевого системного обеспечения; возобновление	
Сбой программного обеспечения модуля; требуется повторная загрузка	Фатальная внутренняя системная ошибка. Свяжитесь с GE Fanuc Automation – NA.
Состояние модуля не позволяет использование функции Comm_Req; игнорируется	COMMREQ был принят, когда интерфейс Ethernet не мог его обработать. Убедитесь, что интерфейс Ethernet сконфигурирован и находится в режиме online.
Неподдерживаемая возможность в конфигурации	Была сделана попытка конфигурирования возможности, не поддерживаемой интерфейсом Ethernet. Проверьте версию ЦПУ; закажите обновление для ЦПУ и/или интерфейса Ethernet.

* Если проблема сохраняется, свяжитесь с GE Fanuc Automation – NA.

Проверка состояния интерфейса Ethernet

Прикладная программа может контролировать состояние интерфейса Ethernet, используя биты состояния, описанные ниже. Начальный адрес данных - параметр *Status Address,* указываемый при конфигурировании ЦПУ. Подробности приведены в главе 6 “Конфигурирование интерфейса Ethernet”.

Интерфейс Ethernet обновляет эти статусные биты в каждом цикле ввода-вывода ПЛК. Обычно, биты состояния интерфейса Ethernet занимают один блок в памяти. Большинство из этих битов зарезервированы. Пять представляют интерес для проверки состояния интерфейса Ethernet:

Биты состояния	Короткое описание
1 – 2	Зарезервированы, всегда 0
3	Полный дуплекс
4-12	Зарезервированы, всегда 0
13	Сеть - OK
14	Проблема ресурсов
15	Зарезервированы, всегда 0
16	Интерфейс сети - OK
17-80	Зарезервированы

Бит 3: Полный дуплекс	Если этот бит установлен в 1, CPUE05 работает в полнодуплексном режиме Ethernet. Полнодуплексный или полудуплексный режим работы автоматически устанавливается между CPUE05 и подключенным к нему сетевым устройством, обычно сетевым коммутатором. Если этот бит установлен в 0, CPUE05 работает в полудуплексном режиме Ethernet. Этот бит соответствует действительности, только если бит 13 (LAN OK) установлен в 1.
Бит 13: Сеть - OK	Этот бит находится в состоянии 1, если интерфейс Ethernet может связываться по сети. Если сеть не доступна из-за локальных или сетевых проблем, этот бит устанавливается в состояние 0. Когда связь восстанавливается, он автоматически устанавливается в 1.
Бит 14: Проблема ресурсов	Этот бит в состоянии 1, если у интерфейса Ethernet есть проблема ресурсов (например, недостаток памяти данных). Бит сбрасывается в 0 в следующем цикле ПЛК. В зависимости от сложности проблемы, интерфейс Ethernet может быть или не быть в состоянии продолжать работу. Используйте таблицу ошибок ПЛК (PLC Fault Table) для идентификации проблемы. Более полная информация - через команды монитора станции STAT B и LOG.
Бит 16: Интерфейс сети - OK	Когда этот бит в состоянии 1, интерфейс Ethernet инициализирован надлежащим образом. Когда этот бит в состоянии 0, все остальные биты состояния интерфейса Ethernet не соответсвуют действительности.

Проверка состояния обмена по Ethernet Global Data

Для проверки состояния любого обмена по Ethernet Global Data необходимо контролировать значение слова состояния обмена (Exchange Status), выбираемого при конфигурировании Ethernet Global Data. ПЛК автоматически записывает информацию о состоянии обмена в эти ячейки памяти, если:

- превышен период передачи/приема (установленное значение периода).
- конфигурация Ethernet Global Data сохраняется в ПЛК.
- включается питание ПЛК, и в нем записана конфигурация Ethernet Global Data.
- происходит перезапуск интерфейса Ethernet, сконфигурированного для Ethernet Global Data.

Если прикладная программа использует слово состояния обмена (Exchange Status word) для проверки состояния обмена, она должна сбрасывать это слово в 0 после записи в него значения, отличного от нуля. Это позволит прикладной программе обнаружить новое состояние обмена в последующих циклах.

Слово состояния обмена использует следующие коды ошибок для отображения состояния обмена. См. также раздел *Основные неисправности в* этой главе ниже.

Значение (Десятичное)	Ошибка	Описание
0	Состояние обмена не изменилось	Исходящий: Начальное значение до первого периода обновления отправителя. Входящий: Данные не были обновлены и значение таймаута не было превышено.
1	Нет ошибок	Исходящий: Исходящий обмен передает данные. Входящий: Данные были обновлены в соответствии с графиком.
3	Ошибка NTP	Только принятый: ЦПУ сконфигурирован для синхронизации с сетевым временем, но не синхронизирован.
4	Ошибка спецификации	Входящий и исходящий: Ошибка конфигурирования обмена. Для CPUE05, эта ошибка не указывает на несоответствие размера принятого обмена.
6	Таймаут обновления.	Только входящий: Значение периода таймаута было превышено, но данные не были обновлены через сеть.
7	Получение данных после таймаута обновления	Только входящий: Данные были обновлены по сравнению с последним приемом, но не были обновлены в течение периода таймаута.
10	Не доступно IP соединение	Входящий и исходящий: Сетевое IP соединение не доступно.
12	Ошибка недостатка ресурсов	Входящий и исходящий: Не доступны местные ресурсы для установления обмена. Подробности см. в таблице ошибок ПЛК (PLC Fault Table).
14	Ошибка длины	Только входящий: Длина принятого пакета не соответствует ожидаемой длине.
18	Потеря интерфейса Ethernet	Входящий и исходящий: Отсутствует связь интерфейса Ethernet с ЦПУ. В таблице ошибок ПЛК (PLC Fault Table) также может быть запись о потере или сбросе модуля . Если отказ имеет кратковременный характер, состояние обмена может позже измениться. Это говорит о том, что последующие передачи обмена были успешными.
22	EGD не поддерживаются	С CPUE05 такая ошибка невозможна.
26	Нет ответа	Входящий и исходящий: Интерфейс Ethernet не может установить обмен.
28	Иная ошибка	Входящий и исходящий: Ошибка, отличная от ошибок 12, 14, 18 или 26, возникшая при установлении обмена. Подробности см. в таблице ошибок ПЛК (PLC Fault Table).
30	Обмен удален	Входящий и исходящий: Обмен был удален, и больше не будет опрашиваться.

Использование функции монитора станции Ethernet

CPUE05 может работать с локальным монитором станции через порт 1 (Port 1). Этот порт может быть сконфигурирован либо для связи ЦПУ по последовательному протоколу (SNP, RTU, Serial I/O), либо для использования локального монитора станции. Если Port 1 сконфигурирован как локальный монитор станции (Station Manager), он не может использоваться для последовательного подключения ЦПУ или для загрузки фирменного программного обеспечения. Однако, если порт сконфигурирован как порт ЦПУ (установка по умолчанию), он может быть временно переведен в режим работы локального монитора станции с помощью кнопки Restart (или с помощью команды монитора станции “chport1”).

CPUE05 также поддерживает работу удаленного монитора станции через сеть Ethernet network по протоколу UDP. Используя протокол UDP, удаленная станция адресуется через IP адрес. В отличие от ряда изделий Series 90 Ethernet, CPUE05 не может посылать или принимать сообщения удаленного монитора станции, посланные по указанному MAC адресу.

Подробная информация по функциям монитора станции приведена в документе GFK-1876, *Монитор Ethernet станции ПЛК VersaMax. Руководство пользователя (VersaMax PLC Ethernet Station Manager User’s Manual).*

Устранение основных неисправностей

Ниже описаны некоторые основные ошибки Ethernet. Ошибки Ethernet обычно отображаются в таблице ошибок ПЛК (PLC Fault Table) и в протоколе событий Ethernet. Как было описано в разделе *Использование таблицы ошибок ПЛК (PLC Fault Table)*, сообщения об ошибках ПЛК, выданные интерфейсом Ethernet, содержат описание событий в дополнительных данных ошибки. Подробное описание ошибок Ethernet приведено в документе *Монитор Ethernet станции ПЛК VersaMax. Руководство пользователя (VersaMax Station Manager Manual)*, GFK-1876.

Ошибки таймаута ПЛК

Когда у CPUE05 SRTP трафик превышает возможности ПЛК по обработке запросов, могут произойти ошибки таймаута ПЛК. ПЛК по ошибке таймаута разрывает соединение сервера SRTP; в этом случае удаленный SRTP клиент должен заново установить SRTP соединение с CPUE05.

Эта ошибка отображается в таблице ошибок ПЛК как:

"Backplane communication with PLC fault; lost request"
Event = 8, Entry 2 = 8

"Backplane communication with PLC fault; lost request"
(без значения Event)

Эти ошибки также могут сопровождаться любыми из следующих сообщений:

"Backplane communication with PLC fault; lost request"
Event = 8, Entry 2 = 6

"LAN system-software fault; resuming"
Event = 8, Entry 2 = 16

Условия таймаута ПЛК возникают, когда CPUE05 не может обработать запросы за указанный период таймаута. Для исправления ситуации следует уменьшить количество запросов, или увеличить возможности обработки ПЛК.

Причина	Действия
Большой SRTP трафик.	Уменьшить размер, количество или частоту запросов SRTP удаленного SRTP клиента.
Большое время цикла ПЛК.	Изменить прикладную программу ПЛК, чтобы уменьшить время цикла ПЛК.
Окно Communication Window ПЛК установлено в режим LIMITED.	Изменить режим на RUN-TO-COMPLETION.

Если ни одно из этих действий невозможно, можно увеличить интервал таймаута. Интервал таймаута устанавливается дополнительным пользовательским параметром (Advanced User Parameter) "crsp_tmot" .

Значение таймаута по умолчанию - 15 секунд. Изменение значений дополнительных пользовательских параметров описано в разделе *Конфигурирование дополнительных пользовательских параметров* главы 6.

Заметьте, что изменение значения таймаута не уменьшает время, требуемое ПЛК для обработки запросов.

Непредвиденный перезапуск Ethernet или ошибки выполнения

Работа в напряженном режиме EGD и/или SRTP может превышать возможности передачи данных и обработки CPUE05. Это может приводить к пропуску обменов EGD, непредвиденным автоматическим перезапускам интерфейса Ethernet CPUE05, или фатальным ошибкам работы интерфейса Ethernet.

Ошибки перезапуска отображается в таблице ошибок ПЛК, как одно или несколько из следующих сообщений:

"Loss of daughterboard" (без значения Event)

"Reset of daughterboard" (без значения Event)

"LAN system-software fault; restarted LAN I/F"
Event = 3, Entry 2 = 1, Entry 3 = 5f0fH

После любой из описанных выше ошибок, интерфейс Ethernet перезапускается автоматически, без ручного вмешательства.

Вышеописанные перезапуски Ethernet могут сопровождаться одним или несколькими из следующих сообщений в таблице ошибок ПЛК:

"Backplane communications with PC fault; lost request" (без значения Event)

"LAN system-software fault; resuming"
Event = 28, Entry 2 = 1, SCode = 95255037H

Ошибки выполнения приостанавливают нормальную работу и вызывают мигание светодиода STAT желтым цветом (мигание отображает код фатальной ошибки). Для восстановления нормальной работы необходимо вручную перезапустить интерфейс Ethernet. Ошибки выполнения с кодами "31" и "33" вызываются большой нагрузкой. Описание кодов фатальных ошибок выполнения приведено в разделе *Проверка светодиодов Ethernet*.

Все обмены по Ethernet Global Data (EGD)по умолчанию выдают код состояния 18 (0012H) при потере или перезапуске интерфейса Ethernet. Обработка EGD возобновится после выполнения перезапуска.

Перезапуск и ошибки выполнения происходят, когда CPUE05 не может обработать требуемый объем запросов EGD и/или SRTP. Т. к. все эти ошибки происходят только, когда CPUE05 подключен к концентратору (hub), в первую очередь следует заменить его на коммутатор (switch). Во вторую

очередь, следует уменьшить количество, размер, или частоту обменов EGD и/или передач через SRTP соединения.

Ошибки несоответствия конфигурации EGD

При использовании Ethernet Global Data, передаваемый обмен (определяемый отправителем) должен соответствовать принимаемому обмену (определяемому получателем). Получатель выдает ошибку, когда размер обмена, полученного из сети, отличается от сконфигурированного.

Эта ошибка отображается в таблице ошибок ПЛК как:

"LAN system-software fault; resuming"
Event = 28, Entry 2 = 1d

Т. к. эта ошибка выдается при каждом приеме обмена несоответствующего размера, журнал регистрации ошибок Ethernet может быстро переполниться ошибками несоответствия.

Причина	Действия
Для исходящего и входящего обменов определен разный размер.	Просмотреть конфликтующие определения обменов отправителя и получателя. Изменить неверное определение обмена так, чтобы передаваемый и принимаемый обмены имели одинаковый размер.

Если требуется, чтобы получатель игнорировал определенную часть принимаемого обмена, убедитесь в правильности длины игнорируемой части. Игнорируемая часть указывается, как количество байт.

Ошибки недостатка ресурсов получателя

Плотный сетевой трафик может вызвать нехватку памяти интерфейса Ethernet, используемого для обмена по сети. Это наиболее часто происходит при плотном трафике Ethernet Global Data (EGD) в загруженной сети. Т. к. сетевой трафик не предсказуем, эта ошибка может возникнуть всегда.

Эта ошибка отображается в таблице ошибок ПЛК как:

"LAN system-software fault; resuming"
Event = 28, Entry 2 = 1

Причина	Действия
Плотный трафик EGD переполняет буферы сетевых данных.	Измените приложение так, чтобы уменьшить количество, размер или частоту исходящих и входящих обменов EGD.
На CPUE05 возникают всплески плотного сетевого трафика.	Проанализируйте сетевой трафик в режиме broadcast и multicast, принимаемый CPUE05. Уменьшите этот трафик, если возможно.

Блокировка монитора станции при большой нагрузке

Длительная большая нагрузка EGD и/или сервера SRTP может использовать все ресурсы интерфейса Ethernet, заблокировав функцию монитора станции. Монитор станции перестает работать как в локальном, так и в удаленном режимах. Интерфейс Ethernet всегда дает более высокий приоритет функциям передачи данных по сравнению с монитором станции. При уменьшении нагрузки монитор станции снова становится доступным.

Эта ситуация не записывается ни в таблицу ошибок ПЛК, ни в журнал регистрации ошибок Ethernet.

Ограничения PING

Для сохранения ресурсов буфера сетевых данных CPUE05 обрабатывает только одно управляющее сообщение ICMP за раз. Запрос ICMP Echo (ping), приходящий в момент, когда CPUE05 обрабатывает другое управляющее сообщение ICMP не учитывается. Когда несколько удаленных устройств пытаются одновременно передать команду “ping” CPUE05, отдельные запросы ping могут игнорироваться в зависимости от синхронизации запросов ping в сети.

CPUE05 может инициировать запросы ping другим устройствам сети с помощью команды “ping” монитора станции. Последовательность запросов ping ограничена одним устройством за раз.

Неучтенные запросы ping не записываются в таблицу ошибок ПЛК и журнал регистрации ошибок Ethernet.

Таймаут соединения SRTP

Когда удаленный SRTP клиент внезапно отсоединяется от CPUE05 (например, из-за отсоединения кабеля Ethernet), TCP соединение пытается восстановить связь. Соединение SRTP в CPUE05 остается открытым в течение приблизительно 5 минут, пока TCP пытается повторно соединиться; в течение этого интервала SRTP соединение недоступно. Если все SRTP соединения CPUE05 используются или недоступны по иной причине, новый SRTP клиент должен ждать, пока не истечет время повторного подключения TCP в установленном соединении.

Таймаут подключения SRTP - нормальное вероятное состояние, совместимое с другими изделиями ПЛК GE Fanuc.

Глава
14

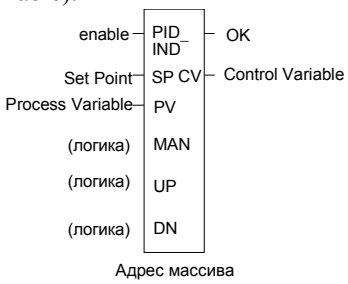
Функция ПИД

Эта глава описывает функцию ПИД (пропорционально-интегрально-дифференциальный регулятор), которая используется для управления с обратной связью. Функция ПИД сравнивает значение обратной связи с требуемой уставкой процесса и корректирует на основе рассогласования управляющую переменную.

- Формат функции ПИД
- Работа функции ПИД
- Блок параметров для функции ПИД
- Выбор алгоритма ПИД-регулирования
- Определение характеристик процесса
- Установка параметров, включая коэффициенты усиления
- Пример вызова ПИД-регулятора

Формат функции ПИД

Функция ПИД использует ПИД-коэффициенты и другие параметры, сохраненные в массиве из 40 16-битных слов, для выполнения ПИД-алгоритма в требуемом интервале времени. Все параметры являются 16-битными целыми словами. Это позволяет использовать память %AI для входной переменной процесса (Process Variable) и %AQ для выходной управляющей переменной (Control Variable).



Функция ПИД не пропускает через себя питание, если имеется ошибка в параметрах конфигурации. Состояние функции при изменении данных может контролироваться с помощью временной катушки.

Параметры функции ПИД

Вход/ Выход	Варианты	Описание
enable	питание	ПИД-алгоритм выполняется при наличии разрешающего сигнала на этом контакте.
SP	I, Q, M, T, G, R, AI, AQ, константа	Уставка регулятора или процесса. Функция ПИД настраивает выход Control Variable так, чтобы переменная Process Variable соответствовала уставке (нулевая ошибка).
PV	I, Q, M, T, G, R, AI, AQ	Вход Process Variable от управляемого процесса, часто вход %AI.
MAN	питание	Когда установлен в 1 (через контакт), блок ПИД находится в ручном режиме. Если ручной режим выключен, блок ПИД находится в автоматическом режиме.
UP	питание	Когда UP и MAN установлены в 1, увеличивает значение переменной Control Variable на 1.*
DN	питание	Когда DN и MAN установлены в 1, уменьшает значение переменной Control Variable на 1.*
Address	R	Размещение управляющего блока ПИД (пользовательские и внутренние параметры). Использует 40 слов %R, которые не могут использоваться другими функциями.
ok	питание, нет	OK устанавливается в 1, когда функция выполняется без ошибок. При возникновении ошибки устанавливается в 0.
CV	I, Q, M, T, G, R, AI, AQ	Выход управления (Control Variable) процессом, часто выход %AQ.

* Увеличивает (UP параметр) или уменьшает (DN параметр) на один (1) за одно обращение к функции ПИД.

Многие параметры должны быть заданы в единицах переменной процесса (PV) или управляющей переменной (CV). Например, вход уставки (SP) должен быть масштабирован в том же диапазоне, что и переменная процесса, т.к. ПИД-блок вычисляет рассогласование путем вычитания этих двух выходов. Переменные PV и CV могут принимать значения от –32000 до 32000 и от 0 до 32000 для отображения переменных в диапазоне от 0.00% до 100.00%. Переменные PV и CV не обязаны иметь одинаковую шкалу.

Работа функции ПИД

Работа в автоматическом режиме

Функция ПИД может вызываться каждый цикл подачей питания на вход Enable при отсутствии питания на входе Manual. Блок сравнивает текущее время ПЛК со временем последнего вызова ПИД, сохраненным во внутреннем массиве RefArray. Если разница больше периода выборки, определенного в третьем слове (%Ref+2) массива RefArray, ПИД-алгоритм выполняется. И время последнего вызова, и выход CV обновляются. В автоматическом режиме выход CV помещается в параметр Manual Command %Ref+13.

Работа в ручном режиме

Блок ПИД переводится в ручной режим подачей питания на входные контакты Enable и Manual. На выходе CV устанавливается значение параметра Manual Command %Ref+13. Если на входы UP или DN подано питание, слово Manual Command увеличивается или уменьшается на одну единицу CV за каждый вызов ПИД. Для более быстрого ручного изменения выхода CV, также возможно добавление или вычитание некоторого количества единиц CV непосредственно к/от слов Manual Command.

Блок ПИД использует параметры CV Upper Clamp и CV Lower Clamp для ограничения выхода CV. Если определен параметр Minimum Slew Time, он используется для ограничения скорости изменения выхода CV. Если пределы амплитуды CV или скорости изменения превышены, значение, сохраненное в интеграторе, регулируется так, чтобы CV не выходил за пределы. Эта коррекция интегратора означает, что если даже рассогласование пытается вывести выход CV за установленные границы в течение долгого периода времени, выход CV пойдет назад, как только значение рассогласования изменит знак.

Такая работа, когда параметр Manual Command отслеживает выход CV в автоматическом режиме и устанавливает выход CV в ручном режиме, обеспечивает безударный переход между автоматическим и ручным режимами. Параметры CV Upper, Lower Clamps и Minimum Slew Time оказывают воздействие на выход CV и в ручном режиме, и внутреннее значение, сохраненное в интеграторе, обновляется. Это означает, что и в ручном режиме выход CV не будет изменяться быстрее, чем это установлено параметром Minimum Slew Time, и он не сможет выйти за пределы параметров CV Upper Clamp или CV Lower Clamp.

Интервал времени для функции ПИД

Функция ПИД не будет выполняться чаще, чем 1 раз за 10 миллисекунд. Если она сконфигурирована для выполнения каждый цикл, а время цикла составляет менее 10 миллисекунд, функция ПИД не будет выполняться, пока не пройдет достаточное количество циклов, чтобы время с момента последнего обращения стало 10 миллисекунд или более. Например, если время цикла составляет 9 миллисекунд, функция ПИД выполняется через цикл, в результате время между выполнением составит 18 миллисекунд. Конкретная функция ПИД не должна вызываться чаще одного раза за цикл.

Наибольший возможный интервал между выполнением составляет 10.9 минуты. Функция ПИД компенсирует действительное время, прошедшее с момента последнего выполнения, в пределах 100 микросекунд.

Алгоритм ПИД выполняется, только если время, прошедшее с момента последнего выполнения, больше или равно времени выполнения алгоритма плюс период выборки. Если для периода выборки установлено значение 0, функция выполняется каждый раз, когда ей это разрешено, однако, не чаще чем один раз в 10 миллисекунд, как указано выше.

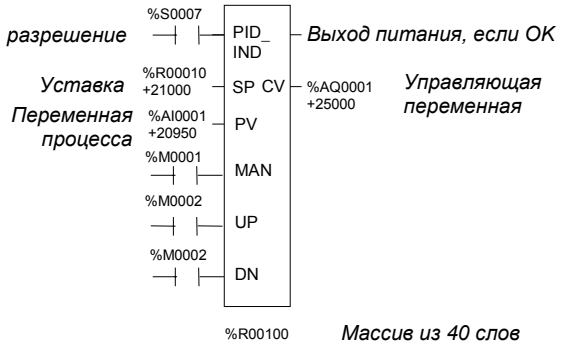
Масштабирование входа и выхода

Все параметры функции ПИД являются 16-битными целыми словами для обеспечения совместимости с 16-битными аналоговыми переменными процесса. Некоторые параметры должны быть заданы в единицах переменной процесса (PV) или управляющей переменной (CV).

Вход уставки (SP) должен быть масштабирован в том же диапазоне, что и переменная процесса, т. к. блок ПИД вычисляет рассогласование путем вычитания этих двух выходов. Переменные PV и CV не обязаны иметь одинаковую шкалу. Переменные PV и CV могут принимать значения от –32000 до 32000 и от 0 до 32000 для отображения переменных в диапазоне от 0.00% до 100.00%. Если переменные PV и CV не используют одинаковое масштабирование, множители включаются в коэффициенты ПИД.

Пример функции ПИД

Пример, показанный ниже, включает типовые входы.



Блок параметров для функции ПИД

Блок параметров для функции ПИД занимает 40 слов в памяти %R. Многие из этих 40 слов используются ПЛК и не конфигурируются. Каждая вызываемая функция ПИД должна использовать различные области памяти, состоящие из 40 слов, даже если все 13 конфигурируемых параметров совпадают.

Первые 13 слов блока параметров должны быть определены до выполнения функции ПИД. Ноль может быть использован для большинства значений по умолчанию. После того, как значения параметров функции ПИД выбраны, они могут быть определены как константы функцией BLKMOV и изменены программой в случае необходимости.

Внутренние параметры в массиве RefArray

Функция ПИД читает 13 параметров и использует остальные слова массива RefArray для внутренних целей. Эти значения обычно не требуется изменять. Если вы вызываете блок ПИД в автоматическом режиме после большого перерыва, вы можете использовать функцию SVC_REQ 16 для загрузки текущего времени ПЛК в %Ref+23, чтобы обновить время последнего вызова блока ПИД во избежание скачкообразных изменений на интеграторе. Если вы установили младший бит управляющего слова Control Word (%Ref+14) в 1, следующие четыре бита управляющего слова Control Word должны быть установлены для управления входными контактами блока ПИД, и внутренние параметры SP и PV должны быть установлены, т. к. вы перехватили контроль над блоком ПИД у релейной логики.

	Параметр	**Единицы измерения**	**Диапазон**	**Описание**
Адрес	Loop Number Номер регулятора	Integer	0 - 255.	Необязательный номер блока ПИД. Это обеспечивает общую идентификацию в ПЛК по номеру регулятора, определенному устройством операторского интерфейса.
Адрес +1	Algorithm Алгоритм	-	Устанавливается ПЛК	1 = алгоритм ISA 2 = независимый алгоритм
Адрес +2	Sample Period Период выборки	10 мс	0 (каждый цикл) - 65535 (10.9 мин) Не менее 10 мс.	Наименьшее время, кратное 10 мс, между выполнением ПИД-алгоритма. Например, используйте 10 для периода выборки 100 мс.

	Параметр	Единицы измерения	Диапазон	Описание
Адрес +3 Адрес +4	Dead Band + and Dead Band – Зона нечувствитель-ности + и Зона нечувстви-тельности -	Единицы PV	0 - 32000 (+ никогда отрицательное) (- никогда положительное)	Верхняя (+) и нижняя (-) границы зоны нечувствительности определяются целым числом в единицах PV. Если зона нечувствительности не требуется, эти значения должны равняться 0. Если ПИД-ошибка (SP - PV) или (PV - SP) выше значения (-) и ниже значения (+), ПИД считает ошибку равной 0. Если границы не равны 0, значение (+) должно быть больше 0, а значение (-) должно быть меньше 0, иначе блок ПИД не будет работать. Оставьте границы равными 0 на время настройки коэффициентов ПИД. Зона нечувствительности добавляется, чтобы избежать малых изменений выхода CV из-за случайных ошибок на входе.
Адрес +5	Proportional Gain –Kp Пропорциональ-ный коэффициент (В версии ISA Controller gain, Kc)	0.01 CV%/PV%	0 - 327.67%	Изменение управляющего выхода Control Variable в единицах CV при изменении рассогласования на 100 единиц PV. Значение 450, введенное для Kp, отображается как 4.50 и результат в виде Kp*Error/100 или 450*Error/100 передается на выход ПИД. Kp обычно является первым коэффициентом, устанавливаемым при настройке ПИД-регулятора. (Error – рассогласование).
Адрес +6	Derivative Gain-Kd Коэффициент Дифференци-рования	0.01 с	0 - 327.67 с	Изменение управляющего выхода Control Variable в единицах CV, если рассогласование или PV изменяется на 1 единицу PV каждые 10мс. Вводится в десятках миллисекунд. Например, значение 120, введенное для Kd, отображается как 1.20с и результат в виде Kd * delta Error/dt или 120*4/3 передается на выход ПИД, если рассогласование изменялось на 4 единицы PV каждые 30мс. Kd может быть использован для ускорения отклика регулятора, но он очень чувствителен к шуму на входе PV. (Error – рассогласование).
Адрес +7	Integral Rate-Ki Коэффициент интегрирования	Повторов/1000 с	0 - 32.767 повторов/с	Изменение управляющего выхода Control Variable в единицах CV, если значение рассогласования постоянно равнялось 1 единице PV. Отображается как 0.000 повторов/с (подразумеваются 3 знака после запятой). Например, значение 1400, введенное для Ki, отображается как 1.400 повторов/с и результат в виде Ki * Error *dt или 1400 * 20 * 50/1000 передается на выход ПИД, для рассогласования 20 единиц PV и времени цикла ПЛК 50мс (Период выборки - 0). Ki обычно является вторым коэффициентом, устанавливаемым после Kp.
Адрес +8	CV Bias/Output Offset Смещение выхода	Единицы CV	-32000 - 32000 (добавляется к выходу интегратора)	Количество единиц CV, добавляемое к выходу ПИД до того, как учитываются ограничения по скорости и амплитуде. Смещение может быть использовано для установки ненулевого значения CV, если используется только пропорциональный коэффициент Kp, или для управления выходом этого ПИД-регулятора другим регулятором.

	Параметр	Единицы измерения	Диапазон	Описание
Адрес +9 Адрес +10	CV Upper and Lower Clamps Верхняя и нижняя границы выхода CV	Единицы CV	-32000 - 32000 (>%Ref+10)	Кол-во единиц CV, определяющее верхнее и нижнее значение CV. Ввод этих значений обязателен. Граница Upper Clamp должна иметь большее значение, чем граница Lower Clamp, иначе блок ПИД не будет работать. Они обычно используются для определения пределов на основе физических ограничений выхода CV. Они также используются для масштабирования изображения диаграммы CV. Блок имеет коррекцию интегратора, изменяющую значение интегратора при достижении границ.
Адрес +11	Minimum Slew Time Минимальное время прохода	Секунд/полный проход	0 (нет) - 32000 с для изменения на 32000 единиц CV	Минимальное кол-во секунд, за которое выход CV должен измениться от 0 до 100% или 32000 единиц CV. Это время определяет скорость изменения выхода CV. Если этот параметр больше 0, CV не может измениться более чем на 32000 единиц CV за время периода выборки, деленное на минимальное время прохода. Например, если период выборки составляет 2.5 секунды, а минимальное время прохода равно 500 с, CV не может измениться более чем на 32000*2.5/500 или 160 единиц CV за один цикл работы блока ПИД. Значение интегратора подстраивается, если ограничение скорости превышено. Если минимальное время прохода равно 0, то ограничение скорости отсутствует. Установите минимальное время прохода равным 0 при настройке коэффициентов блока ПИД.

	Параметр	Единицы измерения	Диапазон	Описание
Адрес +12	Config Word Слово конфигурации	Используются младшие 5 битов	Биты от 0 до 2 для знака рассогласования, полярности выхода, способа вычисления производной.	Младшие 5 битов этого слова используются для изменения трех стандартных настроек блока ПИД. Остальным битам должно быть установлено значение 0. Установите младший бит в 1 для изменения способа вычисления рассогласования: с нормального (SP – PV) на (PV – SP), в результате чего меняется знак обратной связи. Это предназначено для обратного управления, при котором CV должен уменьшаться, когда PV растет. Установите второй бит в 1 для инвертирования полярности выхода. Установите четвертый бит в 1 для изменения дифференцирующего воздействия с использующего изменения рассогласования на использующее изменение обратной связи PV. Младшие 5 бит слова конфигурации Config Word подробно описаны ниже. Бит 0: Определение рассогласования. Когда бит - 0, рассогласование определяется, как SP - PV. Когда этот бит - 1, рассогласование определяется, как PV - SP. Бит 1: Полярность выхода. Когда бит - 0, на выход CV подается вычисленное значение ПИД. Когда этот бит - 1, на выход CV подается инвертированный результат вычисления ПИД. Бит 2: Дифференцирующее воздействие на PV. Когда бит - 0, дифференцирующее воздействие оказывается на рассогласование. Когда этот бит - 1, дифференцирующее воздействие оказывается на PV. Все остальные биты должны быть установлены в 0. Бит 3: Действие зоны нечувствительности. Когда бит - 0, действие зоны нечувствительности не выбрано. Если рассогласование находится в пределах зоны нечувствительности, тогда рассогласование равно 0. Иначе говоря, границы зоны нечувствительности не влияют на рассогласование. Когда бит - 1, действие зоны нечувствительности выбрано. Если рассогласование находится в пределах зоны нечувствительности, тогда рассогласование принудительно устанавливается в 0. Однако, если рассогласование находится вне зоны нечувствительности, тогда оно уменьшается на значение границы зоны нечувствительности (рассогласование = рассогласование – граница зоны нечувствительности). Бит 4: Действие коррекции интегратора. Когда бит - 0, коррекция интегратора использует обратное вычисление. Когда выход находится на ограничении, она заменяет накопленное значение Y значением, необходимым, чтобы расчетное значение выхода точно соответствовало ограничению. Когда бит - 1, она заменяет накопленное значение Y значением Y на момент начала вычисления. В результате, это значение используется до тех пор, пока выход находится на ограничении. Биты являются степенью 2. Например, чтобы установить Config Word, вы должны прибавить 1 чтобы изменить бит 0 с SP–PV на PV–SP, или прибавить 2 чтобы изменить бит 1 с CV = выход ПИД на CV = – выход ПИД, или прибавить 4 чтобы изменить бит 3 с изменения Error на изменения PV, и т. д..

<table>
<tr><th></th><th>Параметр</th><th>Единицы измерения</th><th>Диапазон</th><th>Описание</th></tr>
<tr><td>Адрес +13</td><td>Manual Command
Управляющее воздействие в ручном режиме</td><td>Единицы CV</td><td>Отслеживает CV в автоматическом режиме или устанавливает CV в ручном режиме</td><td>Устанавливается в текущее значение выхода CV, когда блок ПИД находится в автоматическом режиме. Когда блок переключается в ручной режим, это значение используется для установки выхода CV и внутреннего значения интегратора, с учетом ограничений по параметрам Upper Clamp и Lower Clamp и Slew Time.</td></tr>
<tr><td>Адрес +14</td><td>Control Word
Управляющее слово</td><td>Устанавли-вается ПЛК, если не установлен бит 1.</td><td>Устанавливается ПЛК, если не установлен бит 1.</td><td>Если младший бит внешнего управления установлен в 1, это слово и другие внутренние параметры SP, PV и CV должны быть использованы для удаленного управления этим блоком ПИД (см. ниже). Это позволяет удаленному устройству операторского интерфейса, такому как компьютер, взять управление на себя. Предостережение: если вы не хотите, что бы это произошло, убедитесь, что управляющее слово установлено в 0. Если младший бит установлен в 0, следующие 4 бита могут быть прочитаны для контроля состояния входных контактов ПИД, пока на контакт Enable (разрешение) подано питание.
Структура дискретных данных первых пяти бит а порядке следования:
<table>
<tr><th>Бит:</th><th>Значе-ние слова:</th><th>Функция:</th><th>Состояние или внешнее воздействие, если бит 0 установлен в 1:</th></tr>
<tr><td>0</td><td>1</td><td>Внешнее управление</td><td>Если 0, биты ниже информационные. Если 1, задайте их извне.</td></tr>
<tr><td>1</td><td>2</td><td>Ручной /Автомат</td><td>Если 1, блок в ручном режиме, в противном случае – в автоматическом режиме.</td></tr>
<tr><td>2</td><td>4</td><td>Разрешение</td><td>Нормальное значение 1, в противном случае блок не вызывается.</td></tr>
<tr><td>3</td><td>8</td><td>UP /увеличение</td><td>Если 1 и Бит 1 в 1, CV будет увеличиваться каждый вызов блока.</td></tr>
<tr><td>4</td><td>16</td><td>DN /уменьшение</td><td>Если 1 и Бит 1 в 1, CV будет уменьшаться каждый вызов блока.</td></tr>
</table></td></tr>
<tr><td>Адрес +15</td><td>Internal SP
Внутренняя уставка</td><td>Устанавли-вается и ис-пользуется ПЛК</td><td>Не конфигурируется</td><td>Отслеживает вход SP; должна быть установлена извне, если: бит внешнего управления = 1.</td></tr>
<tr><td>Адрес +16</td><td>Internal CV
Внутренняя CV</td><td>“</td><td>“</td><td>Отслеживает выход CV.</td></tr>
<tr><td>Адрес +17</td><td>Internal PV
Внутренняя PV</td><td>“</td><td>“</td><td>Отслеживает вход PV; должна быть установлена извне, если: бит внешнего управления = 1.</td></tr>
<tr><td>Адрес +18</td><td>Output
Выход</td><td>“</td><td>“</td><td>Значение слова со знаком представляет выход функционального блока перед возможной инверсией. Если инверсия выхода не сконфигурирована, и бит полярности выхода в управляющем слове установлен в 0, это значение равно выходу CV. Если выбрана инверсия и бит полярности выхода установлен в 1, это значение равно инвертированному значению выхода CV .</td></tr>
</table>

	Параметр	Единицы измерения	Диапазон	Описание
Адрес +19	Diff Term Storage Дифферен-циальная. составляющая			
Адрес +20 Адрес +21	Int Term Storage Внутренняя память			Используются контроллером для хранения промежуточных значений. Не записывайте в эти ячейки памяти.
Адрес +22	Slew Term Storage Время прохода			
Адрес +23 до Адрес +25	Clock Часы			Хранит внутреннее время (время последнего выполнения блока ПИД). Не записывайте в эти ячейки памяти.
Адрес +26	Y Remainder Storage Остаток Y			Содержит остаток, используемый в интеграторе при установившемся состоянии (рассогласование равно 0).
Адрес +27 Адрес +28	SP, PV Lower and Upper Range Границы диапазона значений SP и PV	Единицы PV	-32000 - 32000	Вспомогательные целые значения в единицах PV, определяющие верхние и нижние отображаемые значения.(Ref +27 должен быть меньше, чем Ref+28)
Адрес +29 до Адрес +39	Reserved Зарезервиро-ваны	Нет	Не конфигурируются	29-34 зарезервированы для внутреннего использования; 35-39 зарезервированы для внешнего использования. Не используйте эти ячейки.

Выбор алгоритма ПИД (PIDISA или PIDIND) и коэффициенты

Блок ПИД может быть запрограммирован с использованием либо независимой версии (PID _IND), либо стандартной ISA версии (PID_ISA) алгоритма ПИД. Единственная разница между алгоритмами заключается в том, как определяются коэффициенты интегрирования и дифференцирования.

Оба алгоритма ПИД вычисляют рассогласование как SP - PV, хотя вычисление может быть изменено на инверсный режим PV – SP установкой параметра Error Term (младший бит 0 в слове Config Word %Ref+12) в 1.

Инверсный режим может быть использован, если вы хотите, чтобы выход CV изменялся в противоположном направлении относительно направления изменения входа PV (CV уменьшается при росте PV), в отличие от нормального режима, когда CV растет при росте PV.

Error* = (SP - PV) или (PV - SP), если младший бит слова Config Word установлен в 1

*Здесь и далее – Error = рассогласование

Дифференцирующая составляющая обычно вычисляется на основе изменения рассогласования, произошедшего с момента последнего выполнения функции ПИД, что может вызвать большое изменение выхода при изменении значения SP (уставки). Если это нежелательно, третий бит слова Config Word может быть установлен в 1, чтобы дифференцирующая составляющая вычислялась на основе изменения значения PV. Шаг дифференцирования (dt) определяется вычитанием времени последнего выполнения данного блока ПИД из текущего времени ПЛК.

dt = Текущее время ПЛК – Время последнего выполнения блока ПИД

производная = (Error - предыдущее Error)/dt

или (PV - предыдущее PV)/dt, если 3 бит слова Config Word установлен в 1

Независимый алгоритм ПИД (PID _IND) вычисляет выход, как:

Выход ПИД = Kp * Error + Ki * Error * dt + Kd * производная + CV Bias

Стандартный алгоритм ISA (PID _ISA) имеет другую форму:

Выход ПИД = Kc * (Error + Error * dt/Ti + d * производная) + CV Bias

где Kc – коэффициент усиления контроллера, Ti – время интегрирования и Td – время дифференцирования. Преимущество алгоритма ISA состоит в том, что настройка Kc изменяет дифференциальную и интегральную составляющую, так же, как и пропорциональную, что облегчает настройку регулятора. Если у вас есть коэффициенты ПИД или времена Ti и Td, то используйте следующие соотношения:

Kp = Kc Ki = Kc/Ti and Kd = Kc/Td

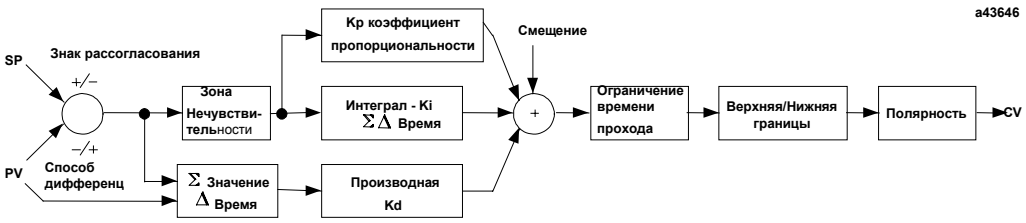
Чтобы преобразовать их для использования в качестве входов пользовательских параметров ПИД.

Слагаемое CV Bias (смещение CV) – дополнительная составляющая, отличная от компонентов ПИД. Она может потребоваться, если вы используете только коэффициент пропорциональности Kp и хотите, что бы выход CV был отличен от 0, когда PV равно SP и рассогласование равно 0. В этом случае, установите значение CV Bias равным требуемому значению CV, когда PV равно SP. CV Bias также может быть использован для прямого управления, когда для установки выхода CV используется другой ПИД регулятор или управляющий алгоритм.

Если используется коэффициент интегрирования Ki, CV Bias обычно равняется 0, т. к. интегратор выполняет смещение автоматически. Просто стартуйте в ручном режиме и используйте слово Manual Command (%Ref+13) для установки интегратора в требуемое значение CV, затем переключитесь в автоматический режим. Это также работает, если Ki = 0, за исключением того, что интегратор не будет настроен по рассогласованию после перехода в автоматический режим.

Независимый алгоритм (PIDIND)

Следующая диаграмма показывает, как работают алгоритмы ПИД:



Алгоритм ISA (PIDISA) точно такой же, за исключением того, что коэффициент Kp выносится за скобки относительно Ki и Kd, так, что коэффициент интегрирования равен Kp * Ki, а коэффициент дифференцирования равен Kp * Kd. Знак рассогласования, способ дифференцирования и полярность устанавливаются битами слова Config Word.

Ограничения амплитуды и скорости изменения выхода CV

Блок не посылает расчетный выход ПИД непосредственно на CV. Оба алгоритма ПИД могут накладывать ограничения по амплитуде и скорости изменения на выход Control Variable (CV). Максимальная скорость изменения определяется делением максимального 100% значения выхода CV (32000) на минимальное время прохода (Minimum Slew Time), если его значение больше 0. Например, если минимальное время прохода составляет 100 секунд, ограничение скорости изменения будет равно 320 единиц CV в секунду. Если время dt с момента последнего выполнения составило 50 миллисекунд, новое значение выхода CV не может измениться более, чем на 320*50/1000 или 16 единиц CV по сравнению с предыдущим значением выхода CV.

Затем выход CV сравнивается со значениями верхней и нижней границ (CV Upper Clamp и CV Lower Clamp). Если хоть один из пределов превышен, выход CV устанавливается в значение ограничения. Если пределы скорости изменения или амплитуды превышены при изменении CV, значение внутреннего интегратора корректируется в соответствии со значением ограничения.

Наконец, блок проверяет полярность выхода (2 бит слова Config Word %Ref+12) и изменяет знак выхода, если бит находится в состоянии 1.

CV = Ограниченный выход ПИД или
- Ограниченный выход ПИД, если установлен бит Output Polarity

Если блок находится в автоматическом режиме, окончательное значение CV помещается в слово Manual Command %Ref+13. Если блок находится в ручном режиме, уравнение ПИД пропускается, т. к. CV устанавливается словом Manual Command, но все ограничения скорости изменения и амплитуды все равно проверяются. Это означает, что слово Manual Command не может установить выход большим, чем значение CV Upper Clamp или меньшим, чем значение CV Lower Clamps, и выход не может изменяться быстрее, чем это установлено параметром Minimum Slew Time.

Период выборки и планирование выполнения блоков ПИД

Блок ПИД является цифровой реализацией аналоговой управляющей функции, поэтому время выборки dt в уравнении выхода ПИД не является бесконечно малой величиной, как в случае аналогового управления. Большинство управляемых процессов может быть аппроксимировано звеном первого или второго порядка, возможно с большим транспортным запаздыванием. Блок ПИД устанавливает выход CV и использует вход обратной связи процесса PV для определения рассогласования, чтобы установить очередное значение выхода CV. Ключевым параметром процесса является общая постоянная времени, указывающая, насколько быстро параметр PV реагирует на изменение значения CV. Как указано в разделе “Установка коэффициентов регулятора”, общая константа времени, Tp+Tc, для систем первого порядка – это время, необходимое для того, чтобы PV достигло 63% от своего окончательного значения при ступенчатом изменении CV. Блок ПИД не сможет управлять процессом, если значение периода выборки не будет значительно меньше половины общей постоянной времени. Больший период выборки приведет к нестабильной работе.

Период выборки не должен быть больше, чем общая постоянная времени деленная на 10 (или, в худшем случае, на 5). Например, если PV достигает около 2/3 от своего окончательного значения за 2 секунды, период выборки должен быть менее, чем 0.2 секунды, или 0.4 секунды, в худшем случае. С другой стороны, период выборки не должен быть слишком маленьким, таким, как общая постоянная времени деленная на 1000, иначе значение Ki * Рассогласование * dt для интегратора ПИД будет округляться до 0. Например, очень медленный процесс, занимающий 10 часов или 36000 секунд для достижения уровня 63%, должен иметь период выборки 40 секунд или более.

Если процесс не очень быстрый, обычно нет необходимости использовать период выборки равный 0 для обеспечения выполнения алгоритма ПИД каждый цикл ПИД. Если используются много ПИД-регуляторов с периодом выборки, большим времени цикла, время цикла может колебаться в широких пределах, если многие регуляторы заканчивают выполнение алгоритма в одно время. Простым решением является циклический сдвиг одного или нескольких бит со значением 1 по массиву, заполненному нулями, для вызова отдельных блоков ПИД при помощи входа «enable».

Определение характеристик процесса

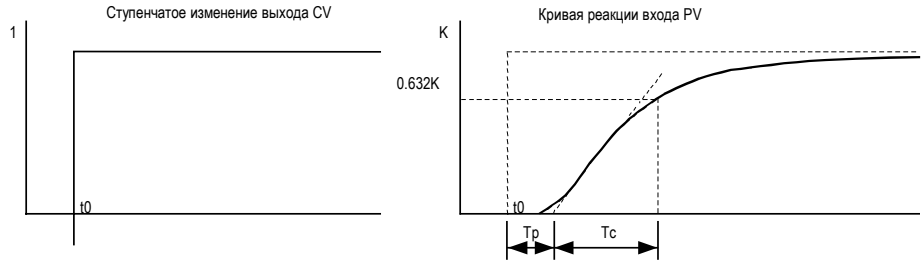
Коэффициенты ПИД-регулятора Kp, Ki и Kd, определяются характеристиками управляемого процесса. При настройке ПИД-регулятора имеются два ключевых вопроса:

1. Насколько велико изменение PV при изменении CV на фиксированное значение, или каков коэффициент усиления разомкнутого контура?
2. Как быстро происходит отклик системы, или как быстро изменяется PV после ступенчатого изменения выхода CV?

Многие процессы могут быть аппроксимированы звеном первого или второго порядка, возможно с большим транспортным запаздыванием. Передаточная функция в частотной области для системы первого порядка с транспортным запаздыванием:

PV(s)/CV(s) = G(s) = K * e **(-Tp s)/(1 + Tc s)

Построение во временной области реакции на ступенчатое возмущение в момент времени t0 дает переходную характеристику системы:



Следующие параметры модели процесса могут быть определены по переходной характеристике PV:

K	Коэффициент усиления открытого контура = окончательное изменение PV/ изменение CV во время t0
Tp	Время задержки процесса от t0 до начала изменения значения PV.
Tc	Постоянная времени процесса первого порядка, время необходимое после Tp, чтобы PV достигло значения 63.2% от своего конечного значения.

Обычно самым простым способом измерения этих параметров является перевод блока ПИД в ручной режим и изменение выхода CV маленькими шагами, путем изменения слова Manual Command %Ref+13, и построение графика изменения PV во времени. Для медленных процессов это может быть сделано вручную, но для более быстрых процессов необходим самописец или компьютерный регистратор. Величина шага CV должна быть достаточно большой для получения видимых изменений PV, но не слишком большой,

чтобы не сорвать измеряемый процесс. Подходящая величина шага может составлять от 2 до 10% от разницы между значениями CV Upper Clamp и CV Lower Clamp.

Установка параметров, включая настройку коэффициентов

Так как все параметры ПИД полностью зависят от регулируемого процесса, предопределенных значений не существует, однако, обычно легко найти подходящие коэффициенты регулирования.

1. Установите все пользовательские параметры в 0, затем установите ожидаемые верхние и нижние значения параметрам CV Upper Clamp и CV Lower Clamps. Установите значение периода выборки от расчетная постоянная времени процесса/10 до расчетная постоянная времени процесса/100.
2. Переведите блок в ручной режим и измените значение слова Manual Command (%Ref+13), чтобы проверить, может ли CV дойти до верхней и нижней границы. Запишите значение PV в некоторой точке CV и загрузите его в SP.
3. Установите маленькое значение, такое как 100 * Max. CV/Max. PV, коэффициенту Kp и выключите ручной режим. Изменяйте SP от 2 до 10% от максимального диапазона PV и наблюдайте отклик PV. Увеличьте Kp, если PV изменяется слишком медленно или уменьшите Kp, если PV перерегулируется и колеблется, не достигая устойчивого значения.
4. Когда Kp определен, начинайте увеличивать Ki, чтобы получить колебания, успокаивающиеся в течение 2 – 3 циклов. Для этого может потребоваться уменьшение Kp. Также попробуйте различные размеры шага и рабочие точки CV.
5. После определения подходящих коэффициентов Kp и Ki, попробуйте добавить коэффициент Kd, чтобы добиться более быстрого отклика, не вызывающего колебаний. Kd часто не нужен, он не будет работать при наличии шума на входе PV.
6. Проверьте коэффициенты в других рабочих точках SP и, при необходимости, добавьте зону нечувствительности и минимальное время прохода. Некоторые инверсные процессы могут потребовать установки в слове Config Word битов Error Sign (знак рассогласования) или Polarity (полярность).

Установка коэффициентов регулирования с помощью аппроксимации Циглера и Николса

Если три параметра модели процесса - K, Tp и Tc, определены, они могут быть использованы для приблизительной оценки начальных коэффициентов ПИД-регулятора. Следующая аппроксимация обеспечивает хорошую реакцию на возмущения системы с коэффициентом затухания, равным 1/4. Коэффициент затухания – отношение второго пика реакции замкнутого регулятора к первому.

1. Вычисляем скорость реакции:

   R = K/Tc

2. Только для пропорционального коэффициента, вычисляем Kp как:

   Kp = 1/(R * Tp) = Tc/(K * Tp)

   Для пропорционального и интегрального управления, используем:

   Kp = 0.9/(R * Tp) = 0.9 * Tc/(K * Tp) Ki = 0.3 * Kp/Tp

   Для пропорционального, интегрального и дифференциального управления,:

   Kp = G/(R * Tp), где G от 1.2 до 2.0
   Ki = 0.5 * Kp/Tp
   Kd = 0.5 * Kp * Tp

3. Проверяем, находится ли период выборки в диапазоне

   от (Tp + Tc)/10 до (Tp + Tc)/1000

Метод идеальной настройки

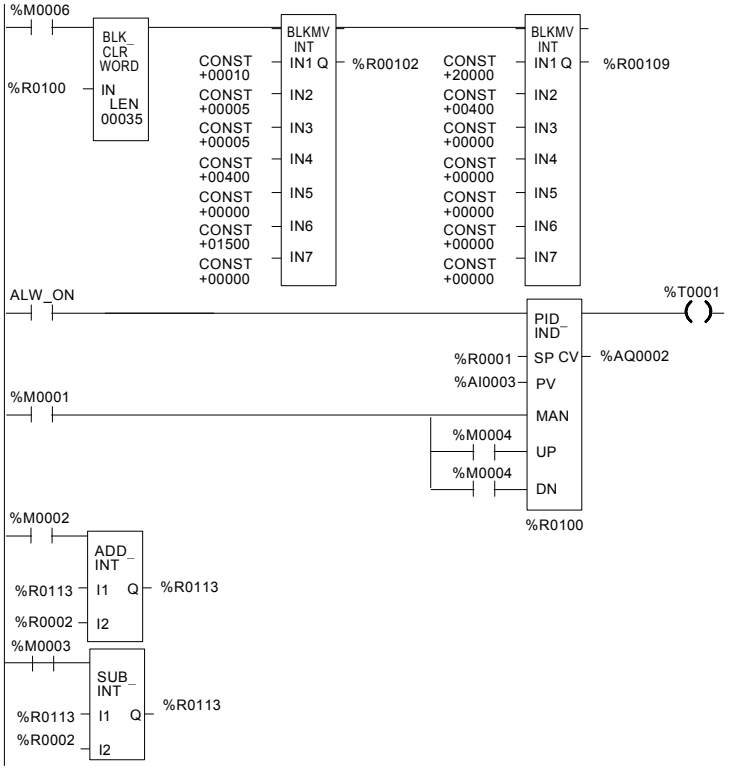
Процедура “Идеальной настройки" обеспечивает наилучшую реакцию на изменения SP, ограниченную только задержкой процесса Tp или временем запаздывания.

Kp = 2 * Tc/(3 * K * Tp)
Ki = Tc
Kd = Ki/4 если используется дифференциальная составляющая

Когда начальные коэффициенты определены, преобразуйте их в целые числа. Рассчитайте коэффициент процесса K, как изменение входа PV в единицах PV деленное на скачкообразное изменение выхода в единицах CV, но не в инженерных единицах процесса. Укажите все времена в секундах. После определения Kp, Ki и Kd, Kp и Kd могут быть умножены на 100 и введены, как целые числа, а Ki может быть умножен на 1000 и введен в пользовательский параметр %RefArray.

Пример вызова функции ПИД

Следующий пример ПИД имеет период выборки 100мс, Kp - 4.00 и Ki - 1.500. Уставка хранится в регистре %R0001, управляющий выход - %AQ0002, а переменная процесса - %AI0003. Параметры CV Upper Clamp и CV Lower Clamp установлены в значения 20000 и 4000, также включена дополнительная небольшая зона нечувствительности от +5 до -5. Массив RefArray из 40 слов начинается с регистра %R0100. Обычно, пользовательские параметры устанавливаются в массиве RefArray, но переменная %M0006 может быть установлена для повторной инициализации 14 слов, начинающихся с регистра %R0102 (%Ref+2), константами, сохраненными в логике (полезный метод).

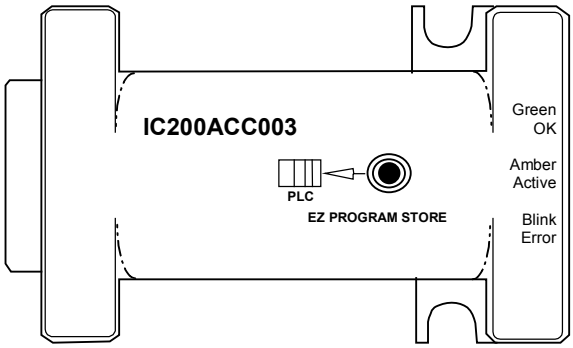


Блок может быть переведен в ручной режим переменной %M1 и параметр Manual Command, %R113, может быть настроен. Биты %M4 или %M5 могут использоваться для увеличения или уменьшения значения регистра %R113 и выхода CV и интегратора ПИД на 1 каждые 100 мс. Для более быстрой ручной работы биты %M2 и %M3 могут использоваться для прибавления или вычитания значения регистра %R2 к/от регистра %R113 в каждом цикле ПЛК. Выход %T1 включен при отсутствии ошибок в блоке ПИД.

Глава
15

Устройство хранения программ EZ

В этой главе описывается устройство хранения программ EZ VersaMax®, которое может использоваться для переноса программы, конфигурации и данных с одного ПЛК на другие того же типа.



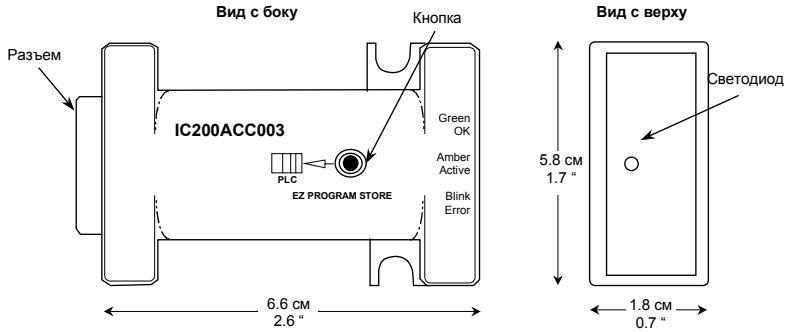
Содержание этой главы:

- Описание устройства хранения программ EZ
- Использование устройства хранения программ EZ
- Чтение/Запись/Проверка данных с помощью программатора
- Запись данных в ЦПУ ПЛК без программатора

IC200ACC003: Устройство хранения программ EZ

Устройство хранения программ EZ (IC200ACC003) может быть использовано для хранения и обновления конфигурации, прикладной программы и данных ПЛК VersaMax. Обновление может включать Ethernet Global Data и дополнительные пользовательские параметры интерфейса Ethernet. Для начальной записи данных в устройство используются программатор и ЦПУ ПЛК. Кроме записи данных в устройство, программатор может читать данные, хранящиеся в устройстве хранения программ EZ, и сравнивать эти данные с такими же файлами, имеющимися в программаторе.

После записи данных в устройство хранения программ EZ данные могут быть записаны в несколько ЦПУ ПЛК одного типа без использования программатора.



И устройство хранения программ EZ и ПЛК не должны иметь OEM-пароля или должны иметь одинаковый OEM-пароль для выполнения обновления. Устройство хранения программ EZ не обслуживает пароли других типов.

Устройство хранения программ EZ вставляется непосредственно в порт 2 ПЛК VersaMax. Не требуются ни кабели, ни разъемы. Питание устройства поступает через порт 2. Т. к. устройство хранения программ EZ не используется во время нормальной работы, нет необходимости крепить его винтами к ПЛК. Устройство можно подключать на ходу к работающему ПЛК и отключать от него без повреждения системы.

Особенности

- 2-Мегабит последовательная флэш-память для хранения данных
- Кнопка, инициирующая передачу данных из устройства в ПЛК
- Двухцветный светодиод состояния
- Конфигурируемая защита OEM-паролем
- Совместимость со всеми моделями ЦПУ VersaMax версии 2.10 и выше.

Чтение/Запись/Сравнение данных при подключенном программаторе

При подключенном программаторе ЦПУ ПЛК может считывать, записывать или сравнивать программу, конфигурацию и таблицы в устройстве хранения программ EZ. При считывании или сравнении данных можно выбрать конфигурацию оборудования, логику и/или память данных. Однако, при записи данных в устройство хранения программ EZ должны быть записаны данные всех трех типов. Если конфигурация оборудования включает в себя Ethernet Global Data и/или дополнительные пользовательские параметры для интерфейса Ethernet, они также будут включены.

В программаторе должно использоваться программное обеспечение VersaPro версии 1.5 или выше.

ПРЕДУПРЕЖДЕНИЕ

Не пользуйтесь кнопкой на устройстве хранения программ EZ для выполнения обновления, если:

1. Загружаются логика программы, данные конфигурации и/или память данных из ПЛК в программатор.
2. Сравниваются логика программы, данные конфигурации и/или память данных в ПЛК и программаторе.

Это действие может повредить загружаемые или сравниваемые данные и привести к непредсказуемым результатам. Для восстановления нормальной работы вам следует перезапустить ПЛК.

Включение всей необходимой информации

Когда устройство хранения программ EZ обновляет ПЛК, оно перезаписывает существующую конфигурацию, программные файлы и данные в ПЛК. Поэтому, важно быть уверенным, что в устройство хранения программ EZ помещена полная информация, необходимая для надлежащей работы ПЛК. Например, если устройство хранения программ EZ содержит прикладную программу, но вместо пользовательской конфигурации оборудования оно содержит конфигурацию ПЛК по умолчанию, данные конфигурации в ПЛК будут перезаписаны. Если это произойдет, модули в системе ПЛК будут использовать свою конфигурацию по умолчанию, что может вызвать непредсказуемые последствия.

Устройство хранения программ EZ: IC200ACC003

Согласование OEM-защиты

Если ПЛК, который будет обновляться с помощью устройства хранения программ EZ, защищен OEM-паролем, убедитесь, что в конфигурации, находящейся в устройстве хранения программ EZ, установлен тот же самый OEM-пароль, в противном случае обновление будет невозможно. Если в ПЛК, который будет обновляться, не установлен OEM-пароль, в устройстве хранения программ EZ так же не должно быть пароля. Устройство не использует другие пароли системы. (Дополнительная информация о паролях приведена в главе 7, Работа ЦПУ).

Настройка таймаутов

Чтение и запись больших программ, конфигурации оборудования и ссылочных таблиц в устройство хранения программ EZ может занимать 30 секунд или более. Для предупреждения возможного ошибочного отсоединения или ошибок чтения/записи настройте таймауты запроса в конфигурации на значения 30 – 63 секунды (30000 – 63000 мс).

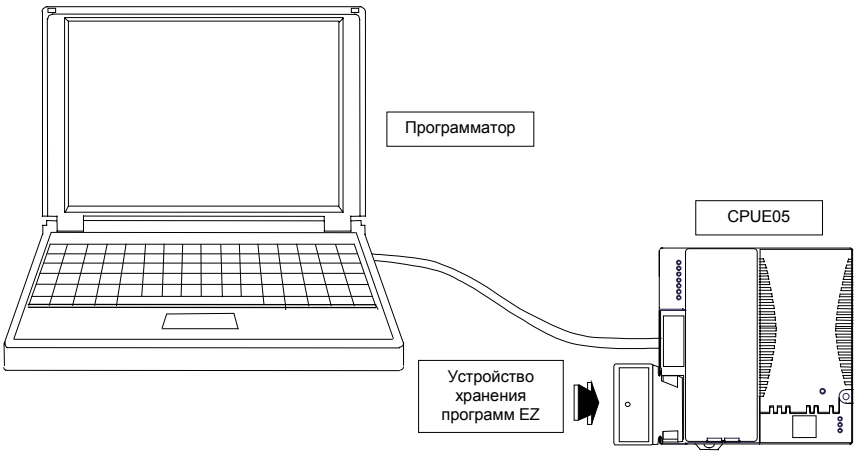
Запись данных в ОЗУ или флэш-память

Сохранение данных в устройство хранения программ EZ осуществляется точно так же, как и сохранение данных во флэш-память. При записи как во флэш-память, так и в устройство хранения программ EZ, всегда записываются все данные проекта (независимо от выбранных типов данных). Данные, сохраненные в устройстве хранения программ EZ, сравниваются тем же способом, что и данные, сохраненные во флэш-памяти. Данные так же могут быть прочитаны из устройства так же, как и из флэш-памяти.

Устройство хранения программ EZ может быть использовано для обновления данных как только в ОЗУ ПЛК, так и в ОЗУ и флэш-памяти. При конфигурировании данных, сохраняемых в устройстве хранения программ EZ, убедитесь, что указан соответствующий тип памяти. Выберите “RAM only” для обновления только ОЗУ ПЛК. Выберите “RAM & FLASH” для обновления памяти обоих типов.

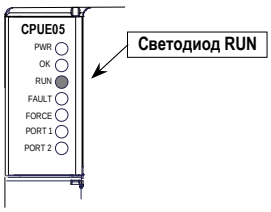
IC200ACC003: Устройство хранения программ EZ

Использование устройства хранения программ EZ с программатором



Для чтения/записи или сравнения части или всех данных, выполните следующие действия:

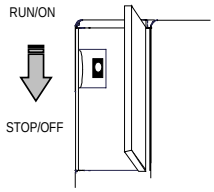
1. Вставьте устройство хранения программ EZ в порт 2 ЦПУ ПЛК VersaMax. Светодиод устройства засветится зеленым светом через 2 секунды. Задержка обусловлена временем, необходимым для подключения устройства.
2. Если ПЛК находится в режиме Run при подключении устройства хранения программ EZ, светодиод Run ПЛК мигает с частотой 1 Гц.



Это мигание указывает, что разрешено использовать переключатель режима работы Run/Stop, независимо от конфигурации переключателя.

Устройство хранения программ EZ: IC200ACC003

3. Если светодиод устройства хранения программ EZ светится зеленым светом, а светодиод Run ПЛК мигает, остановите ПЛК переводом переключателя Run/Stop из положения On/Run в положение Stop/Off.



   Если переключатель уже находится в положении Stop/Off, переведите его в Run, а затем в Stop, чтобы подтвердить изменение. После установки режима Stop No I/O светодиод Run погаснет.

   Заметьте, что для изменения режима работы ПЛК из Run в Stop или из Stop в Run при подключенном устройстве хранения программ EZ должен использоваться переключатель ПЛК Run/Stop. Если в это время программатор (компьютер) также подключен к ПЛК, программатор не может быть использован для изменения режима работы ПЛК.

4. Запустите инструментальное программное обеспечение и установите требуемое значение таймаута запроса.
5. Подключите программатор к ЦПУ ПЛК.
6. Используйте инструментальное программное обеспечение для чтения, записи или сравнения данных.

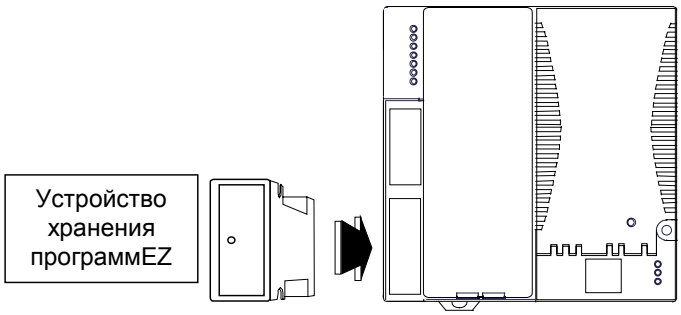
   При выполнении обновления с подключенным программатором кнопка на устройстве хранении программ EZ не используется.

Обновление ЦПУ ПЛК без подключенного программатора

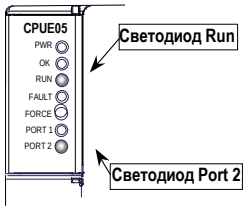
Если программа, конфигурация, таблицы, Ethernet Global Data и дополнительные пользовательские параметры (если такие есть) уже сохранены в устройстве хранения программ EZ, устройство может быть использовано для обновления одного или нескольких ЦПУ ПЛК одного типа. Все данные, хранящиеся в устройстве хранения программ EZ, будут обновлены в ЦПУ ПЛК.

Для обновления всех данных в ЦПУ ПЛК VersaMax выполните следующие действия:

1. Вставьте устройство хранения программ EZ в порт 2 ЦПУ ПЛК VersaMax.

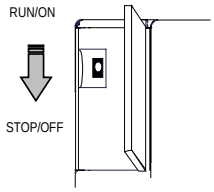


Если ПЛК находится в режиме Run при подключении устройства хранения программ EZ, светодиод Run ПЛК мигает с частотой 1 Гц. Это мигание указывает, что разрешено использовать переключатель режима работы Run/Stop, независимо от конфигурации переключателя.



Устройство хранения программ EZ: IC200ACC003

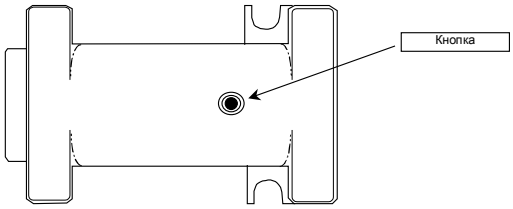
2. Если светодиод устройства хранения программ EZ светится зеленым светом, а светодиод Run ПЛК мигает, остановите ПЛК переводом переключателя Run/Stop из положения On/Run в положение Stop/Off.



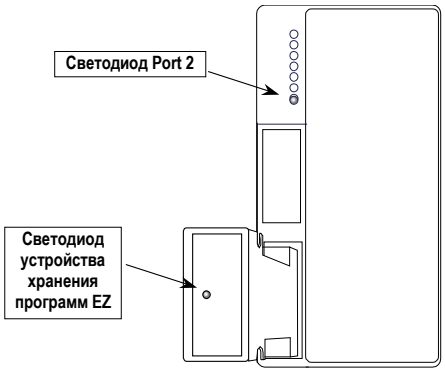
Если переключатель уже находится в положении Stop/Off, переведите его в Run, а затем в Stop, чтобы подтвердить изменение.

После установки режима Stop No I/O, светодиод Run погаснет.

3. Для начала обновления нажмите кнопку на устройстве хранения программ EZ.

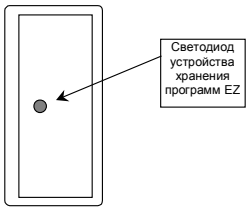


Светодиод на устройстве хранения программ EZ станет желтым, а светодиод Port 2 ПЛК начнет мигать.



4. Дождитесь завершения обновления. Чтение и запись больших программ, конфигурации оборудования и данных может занять 30 секунд или более.

Когда светодиод устройства засветится ровным зеленым светом, а светодиод Run ЦПУ начнет мигать, это будет означать успешное завершение обновления.



При установке ПЛК в режим Run (переводом переключателя Run/Stop из положения Stop/Off в положение Run/On), он немедленно начнет использовать новые данные.

Ошибки во время обновления

Если светодиод устройства хранения программ EZ мигает зеленым/желтым, и светодиод Run ЦПУ также мигает, это означает, что ошибка была обнаружена до стирания старых данных. При установке ПЛК в режим Run он продолжит использовать старые данные.

Если светодиод устройства мигает зеленым/желтым, а светодиод Run ЦПУ не светится, это означает, что ошибка произошла при передаче после того, как данные в ПЛК были стерты. Попробуйте произвести обновление опять, отключив и подключив снова устройство и нажав кнопку. Если повторное обновление также не будет выполнено, обратитесь к поставщику обновления за поддержкой.

Ошибки обновления фиксируются как ошибки чтения флэш-памяти в таблице ошибок ПЛК. Первые два байта дополнительных данных ошибки описывают ошибку.

A Эксплуатационные данные

В этом разделе представлены эксплуатационные данные ЦПУ VersaMax IC200CPU001, CPU002, CPU005 и CPUE05. Данные включают базовое время цикла, время выполнения булевых инструкций, время выполнения функциональных блоков, размеры функциональных блоков и время опроса модулей ввода-вывода.

Базовое время цикла

В следующей ниже таблице приведено базовое время цикла для программы по умолчанию в режиме Run без сконфигурированных модулей ввода-вывода и без подключений к последовательным портам.

Модель	Время (в миллисекундах)
CPU001/002	1.605
CPU005	1.039
CPUE05	1.910

Время выполнения булевых инструкций

В следующей таблице приведено время выполнения булевых инструкций:

Модель	Типовое время (в микросекундах)
CPU001/002	1.7
CPU005/E05	0.8

A

Временные параметры функциональных блоков

В следующих ниже таблицах приведено время выполнения и размеры для всех функциональных блоков, поддерживаемых ЦПУ.

Временные параметры

В следующей таблице приведены временные параметры для каждой функции. Время приращения приведено для функций, имеющих входы переменной длины (табличные функции):

Enabled	Время (в микросекундах) когда функциональный блок разрешено использовать; питание подано на функциональный блок.
Disabled	Время (в микросекундах) когда функциональный блок запрещено использовать; питание не подано на функциональный блок.
Приращение	Приращение времени (в микросекундах /единицы входа), добавляемое к базовому времени функции для каждой дополнительной единицы длины входного параметра. Применяется только для табличных функций, имеющих переменную длину входного параметра (Search, Array Moves и т. д.).

Все значения являются типовым временем выполнения. Он могут изменяться в зависимости от ошибок и условий ввода. Каждое значение включает в себя время выполнения одного контакта и обычные затраты времени, включающие соединение с программатором. (**Примечание**: времена, приведенные в предыдущей версии этого руководства не включали эти затраты.)

- Для табличных функций, приращение приведено в указанных единицах длины.
- Для функций битовых операций, микросекунды/бит.
- Для функций переноса данных, микросекунды/количество бит или слов.
- Для функций, имеющих значение приращения, умножьте приращение на (длина –1) и добавьте полученное значение к базовому времени, чтобы получить общее время инструкции.

Размеры таймеров, счетчиков, математических, тригонометрических и логарифмических функций

Размер функции – число байт, используемых для каждого применения функции в релейной логике прикладной программы.

Группа	Функция	CPU001/002		CPU005/E05		Приращение	Размер
		Enabled	Disabled	Enabled	Disabled		
Таймеры	On-Delay Timer	119	90	90	69	–	15
	Timer	110	80	81	60	–	15
	Off-Delay Timer	110	80	81	60	–	15
Счетчики	Up Counter	90	90	70	70	–	13
	Down Counter	93	90	70	70	–	13
Математика	Addition (INT)	62	12	50	10	–	13
	Addition (DINT)	60	12	50	10	–	19
	Addition (REAL)	139	12	99	10	–	17
	Subtraction (INT)	62	12	50	10	–	13
	Subtraction (DINT)	60	12	50	10	–	19
	Subtraction (REAL)	139	12	100	10	–	17
	Multiplication (INT)	70	12	50	10	–	13
	Multiplication (DINT)	99	12	50	10	–	19
	Multiplication (REAL)	155	12	108	10	–	17
	Division (INT)	80	12	60	10	–	13
	Division (DINT)	70	12	51	10	–	19
	Division (REAL)	244	12	160	10	–	17
	Modulo Division (INT)	84	12	60	10	–	13
	Modulo Division (DINT)	80	12	60	10	–	19
	Square Root (INT)	85	12	60	10	–	10
	Square Root (DINT)	126	12	70	10	–	13
	Square Root (REAL)	514	12	340	10	–	11
	Scale (INT)	112	12	78	10	–	22
	Scale (WORD)	110	12	73	10	–	22
Тригонометрия	SIN (REAL)	1432	12	945	10	–	11
	COS (REAL)	1437	12	945	10	–	11
	TAN (REAL)	2135	20	1400	20	–	11
	ASIN (REAL)	1838	12	1200	10	–	11
	ACOS (REAL)	1793	12	1200	10	–	11
	ATAN (REAL)	820	12	542	10	–	11
Логарифмы	LOG (REAL)	878	12	577	10	–	11
	LN (REAL)	821	12	542	10	–	11

Размеры экспоненциальных функций, функций преобразования радианов и функций сравнения

Размер функции – число байт, используемых для каждого применения функции в релейной логике прикладной программы.

Группа	Функция	CPU001/002		CPU005/E05		Приращение	Размер
		Enabled	Disabled	Enabled	Disabled		
Экспоненциальные	Power of e	592	12	393	10	–	11
	Power of X	365	12	249	10	–	17
Пребразования радианов	Convert RAD to DEG	328	12	214	10	–	11
	Convert DEG to RAD	106	12	70	10	–	11
Сравнения	Equal (INT)	43	12	30	10	–	10
	Equal (DINT)	50	12	37	10	–	16
	Equal (REAL)	60	12	41	10	–	14
	Not Equal (INT)	40	12	30	10	–	10
	Not Equal (DINT)	45	12	30	10	–	16
	Not Equal (REAL)	60	12	40	10	–	14
	Greater Than (INT)	40	12	30	10	–	10
	Greater Than (DINT)	45	12	30	10	–	16
	Greater Than (REAL)	60	12	40	10	–	14
	Greater Than/Equal (INT)	40	12	30	10	–	10
	Greater Than/Equal (DINT)	46	12	30	10	–	10
	Greater Than/Equal (REAL)	60	12	40	10	–	14
	Less Than (INT)	40	12	30	10	–	10
	Less Than (DINT)	46	12	30	10	–	16
	Less Than (REAL)	60	12	40	10	–	14
	Less Than/Equal (INT)	40	12	30	10	–	10
	Less Than/Equal (DINT)	46	12	30	10	–	16
	Less Than/Equal (REAL)	60	12	40	10	–	14
	Range (INT)	50	12	33	10	–	13
	Range (DINT)	55	12	40	10	–	22
	Range (WORD)	50	12	33	10	–	13

Размеры битовых операций и функций переноса данных

Размер функции – число байт, используемых для каждого применения функции в релейной логике прикладной программы.

Группа	Функция	CPU001/002		CPU005/E05		Приращение	Размер
		Enabled	Disabled	Enabled	Disabled		
Битовые Операции	Logical AND	60	12	50	10	–	13
	Logical OR	60	12	50	10	–	13
	Logical Exclusive OR	60	12	50	10	–	13
	Logical Invert, NOT	50	12	40	10	–	10
	Shift Bit Left	134	12	80	10	14.78	16
	Shift Bit Right	129	12	80	10	16.31	16
	Rotate Bit Left	110	12	70	10	18.45	16
	Rotate Bit Right	111	12	70	10	18.41	16
	Bit Position	76	12	57	10	–	13
	Bit Clear	70	12	56	10	–	13
	Bit Test	60	12	44	10	–	13
	Bit Set	70	12	56	10	–	13
	Mask Compare (WORD)	158	12	110	10	–	25
	Mask Compare (DWORD)	150	12	100	10	–	25
	Bit Sequencer	150	109	101	77	0.24	16
Перенос Данных	Move (INT)	45	12	32	10	2.83	10
	Move (BIT)	80	12	60	10	10.76	13
	Move (WORD)	46	12	32	10	2.82	10
	Move (REAL)	60	12	47	10	2.75	13
	Block Move (INT)	60	12	50	10	–	28
	Block Move (WORD)	60	12	50	10	–	28
	Block Move (REAL)	113	12	94	10	–	13
	Block Clear	100	12	83	10	4.63	11
	Shift Register (BIT)	130	12	94	10	0.45	16
	Shift Register (WORD)	120	12	100	10	2.76	16
	COMM_REQ *	175	175	120	120	–	13

* Commreq, посланный модулю быстрого счетчика.

Размеры табличных функций

Размер функции – число байт, используемых для каждого применения функции в релейной логике прикладной программы.

Группа	Функция	CPU001/002		CPU005/E05		Приращение	Размер
		Enabled	Disabled	Enabled	Disabled		
Таблицы	Array Move						
	INT	110	12	90	10	5.50	22
	DINT	100	12	80	10	2.76	22
	BIT	129	12	92	10	1.08	22
	BYTE	109	12	80	10	4.75	22
	WORD	110	12	90	10	5.50	22
	Search Equal						
	INT	90	12	70	10	6.59	19
	DINT	90	12	60	10	7.14	22
	BYTE	81	12	60	10	2.58	19
	WORD	90	12	70	10	6.59	19
	Search Not Equal						
	INT	100	12	78	10	6.66	19
	DINT	110	12	81	10	7.14	22
	BYTE	74	12	57	10	2.56	19
	WORD	100	12	78	10	6.66	19
	Search Greater Than						
	INT	100	12	80	10	6.69	19
	DINT	94	12	70	10	7.12	22
	BYTE	90	12	69	10	2.58	19
	WORD	100	12	76	10	6.69	19
	Search Greater Than/Equal						
	INT	90	12	70	10	6.79	19
	DINT	90	12	60	10	7.15	22
	BYTE	81	12	60	10	2.56	19
	WORD	90	12	70	10	6.79	19
	Search Less Than						
	INT	80	12	60	10	6.59	19
	DINT	110	12	80	10	7.13	22
	BYTE	73	12	56	10	2.58	19
	WORD	80	12	60	10	6.66	19
	Search Less Than/Equal						
	INT	80	12	60	10	6.66	19
	DINT	90	12	60	10	7.13	22
	BYTE	72	12	54	10	2.59	19
	WORD	80	12	60	10	6.66	19

Размеры функций преобразования и управляющих функций

Размер функции – число байт, используемых для каждого применения функции в релейной логике прикладной программы.

Группа	Функция	CPU001/002		CPU005/E05		Приращение	Размер
		Enabled	Disabled	Enabled	Disabled		
Преобразование	Convert INT to REAL	60	12	40	10	–	10
	Convert REAL to INT	683	12	455	10	–	13
	Convert DINT to REAL	60	12	40	10	–	13
	Convert REAL to DINT	673	12	451	10	–	13
	Convert WORD to REAL	60	12	40	10	–	10
	Convert REAL to WORD	642	12	429	10	–	13
	Convert BCD to INT	57	12	40	10	–	10
	Convert INT to BCD	167	12	120	10	–	10
	Convert BCD to REAL	70	12	50	10		10
	Truncate to INT	188	12	130	10	–	13
	Truncate to DINT	179	12	128	10	–	13
Управление	Call a Subroutine	60	12	40	10	–	7
	Do I/O *	130	12	130	10	–	13
	PID – ISA Algorithm	231	85	150	57	–	16
	PID – IND Algorithm	231	85	150	57	–	16
	Service Request						
	#6	77	12	60	10	–	10
	#7 (Read)	221	12	173	10	–	10
	#7 (Set)	2610	12	2211	10	–	10
	#14 **	169	12	139	10	–	10
	#15	100	12	72	10	–	10
	#16	110	12	80	10	–	10
	#18	346	12	251	10	–	10
	#23	377	12	361	10	–	10
	#26//30 ***	912	12	912	10	–	10
	#29	72	12	60	10	–	10
	Nested MCR/ENDMCR Combined	31	33	31	33	–	4
	Drum Sequencer	267	222	184	152	–	34

* Время DO I/O – время вывода значений на дискретный выходной модуль.

** Время Service Request #14 (Очистка таблицы ошибок) дано при условии отсутствия ошибок в таблице.

*** Время Service Request #26/30 (проверка ввода-вывода) дано при пустой конфигурации ввода-вывода и при физическом наличии модулей MDL740 (16 выходов) и MDL640 (16 входов).

Время опроса модулей ввода-вывода

В следующих таблицах приведены типовые времена опроса модулей в ПЛК VersaMax. Каждый модуль был сконфигурирован с установками по умолчанию, и внешнее питание подавалось в случае необходимости. Четыре таблицы включают:

- Модули, расположенные в главном крейте
- Модули, расположенные в локальном единичном крейте
- Модули, расположенные в удаленном крейте при множественном подключении
- Модули, расположенные в изолированном крейте

Типы дискретных модулей в таблицах времени опроса

В таблицах времени опроса дискретные модули сгруппированы по типам:

Тип модуля	**Номер модуля по каталогу, IC200:**					
Дискретный ввод Тип 1	MDL140	MDL141	MDL143	MDL144	MDL631	MDL635
	MDL640	MDL643	MDD842	MDD843	MDD844	MDD845
	MDD846	MDD847	MDD848	MDD849	MDD850	MDL930
Дискретный ввод Тип 2	MDL240	MDL241	MDL243	MDL244	MDL632	MDL636
	MDL644	MDL650	MDD840			
Дискретный вывод Тип 1	MDL329	MDL331	MDL740	MDL741	MDL743	MDD842
	MDD843	MDD844	MDD845	MDD846	MDD847	MDD848
	MDD849	MDD850				
Дискретный вывод Тип 2	MDL330	MDL742	MD744	MDL750	MDL840	MDL940
Дискретный вывод с электронной защитой от КЗ	MDL730					

Дополнительная информация по модулям ввода-вывода VersaMax приведена в документе *Модули, источники питания и шасси VersaMax. Руководство пользователя (VersaMax Modules, Power Supplies, and Carrier User's Manual) GFK-1504*.

Модули, расположенные в главном крейте ПЛК

Тип модуля	CPU005/CPUE05		CPU001/CPU002	
	Главный крейт		*Главный крейт*	
	Ввод	*Вывод*	*Ввод*	*Вывод*
Дискретный ввод Тип 1 *	95	---	158	---
Дискретный ввод Тип 2 *	117	---	189	---
Дискретный вывод Тип 1 *	---	84	---	132
Дискретный вывод Тип 2 *	---	101	---	152
Дискретный вывод с электронной защитой от КЗ	---	116	---	190
Интеллектуальный дискретный ввод 20 точек	349	---	389	---
Интеллектуальный дискретный вывод 12 точек	---	294	---	369
Аналоговый ввод 4 Канала	160	---	190	---
Аналоговый ввод 8 Канала	239	---	312	---
Аналоговый ввод 15 Каналов	377	---	526	---
Аналоговый вывод 2 Канала	---	109	---	161
Аналоговый вывод 4 Канала	---	145	---	202
Аналоговый вывод 8 Каналов	---	217	---	285
Аналоговый вывод 12 Каналов	---	289	---	367
Интеллектуальный аналоговый ввод 4 Канала	237	---	281	---
Интеллектуальный аналоговый ввод 7 Каналов	261	---	305	---
Интеллектуальный аналоговый ввод 8 Каналов	272	---	313	---
Интеллектуальный аналоговый вывод 4 Канала	---	212	---	264
Сетевой коммуникационный модуль Profibus-DP Slave	**	**	**	**
Сетевой коммуникационный модуль DeviceNet Master/Slave	**	**	**	**

* Комбинированные модули имеют значения времени опроса и по вводу, и по выводу.

** Время опроса сетевых коммуникационных модулей меняется в зависимости от сетевой конфигурации.

Модули, расположенные в единичном крейте расширения

В таблице ниже приведены времена опроса для модулей, расположенных в единичном крейте расширения с неизолированным принимающим модулем расширения (IC200ERM002). В этом типе системы ОТСУТСТВУЕТ передающий модуль расширения (IC200ETM001) в главном крейте.

Тип модуля	CPU005/CPUE05		CPU001/CPU002	
	Локальный крейт		*Локальный крейт*	
	Input	*Output*	*Input*	*Output*
Дискретный ввод Тип 1 *	127	---	191	---
Дискретный ввод Тип 2 *	179	---	262	---
Дискретный вывод Тип 1 *	---	116	---	167
Дискретный вывод Тип 2 *	---	167	---	222
Дискретный вывод с электронной защитой от КЗ	---	176	---	260
Интеллектуальный Дискретный ввод 20 точек	643	---	763	---
Интеллектуальный Дискретный вывод 12 точек	---	714	---	756
Аналоговый ввод 4 Канала	317	---	389	---
Аналоговый ввод 8 Каналов	527	---	631	---
Аналоговый ввод 15 Каналов	896	---	1054	---
Аналоговый вывод 2 Канала	---	204	---	266
Аналоговый вывод 4 Канала	---	296	---	374
Аналоговый вывод 8 Каналов	---	480	---	592
Аналоговый вывод 12 Каналов	---	664	---	809
Интеллектуальный Аналоговый ввод 4 Канала	438	---	533	---
Интеллектуальный Аналоговый ввод 7 Каналов	479	---	580	---
Интеллектуальный Аналоговый ввод 8 Каналов	493	---	596	---
Интеллектуальный Аналоговый вывод 4 Канала	---	484	---	613
Сетевой коммуникационный модуль Profibus-DP Slave	**	**	**	**
Сетевой коммуникационный модуль DeviceNet Master/Slave	**	**	**	**

* Комбинированные модули имеют значения времени опроса и по вводу, и по выводу.

** Время опроса сетевых коммуникационных модулей меняется в зависимости от сетевой конфигурации.

Модули, расположенные в удаленном крейте при множественном подключении

В таблице ниже приведены времена опроса для модулей, расположенных в крейте расширения мультикрейтовой системы, использующей только изолированные принимающие модули расширения (IC200ERM001). В главном крейте системы находится передающий модуль расширения (IC200ETM001).

Тип модуля	CPU005/CPUE05		CPU001/CPU002	
	Удаленный крейт		*Удаленный крейт*	
	Ввод	*Вывод*	*Ввод*	*Вывод*
Дискретный ввод Тип 1 *	130	---	193	---
Дискретный ввод Тип 2 *	181	---	258	---
Дискретный вывод Тип 1 *	---	118	---	167
Дискретный вывод Тип 2 *	---	165	---	223
Дискретный вывод с электронной защитой от КЗ	---	177	---	261
Интеллектуальный Дискретный ввод 20 точек	651	---	766	---
Интеллектуальный Дискретный вывод 12 точек	---	728	---	757
Аналоговый ввод 4 Канала	324	---	393	---
Аналоговый ввод 8 Каналов	541	---	646	---
Аналоговый ввод 15 Каналов	920	---	1087	---
Аналоговый вывод 2 Канала	---	206	---	267
Аналоговый вывод 4 Канала	---	300	---	377
Аналоговый вывод 8 Каналов	---	489	---	596
Аналоговый вывод 12 Каналов	---	678	---	815

Интеллектуальный Аналоговый ввод 4 Канала	442	---	535	---
Интеллектуальный Аналоговый ввод 7 Каналов	484	---	582	---
Интеллектуальный Аналоговый ввод 8 Каналов	497	---	598	---
Интеллектуальный Аналоговый вывод 4 Канала	---	490	---	615
Сетевой коммуникационный модуль Profibus-DP Slave	**	**	**	**
Сетевой коммуникационный модуль DeviceNet Master/Slave	**	**	**	**

* Комбинированные модули имеют значения времени опроса и по вводу, и по выводу.

** Время опроса сетевых коммуникационных модулей меняется в зависимости от сетевой конфигурации.

Модули, расположенные в единичном изолированном крейте расширения

В таблице ниже приведены времена опроса для модулей, расположенных в единичном крейте расширения с изолированным принимающим модулем расширения (IC200ERM001) в крейте расширения и передающим модулем расширения (IC200ETM001) в главном крейте.

Module Type	CPU005/CPUE05		CPU001/CPU002	
	Изолированный крейт		*Изолированный крейт*	
	Ввод	***Вывод***	***Ввод***	***Вывод***
Дискретный ввод Тип 1 *	466	---	524	---
Дискретный ввод Тип 2 *	869	---	913	---
Дискретный вывод Тип 1 *	---	452	---	496
Дискретный вывод Тип 2 *	---	837	---	875
Дискретный вывод с электронной защитой от КЗ	---	850	---	914
Интеллектуальный Дискретный ввод 20 точек	4050	---	4086	---
Интеллектуальный Дискретный вывод 12 точек	---	5135	---	5135
Аналоговый ввод 4 Канала	2054	---	2093	---
Аналоговый ввод 8 Каналов	3660	---	3660	---
Аналоговый ввод 15 Каналов	6471	---	6471	---
Аналоговый вывод 2 Канала	---	1221	---	1251
Аналоговый вывод 4 Канала	---	1991	---	2021
Аналоговый вывод 8 Каналов	---	3531	---	3560
Аналоговый вывод 12 Каналов	---	5071	---	5099
Интеллектуальный Аналоговый ввод 4 Канала	3155	---	3196	---
Интеллектуальный Аналоговый ввод 7 Каналов	3401	---	3444	---
Интеллектуальный Аналоговый ввод 8 Каналов	3483	---	3526	---
Интеллектуальный Аналоговый вывод 4 Канала	---	2751	---	2811
Сетевой коммуникационный модуль Profibus-DP Slave	**	**	**	**
Сетевой коммуникационный модуль DeviceNet Master/Slave	**	**	**	**

* Комбинированные модули имеют значения времени опроса и по вводу, и по выводу.

** Время опроса сетевых коммуникационных модулей меняется в зависимости от сетевой конфигурации.

Время обмена Ethernet Global Data

В зависимости от соотношения времени цикла ЦПУ и периода обмена Ethernet Global Data (EGD), данные могут передаваться каждый цикл или периодически, после некоторого количества циклов. Таким образом, время цикла будет меняться в зависимости от количества обменов, намеченных для передачи в течение цикла. Однако, в некоторый момент работы ПЛК, все обмены будут намечены для передачи в течение одного цикла. Следовательно, в самом худшем варианте, должны учитываться все обмены.

Цикл обмена Ethernet Global Data (EGD) состоит из двух частей - цикл приема и цикл передачи:

Время цикла EGD = Время цикла приема + Время цикла передачи

И цикл приема, и цикл передачи состоят из двух частей - служебное время и время передачи байта:

Время цикла = Служебное время + Время передачи байта

Служебное время

Служебное время включает установку времени для каждого обмена, который будет осуществляться в течение цикла. Служебное время меняется в зависимости от типа обмена – прием или передача и от того, откуда берется метка времени – с самого ПЛК или от NTP-сервера. При подсчете цикла учитывается служебное время для каждого обмена.

	Входящий обмен	**Исходящий обмен**
Служебное время*	80	110 (304**)

* Время в микросекундах.

** В случае, если метка времени берется с самого ПЛК, а не с NTP-сервера.

Время передачи байта

Это время, требуемое для передачи данных между модулем ЦПУ ПЛК и модулем Ethernet. В следующей таблице приведено время, требуемое для передачи одного байта.

	Входящий обмен	**Исходящий обмен**
Время передачи байта*	1 (3.6**)	1

* Время в микросекундах.

** Время передачи, если тип памяти поддерживает принудительную установку.

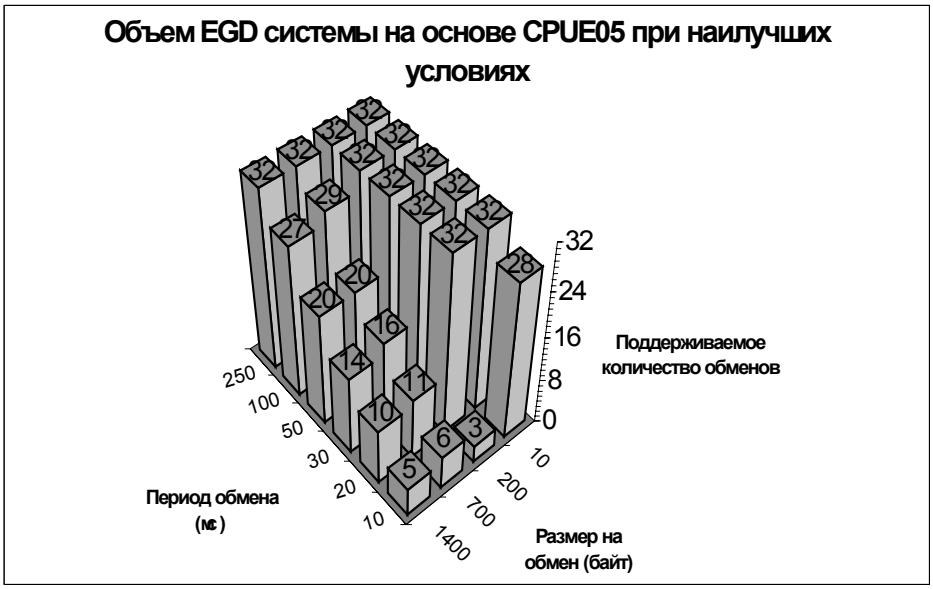
A

Поддержка больших объемов Ethernet Global Data

CPUE05 VersaMax поддерживает конфигурацию Ethernet Global Data (EGD), содержащую до 32 обменов, с периодом до 10 мс, и размером данных до 1400 байт. Однако, CPUE05 не может поддерживать конфигурацию, в которой все параметры EGD установлены в максимальное значение. Приведенная ниже диаграмма указывает максимальное количество обменов EGD, которое CPUE05 действительно поддерживает при определенном размере данных и периоде обновления данных при «наилучших условиях». Эти значения будут уменьшаться в зависимости от размера пользовательской программы, наличия другого обмена по сети Ethernet, и т. д.

Термин “наилучшие условия” указывает на следующие настройки параметров:

- Отсутствует пользовательская логика, в результате время цикла логики близко к 0.
- В системе отсутствуют модули.
- В сети Ethernet отсутствует другой обмен.
- Назначенный таймаут составляет 2 * период обновления +10мс



Указатель

V

A

Б

B

Указатель

Т

У

Ф

Указатель

Ц

Ч

Ш

Я