FANUC

GE Fanuc Automation

Программируемые устройства управления

ПЛК Series 90™-30/20/Micro
Система команд

Справочное руководство

GFK-0467L-RU *Июнь 1999*

Предупреждения и Примечания, используемые в документе

Предупреждение

Предупреждения используются в этом документе, чтобы уведомить об опасном напряжении, токе, температуре, или других факторах, которые могут нанести вред здоровью при работе с оборудованием.

Предупредительные надписи применяют, когда невнимательность может привести к повреждению здоровья и оборудования.

Внимание

Внимание обращается на действия, которые могут привести к повреждению оборудования.

Примечание

Примечания привлекают внимание к информации, которая важна для понимания принципов функционирования оборудования.

Документ содержит информацию, доступную на момент публикации. Информация может быть уточнена. Документ не описывает все возможные варианты аппаратного и программного обеспечения, а также способы их установки, функционирования и обслуживания. Описываемые функции могут присутствовать не во всех видах оборудования. Компания GE Fanuc Automation оставляет за собой право вносить изменения в документ без дополнительного уведомления.

Компания GE Fanuc Automation не дает явных либо подразумеваемых гарантий точности, полноты, достаточности, или полезности информации, содержащейся в документе. Не предоставляются также гарантии коммерческой ценности или соответствия целям пользователя.

Ниже перечислены товарные знаки, принадлежащие GE Fanuc Automation North America, Inc.

Alarm Master	Genius	PROMACRO	Series Six
CIMPLICITY	Helpmate	PowerMotion	Series Three
CIMPLICITY 90–ADS	Logicmaster	PowerTRAC	VersaMax
CIMSTAR	Modelmaster	Series 90	VersaPro
Field Control	Motion Mate	Series Five	VuMaster
Genet	ProLoop	Series One	Workmaster

Настоящее руководство описывает принципы работы, обработку сбоев и программирование с помощью Logicmaster 90™ программируемых логических контроллеров (ПЛК) Series 90™-30, Series 90-20 и Series 90 Micro. ПЛК Series 90™-30, Series 90-20 и Series 90 Micro входят в семейство программируемых логических контроллеров Series 90, выпускаемых компанией GE Fanuc Automation.

Предварительная информация об этом Руководстве

- Модель ЦП 364 (release 9.0 и выше) поддерживает соединение с сетью Ethernet через два встроенных порта. AAUI и 10BaseT порты присутствуют. Модель ЦП 364 только ЦП серий 90-30 с поддержкой Ethernet Global Data (стр. 2-39).
- Hand Held Programmer не позволяет изменять дату и время в астрономических часах, пока активирован ключ (стр. 2-14).
- Рисунки цикла ПЛК для конфигурации DSM были перемещены в главу 2.
- Различия в операциях логарифмические/экспоненциальные (стр. 6-12) и функций округления (стр. 11-11) были идентифицированы для модуля CPU352.
- Рекордер последовательности явлений (SER) был исправлен, что бы сделать понятной эту функцию. Детальная информация об этой функции приведена в приложении A. Эта функция доступна в реализации 9.0 и выше для модулей ЦП 35x и 36x (стр. 12-8).
- Новый сервисный запрос, «Skip Next Output and Input Scan» (SVCREQ #45) доступен для реализации в модулях ЦП 35x и 36x версии 9.0 и выше (стр. 12-63).
- Распечатки параметра блока для чтения и записи функций сервисного запроса «Fast Backplane Status Access» (SVCREQ #46) исправлено (стр. 12-64).
- Другие исправления приведены по мере необходимости

Содержание этого руководства

Глава 1. Введение: обзор ПЛК серий 90-30, 90-20 и 90 Micro и также система команд для серий ПЛК 90-30/20/Micro.

Глава 2. Принципы работы системы: описывает некоторые системные операции ПЛК серий 90-30, 90-20 или 90 Micro. Это включает в себя последовательность системного цикла прогона, последовательностей включения и выключения питания, часы и таймеры, вопросы безопасности, ввод/вывод, и описания сбоев. Также включена информация о базовых понятиях программирования релейной логики.

Глава 3. Сбои и их исправление: обеспечивает информацией о решениях проблем для ПЛК серий 90-30, 90-20 или Micro. Раскрывает описания сбоев в таблице сбоев ПЛК и категории сбоев в таблице B/B.

Главы 4—12. Система команд ПЛК серий 90-30/20/Micro: описывает инструкции программирования, располагаемые для ПЛК серий 90-30, 90-20 и 90 Micro. Эти главы соответствуют группам функций.

Приложение A. Время выполнения инструкций: перечисляет размер памяти в байтах и время выполнения в миллисекундах для каждой инструкции программирования. Размер памяти это число байт, необходимых функции релейной логики для корректной работы.

Приложение B. Интерпретация таблиц сбоев: описывает, как интерпретировать формат структуры сообщений, при чтении таблиц сбоев, используя ПО Logicmaster 90-30/20/Micro.

Приложение C. Мнемоника инструкций: список мнемоник, которые можно использовать для отображения инструкций, для поиска или редактирования.

Приложение D. Быстрые клавиши: список быстрых клавиш, используемых в ПО для программирования Logicmaster 90-30/20/Micro.

Приложение E. Числа с плавающей точкой: описывает специальные соображения для использования математических операций над числами с плавающей точкой

Документы, связанные с этим Руководством

Logicmaster™ 90 среда программирования ПЛК серий 90™-30/20/Micro. Руководство пользователя (GFK-0466).

Logicmaster™ 90 для ПЛК Series 90-30 и 90-20. Важная информация о продукте (GFK-0468).

Программируемый контроллер Series 90™-30. Руководство по установке (GFK-0356).

Программируемый контроллер Series 90™-20 Руководство по установке (GFK-0551).

Модули B/B Series 90™-30. Техническое руководство (GFK-0898).

Модуль программируемого сопроцессора Series 90™. Руководство пользователя (GFK-0255).

Series 90™ PCM Development Software (PCOP) Руководство пользователя (GFK-0487).

CIMPLICITY™ 90-ADS Alphanumeric Display System Руководство пользователя (GFK-0499).

CIMPLICITY™ 90-ADS Alphanumeric Display System Reference Manual (GFK-0641).

Alphanumeric Display Coprocessor Module Data Sheet (GFK-0521).

Переносной ручной программатор для ПЛК серий 90™-30 и 90-20. Руководство пользователя (GFK-0402).

Power Mate APM для ПЛК Series 90™-30 — Стандартный режим. Руководство пользователя (GFK-0840).

Power Mate APM для ПЛК Series 90™-30 — Режим слежения. Руководство пользователя (GFK-0781).

Motion Mate™ DSM302 для ПЛК Series 90™-30 Руководство пользователя (GFK-1464)

Высокоскоростной счетчик Series 90™-30. Руководство пользователя (GFK-0293).

Коммуникационный модуль Genius Series 90™-30. Руководство пользователя (GFK-0412).

Коммуникационный модуль Genius. Data Sheet (GFK-0272).

Контроллер сети Genius™ Series 90™-30. Руководство пользователя (GFK-1034).

Контроллер сети FIP Series 90™-70 Руководство пользователя (GFK-1038).

Сканер удаленного В/В по сети FIP Series 90™-30. Руководство пользователя (GFK-1037).

Система распределенного В/В и управления Field Control™. Модуль сетевого интерфейса Genius™. Руководство пользователя (GFK-0825).

Программируемый логический контроллер Series 90™ Micro. Руководство пользователя (GFK-1065).

Последовательный интерфейс связи для ПЛК Series 90™. Руководство пользователя (GFK-0582).

Оглавление

Оглавление

Глава 1 Введение

Программируемые логические контроллеры (далее ПЛК) Series 90-30, 90-20, и Micro принадлежат семейству Series 90 контроллеров фирмы GE Fanuc. Они просты в установке и настройке, имеют улучшенные возможности для программирования и полностью совместимы с серией контроллеров 90-70.

Контроллер 90-20 обеспечивает наиболее эффективное решение для задач, использующих небольшое количество точек ввода/вывода. Основные цели применения контроллеров 90-20 следующие:

- Предоставление таких ПЛК, которые обеспечивают простоту использования, установки, обновления и их поддержки.
- Обеспечение максимальной совместимости с контроллерами семейства Series 90.
- Обеспечение возможности простого их внедрения, посредством использования стандартного коммуникационного аппаратного обеспечения и стандартных же протоколов.

Контроллеры Series 90-Micro также обеспечивают наиболее эффективную базу для систем с небольшим количеством каналов ввода/вывода. Основные цели применения контроллеров Micro такие же, как и у контроллеров 90-20, но есть некоторые дополнения:

- Контроллеры Micro имеют, собранные в одном моноблоке, модули ЦП, питания, входов и выходов.
- Большинство моделей имеют высокоскоростной счетчик.
- Так как модули ЦП, питания, входов и выходов собраны в одном устройстве, то процедура конфигурации намного упрощается.

Структура программного обеспечения для модулей ЦП 341 и младше ПЛК 90-30 и ПЛК 90-20 основывается на использовании архитектуры управления памятью и выполнением приоритетов, как в микропроцессоре 80188. Модули ЦП серий 35x и 36x ПЛК 90-30 используют микропроцессор 80386EX. Контроллеры 90 Micro используют микропроцессор H8. Этот процессор поддерживает, как выполнение программы, так и базовые служебные операции, такие как операции диагностики, сканирования входов/выходов, и обработку тревожных сообщений. Системное программное обеспечение ПЛК также имеет процедуры для взаимодействия с программатором. Эти процедуры предназначены для загрузки и выгрузки программы контроллера, получения информации о состоянии контроллера и управления контроллером.

В контроллерах Series 90-30, прикладная программа, управляющая окончанием процесса выполняемого на ПЛК, управляется Сопроцессором Последовательности Инструкций (Instruction Sequencer Coprocessor ISCP). Этот сопроцессор выполнен на аппаратном уровне в

ЦП моделей 313 и старше, и на программном уровне в ЦП моделях 311, и в контроллерах 90-Micro. Микропроцессор 80188 и ISCP могут работать одновременно, позволяя микропроцессору выполнять обслуживание связей, пока ISCP выполняет прикладную программу; тем не менее, микропроцессор обязан сам выполнять операции отличные от булевых.

При возникновении условий, при которых нарушается работоспособность и производительность системы, регистрируются сбои в ПЛК серий 90-30, 90-20, и Micro. Такие условия могут лишить ПЛК возможности управлять агрегатом или процессом. В остальных случаях регистрируется предупреждение, например, такое как, сигнал о разряде батареи, указывающий на то, что напряжение в батарее, защищающей память, слишком мало и батарея должна быть заменена.

Регистрируемые сбои обрабатываются специальной функцией ПО, которая записывает все сбои либо в таблицу сбоев ПЛК, либо в таблицу сбоев В/В. (Для ЦП модели 331 или старше возможна также регистрация времени происхождения сбоя) Таблицы сбоев могут быть отображены посредством программного обеспечения для программирования ПЛК на экранах в ПО Logicmaster 90-30/20/Micro, используя функции управления и статуса.

Примечание

Операции с плавающей точкой поддерживаются ***только*** в ЦП моделях 35x и 36x Версии 9 или более поздних, и во всех версиях ЦП модели 352.

ЦП модели 364 (Версии 9.10 или более поздней) для ПЛК Series 90-30 поддерживает Ethernet Global Data (EGD).

Примечание

Дополнительную информацию смотрите в приложениях расположенных в конце данного руководства.

- Приложение A содержит информацию относительно размеров памяти в байтах и времени выполнения каждой инструкции в мкс.
- Приложение B описывает, каким образом необходимо понимать текстовые сообщения появляющиеся в таблицах сбоев ПЛК и В/В.
- Приложение C содержит список мнемонических кодов инструкций для осуществления поиска или редактирования программы.
- Приложение D содержит список горячих клавиш, используемых в ПО Logicmaster 90-30/20/Micro.
- Приложение E описывает использование математических операций с плавающей точкой.

Глава 2

Принципы работы системы

Настоящая глава подробно описывает принципы функционирования программируемых логических контроллеров (ПЛК) Series 90-30, 90-20, и Micro.

Содержание главы:

Раздел 1: Общие сведения о цикле ПЛК

Прикладная программа в ПЛК Series 90-30, 90-20, и Micro выполняется циклически до тех пор, пока не будет остановлена командой с программатора или другого устройства. Последовательность операций, необходимых для однократного выполнения программы, называется *цикл*. Помимо выполнения прикладной программы, во время цикла выполняется получение данных от устройств ввода, пересылка данных на устройства вывода, служебные операции, связь с программатором и с другими устройствами.

Обычно ПЛК работает в режиме **СТАНДАРТНОГО ЦИКЛА** (**STANDARD PROGRAM SWEEP**). Возможны также следующие режимы работы:

STOP WITH I/O DISABLED – СТОП БЕЗ В/В

STOP WITH I/O ENABLED - СТОП С В/В

CONSTANT SWEEP - РЕЖИМ ПОСТОЯННОГО ВРЕМЕНИ ЦИКЛА

Все эти режимы, описанные в настоящей главе, переключаются по внешним событиям или из приложений. ПЛК устанавливает режим работы в начале каждого цикла.

Стандартный цикл ПЛК

Режим **СТАНДАРТНОГО ЦИКЛА** – обычный режим работы ПЛК. Центральный процессор (ЦП) выполняет прикладную программу, сканирование В/В, поддерживает связь с внешними устройствами и т.д. Это происходит в повторяющемся цикле – цикле ЦП. Выполнение стандартного цикла состоит из семи частей:

1. Служебные операции в начале цикла
2. Сканирование (чтение) входов
3. Выполнение прикладной программы
4. Обновление выходов
5. Связь с программатором
6. Связь с другими устройствами
7. Диагностика

Все эти шаги выполняются в каждом цикле. Хотя окно связи с программатором есть в каждом цикле, обмен данными осуществляется только, если программатор подключен и прислал соответствующий запрос, то есть, если запроса нет, то ЦП переходит к следующему шагу цикла. Ниже на рисунке приведена схема стандартного цикла ПЛК.

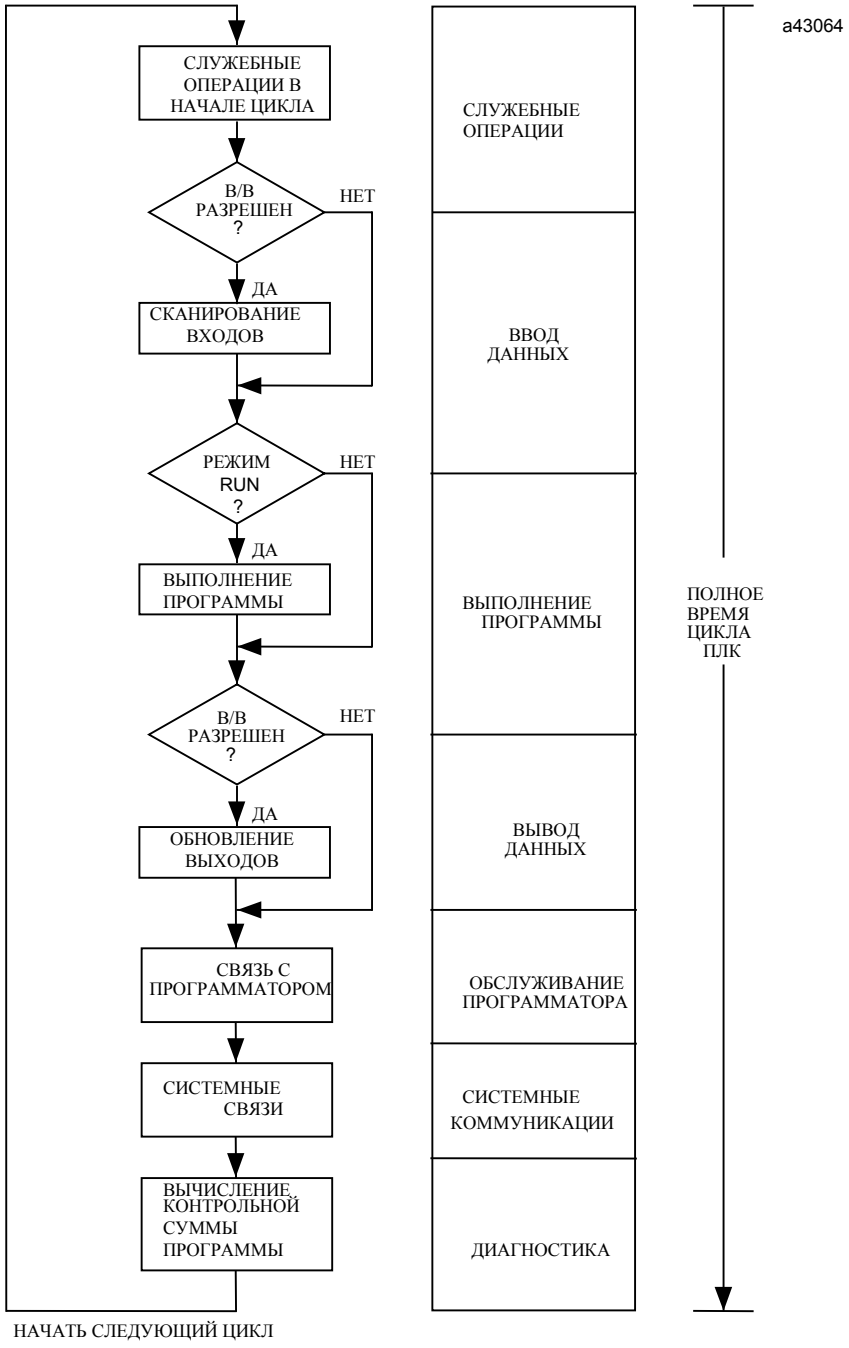


Рисунок 2-1. Цикл ПЛК

Выше показано, что цикл ПЛК состоит из нескольких этапов. Полное время цикла ПЛК распределено по этим этапам, как показано в следующей таблице.

Таблица 2-1. Распределение времени цикла

Элемент цикла	Описание		Распределение времени (мс) [4]
			Для ЦП 351, 352, 350, и серии 36x (время для ЦП 350 и ЦП серии 36x совпадает)
Служебные операции	• Расчет времени цикла. • Назначение старта следующего цикла. • Установка режима следующего цикла. • Обновление таблицы сбоев. • Сброс сторожевого таймера.		0.279
Ввод данных	Получение данных от модулей ввода и дополнительных модулей.		Подробнее см. таблицу 2-2.
Выполнение программы	Решение логики пользователя.		Время выполнения зависит от длины программы и типа используемых в программе инструкций. Время выполнения инструкций приведено в приложении A.
Вывод данных	Отправка данных модулям вывода и дополнительным модулям.		Подробнее см. таблицу 2-2.
Обслуживание внешних устройств	Обработка сервисных запросов от устройств программирования и интеллектуальных модулей. [1]	HHP	0.334
		LM-90	0.517
		PCM [2]	0.482
Реконфигурация	Контроль пустых слотов и слотов со сбойными модулями		0.319[6]
Диагностика	Проверка целостности программы пользователя		0.010 на каждое проверяемое слово [3, 7]

1. Время обработки запросов внешних устройств зависит от режима окна связи. Если установлен режим **`LIMITED`**, то на обработку запросов отводится не более 8 мс в ЦП моделей 311, 313, 323, и 331; и не более 6 мс в ЦП моделей 340 и выше. Если установлен режим **`RUN-TO-COMPLETION`**, на обработку может быть затрачено до 50 мс, в зависимости от количества одновременно поступивших запросов.
2. Данные получены для модуля PCM, который физически установлен, но не сконфигурирован и не загружен прикладной задачей.
3. Количество проверяемых контрольным суммированием слов программы может быть изменено функцией SVCREQ.
4. Данные получены при отсутствии прикладной программы и при конфигурации по умолчанию. Модуль ЦП устанавливался на пустую 10-слотовую базовую плату без подключения стоек расширения. Также, подразумевается отсутствие активных периодических подпрограмм-прерываний. При наличии таковых, время увеличивается.
5. Для ПЛК Series Micro время на ввод данных может быть определено: 0.365 мс (fixed scan) + 0.036 мс (filter time) x (полное время цикла)/0.5 мс.
6. ПЛК Series Micro имеет фиксированное число каналов B/B, поэтому реконфигурация не требуется.
7. ПЛК Series 90 Micro хранит программу в памяти Flash, поэтому проверка целостности не требуется.

Таблица 2-2. Затраты времени на обмен данными с модулями В/В для ЦП моделей 35x и 36x (в миллисекундах)

Тип модуля		ЦП серий 35x и 36x		
		Главная стойка	Стойка расширения	Удаленная стойка
Ввод дискретный 8 каналов		0.030	0.055	0.206
Ввод дискретный 16 каналов		0.030	0.055	0.206
Ввод дискретный 32 канала		0.043	0.073	0.269
Вывод дискретный 8 каналов		0.030	0.053	0.197
Вывод дискретный 16 каналов		0.030	0.053	0.197
Вывод дискретный 32 канала		0.042	0.070	0.259
Комбинированный дискретный В/В		0.060	0.112	0.405
Ввод аналоговый 4 канала		0.075	0.105	0.396
Вывод аналоговый 2 канала		0.058	0.114	0.402
Ввод аналоговый 16 каналов (ток или напряжение)		0.978	1.446	3.999
Вывод аналоговый 8 каналов		1.274	1.988	4.472
Комбинированный аналоговый В/В		1.220	1.999	4.338
Высокоскоростной счетчик		1.381	2.106	5.221
Процессор В/В		1.574	2.402	6.388
Интерфейс Ethernet (не подключен)		0.038	0.041	0.053
Модуль Power Mate APM (1 ось)		1.527	2.581	6.388
Модуль Power Mate APM (2 оси)		1.807	2.864	7.805
DSM 302 *	40 AI, 6 AQ	2.143	3.315	9.527
	50 AI, 9 AQ	2.427	3.732	11.092
	64 AI, 12 AQ	2.864	4.317	13.138
GCM	нет устройств	0.911	1.637	5.020
	8 64-канальных устройств	8.826	16.932	21.179
GCM+	нет устройств	0.567	0.866	1.830
	32 64-канальных устройства	1.714	2.514	5.783
GBC	нет устройств	0.798	1.202	2.540
	32 64-канальных устройства	18.382	25.377	70.777
PCM 311	не сконфигурирован или без прикладной задачи	0.476	-	-
	чтение 128 регистров (%R) с макс. скоростью	0.485	-	-
ADC (без задачи)		0.476	-	-
Модуль I/O Link Master	нет устройств	0.569	0.865	1.932
	16 64-канальных устройств	4.948	7.003	19.908
Модуль I/O Link Slave	32 канала	0.087	0.146	0.553
	64 канала	0.154	0.213	0.789

* В приложениях, где обслуживание модулей DSM приводит к резкому уменьшению производительности системы, Вам может потребоваться применение инструкции Do I/O и служебных запросов Suspend I/O и Fast Backplane Status Access для порционного обмена данными с модулем DSM. Подробнее см. руководство GFK1464 *Motion Mate DSM302 для ПЛК Series 90-30. Руководство пользователя*.

Таблица 2-3. Затраты времени на обмен данными с модулями В/В для ЦП моделей до 341 (в миллисекундах)

Тип модуля		Модель ЦП						
		311/313	331			340/341		
			Главная стойка	Стойка расширения	Удаленная стойка	Главная стойка	Стойка расширения	Удаленная стойка
Ввод дискретный 8 каналов		0.076	0.054	0.095	0.255	0.048	0.089	0.249
Ввод дискретный 16 каналов		0.075	0.055	0.097	0.257	0.048	0.091	0.250
Ввод дискретный 32 канала		0.094	0.094	0.126	0.335	0.073	0.115	0.321
Вывод дискретный 8 каналов		0.084	0.059	0.097	0.252	0.053	0.090	0.246
Вывод дискретный 16 каналов		0.083	0.061	0.097	0.253	0.054	0.090	0.248
Вывод дискретный 32 канала		0.109	0.075	0.129	0.333	0.079	0.114	0.320
Ввод/вывод 8 каналов		0.165	0.141	0.218	0.529	0.098	0.176	0.489
Ввод аналоговый 4 канала		0.151	0.132	0.183	0.490	0.117	0.160	0.462
Вывод аналоговый 2 канала		0.161	0.138	0.182	0.428	0.099	0.148	0.392
Высокоскоростной счетчик		2.070	2.190	2.868	5.587	1.580	2.175	4.897
Модуль Power Mate APM (1 ось)		2.330	2.460	3.175	6.647	1.750	2.506	5.899
Модуль Power Mate APM (2 оси)		3.181	3.647	4.497	9.303	2.154	3.097	7.729
DSM 302	40 AI, 6 AQ	3.613	4.081	5.239	11.430	2.552	3.648	9.697
	50 AI, 9 AQ	4.127	4.611	5.899	13.310	2.911	4.170	11.406
	64 AI, 12 AQ	4.715	5.276	6.759	15.747	3.354	4.840	13.615
GCM	нет устройств	0.041	0.054	0.063	0.128	0.038	0.048	0.085
	8 64-канальных устройств	11.420	11.570	13.247	21.288	9.536	10.648	19.485
GCM+	нет устройств	0.887	0.967	1.164	1.920	0.666	0.901	1.626
	32 64-канальных устройства	4.120	6.250	8.529	21.352	5.043	7.146	20.052
PCM 311	не сконфигурирован или без прикладной задачи	-	3.350	-	-	1.684	-	-
	чтение 128 регистров (%R) с макс. скоростью	-	4.900	-	-	2.052	-	-
ADC 311		-	3.340	-	-	1.678	-	-
Ввод аналоговый 16 каналов (ток или напряжение)		1.370	1.450	1.937	4.186	1.092	1.570	3.796
Модуль I/O Link Master	нет устройств	1.910	2.030	1.169	1.925	0.678	0.904	1.628
	16 64-канальных устройств	6.020	6.170	8.399	21.291	4.992	6.985	20.010
Модуль I/O Link Slave	32 канала	0.206	0.222	0.289	0.689	0.146	0.226	0.636
	64 канала	0.331	0.350	0.409	1.009	0.244	0.321	0.926

* В приложениях, где обслуживание модулей DSM приводит к резкому уменьшению производительности системы, Вам может потребоваться применение инструкции Do I/O и служебных запросов Suspend I/O и Fast Backplane Status Access для порционного обмена данными с модулем DSM. Подробнее см. руководство GFK1464 *Motion Mate DSM302 для ПЛК Series 90-30. Руководство пользователя*.

Расчет времени цикла

В таблице 2-1 перечислены семь составляющих полного времени цикла ПЛК. Время цикла состоит из постоянных частей (служебные операции и диагностика) и переменных частей. Длительность переменных частей изменяется в соответствии с конфигурацией В/В, размером программы пользователя и типом программирующего устройства, подключенного к ПЛК.

Пример расчета времени цикла

Ниже в табличной форме приведен пример расчета времени цикла ПЛК Series 90-30 с ЦП модели 331.

Данные о модулях В/В и программе ПЛК, использованные в расчете:

- Модули ввода: пять 16-канальных модулей Series 90-30.
- Модули вывода: четыре 16-канальных модуля Series 90-30.
- Прикладная программа: 1200-шаговая программа, состоящая из 700 дискретных команд (LD, AND, OR, и т.д.), 300 обмоток (OUT, OUTM, и т.д.), и 200 математических функций (ADD, SUB, и т.д.).

Служебные операции

На стадии служебных операций выполняются все действия, необходимые для начала выполнения цикла ПЛК. Если ПЛК находится в режиме **CONSTANT SWEEP** , то старт цикла задерживается до истечения установленного времени. Если же это время уже истекло, то бит OV_SWP %SA0002 устанавливается в 1, и цикл начинается без задержки. Затем обновляются значения таймера (секунды, десятые и сотые доли секунды) путем вычисления разности между временем старта предыдущего цикла и текущим временем. Фактическое время старта цикла фиксируется с точностью до 100 мкс. Каждый таймер имеет специальное поле, хранящее количество 100-микросекундных шагов с момента последнего обновления таймера.

Сканирование входов

Сканирование входов ПЛК предшествует стадии выполнения прикладной программы. На этой стадии цикла сканируются все модули ввода и их данные сохраняются в соответствующих областях памяти типа %I (дискретные входы) и %AI(аналоговые входы). Данные, получаемые модулями Genius Communications Module, Enhanced Genius Communications Module, или Genius Bus Controller сохраняются в памяти %G.

Модули опрашиваются в порядке возрастания назначенных им адресов памяти, причем, сначала сканируются модуль Genius Communications Module, затем дискретные модули ввода, и, наконец, аналоговые модули ввода.

Если ЦП находится в режиме **STOP** и отключен режим сканирования В/В, то стадия ввода данных пропускается.

Выполнение прикладной программы

На этой стадии выполняется прикладная программа пользователя. Выполнение программы всегда начинается с первой инструкции сразу же по завершении сканирования входов. В результате выполнения программы обновляется память выходных данных ПЛК. Выполнение программы завершается на инструкции END (инструкция END невидима, если Вы не используете ручной переносной программатор ПЛК).

Программа выполняется микропроцессорами ISCP и 80C188. В моделях ЦП 313 и выше процессор ISCP выполняет булевы операции, процессор 80C188 или 80386EX выполняют таймеры, счетчики и функциональные блоки. В ЦП модели 311 и ЦП Series 90-20, процессор 80C188 выполняет все виды инструкций. В ПЛК Series Micro все виды инструкций выполняет процессор H8.

Время выполнения всех видов инструкций центральным процессором ПЛК приведено в приложении A.

Сканирование выходов

Стадия вывода данных (обновление выходов) выполняется незамедлительно после выполнения прикладной программы. Выходы обновляются с использованием данных из областей памяти %Q (для дискретных выходов) и %AQ (для аналоговых выходов). Если в системе сконфигурированы сетевые модули Genius, публикующие данные, то тогда данные из памяти %G отсылаются модулям GCM, GCM+, или GBC. В ПЛК Series 90-20 и Micro вывод данных включает в себя только обновление дискретных выходов.

При выполнении вывода данных все выходные модули сканируются в порядке возрастания назначенных им адресов памяти.

Если ЦП находится в режиме **STOP** и отключен режим сканирования В/В, то стадия ввода данных пропускается. Стадия вывода данных завершается, если все выходные данные отправлены соответствующим модулям вывода.

Вычисление контрольной суммы программы

При вычислении контрольной суммы на каждом цикле обрабатывается память прикладной программы. Так как контрольное суммирование всей программы целиком может занимать много времени, Вы можете задать количество слов (от 0 до 32), проверяемых центральным процессором ПЛК.

Если вычисленная сумма не совпала с контрольным значением, устанавливается флаг сбоя по контрольной сумме. В таблицу сбоев ПЛК помещается соответствующая запись, ПЛК переводится в режим **STOP**. Сбой по контрольной сумме не влияет на выполнение окна связи с программатором. По умолчанию ЦП вычисляет контрольную сумму 8 слов памяти программ.

Окно связи с программатором

Эта часть цикла предназначена для связи с программатором. Если к ПЛК подключен программатор, ЦП поддерживает связь с ним. Окно связи с программатором не выполняется, если программатор не подключен или не сконфигурирован канал связи с ним. В течение одного цикла обслуживается только один канал.

Поддерживается как ручной программатор, так и другие виды программаторов, которые могут быть подключены к последовательному порту и используют протокол SNP (Series Ninety Protocol). Возможна также связь с программатором, подключенным через интеллектуальные дополнительные модули связи.

По умолчанию длительность окна связи ограничена. ЦП выполняет не более одного запроса от программатора за один цикл. Если программатор генерирует запрос, требующий более 6 (или 8 в зависимости от ЦП – см. примечание) миллисекунд на обработку, то он обрабатывается за несколько циклов, так, чтобы длительность окна связи в каждом цикле не превышала 6 (или 8 в зависимости от ЦП – см. примечание) миллисекунд.

Примечание

Лимит времени для окна связи с программатором составляет 6 мс для ЦП модели 340 и выше и 8 мс для ЦП моделей 311, 313, 323, и 331.

На следующем рисунке приведена блок-схема выполнения окна связи с программатором.

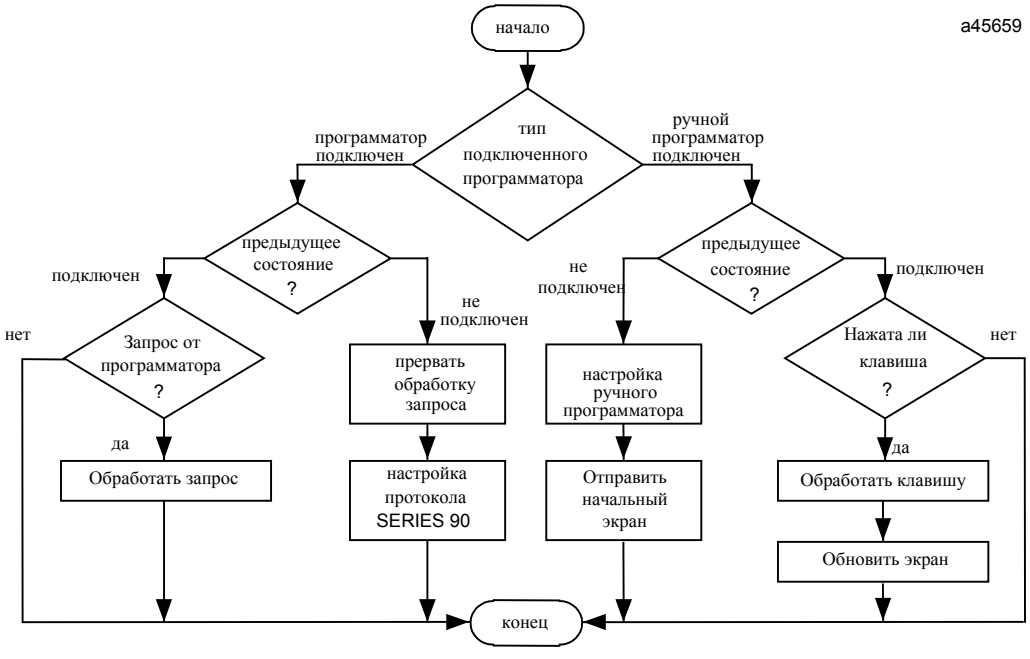


Рисунок 2-2. Блок-схема выполнения окна связи с программатором

Окно связи с системой

В этой части цикла обрабатываются коммуникационные запросы от интеллектуальных дополнительных модулей, таких как PCM или DSM. Запросы обрабатываются в порядке их поступления. Однако, так как опрос интеллектуальных дополнительных модулей производится последовательно, то при обработке все они имеют одинаковый приоритет.

По умолчанию в режиме **Run-to-Completion**, длительность окна системной связи ограничена 50 мс. Если интеллектуальный модуль делает запрос, обработка которого требует более 50 мс, то этот запрос обрабатывается за несколько циклов так, чтобы длительность окна системной связи в каждом цикле не превышала 50 мс.

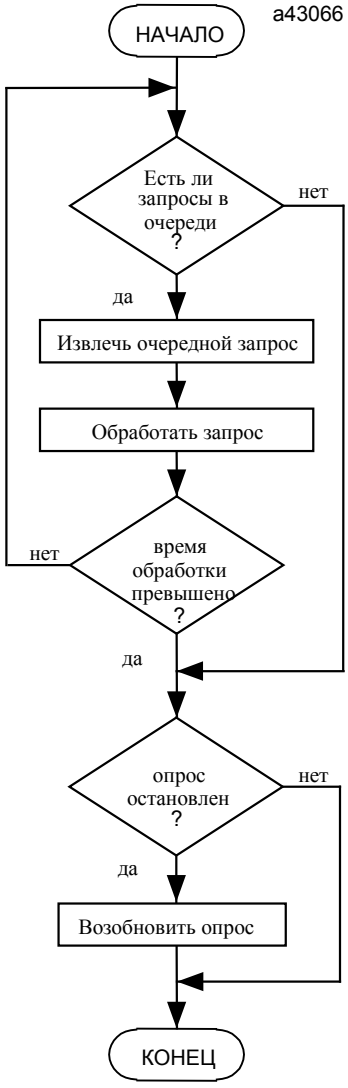


Рисунок 2-3. Блок-схема выполнения окна системной связи

Связь модуля PCM с ЦП ПЛК (модели 331 и старше)

Интеллектуальный дополнительный модуль (ИДМ), например PCM, не может прерывать работу ЦП. ЦП должен опрашивать каждый такой модуль. Опрос производится асинхронно в фоновом режиме во время цикла (см. блок-схему ниже).

При опросе интеллектуальный дополнительный модуль посылает ЦП служебный запрос, который помещается в очередь на обработку в окне системной связи.

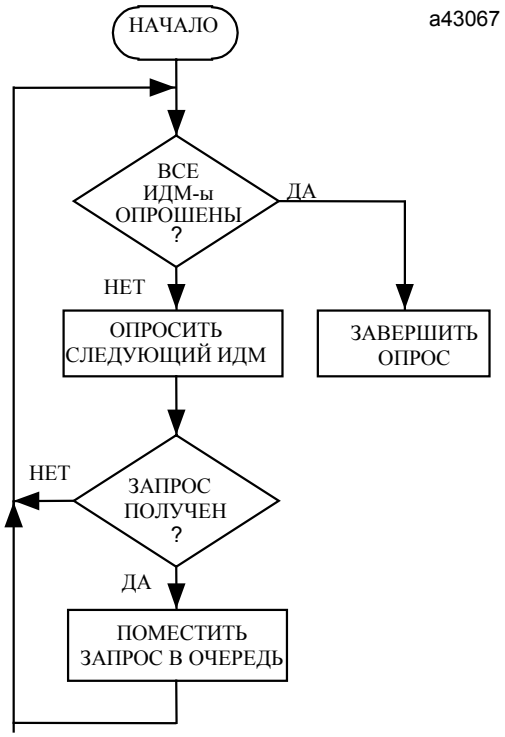


Рисунок 2-4. Связь модуля PCM с центральным процессором ПЛК

Связь модуля DSM с ЦП ПЛК

DSM302 – дополнительный интеллектуальный модуль, работающий асинхронно с ЦП контроллера 90-30. Обмен данными между модулем DSM302 и модулем ЦП производится автоматически.

Модуль DSM может быть сконфигурирован на три различных объема данных типа %AI и %AQ. Процессору требуется время фонового чтения/записи этих данных. Таблица 2-2 содержит время обмена данными с модулем DSM302 в трех конфигурациях. Для дополнительной информации о времени обмена данными смотри руководство GFK-1464 «*Motion Mate DSM302 для ПЛК Series 90-30. Руководство пользователя*».

Варианты режимов цикла ПЛК

Помимо стандартного режима цикла ПЛК, могут быть установлены дополнительные режимы и/или разновидности режимов цикла. Эти режимы, описываемые в следующих параграфах, можно контролировать и/или переключать из программного обеспечения.

Режим постоянного времени цикла

В стандартном режиме время начала следующего цикла определяется временем, затраченным на выполнение предыдущего. Однако, возможен режим постоянного времени цикла (**CONSTANT SWEEP TIME**), при котором на выполнение каждого цикла отводится одинаковое время. При включении этого режима он становится режимом по умолчанию, то есть вступает в силу каждый раз, когда ПЛК переходит из режима **STOP** в **RUN**. Допустимые значения заданного времени цикла – от 5 до 200 мс (для ЦП моделей 35x и 36x – до 500 мс). По умолчанию значение постоянного времени цикла 100 мс.

Так как длительность отдельных стадий цикла может изменяться, время постоянного цикла должно быть задано как минимум на 10 мс больше, чем время, которое затрачивает ПЛК на цикл в стандартном режиме. Это позволяет исключить внешние причины превышения времени цикла.

Используйте режим постоянного времени цикла, если согласно алгоритму управления каналы В/В должны опрашиваться с постоянной частотой. Одной из причин использования режима постоянного времени цикла может быть потребность обеспечить обновление каналов В/В через равные интервалы времени. Другой причиной может быть необходимость обеспечить задержку времени между обновлением выходов одного цикла и обновлением входов в следующем цикле.

Если время цикла истекает до завершения всех его стадий, то завершается весь цикл, включая окна связи. При этом в начале следующего цикла в таблицу сбоев ПЛК добавляется запись о превышении времени цикла.

Примечание

В отличие от режима Active Constant Sweep, применяемого только в состоянии **RUN**, режим Configured Constant Sweep может быть применен, только если ПЛК находится в состоянии **STOP**. Для того чтобы он вступил в силу, необходимо в программаторе выполнить операцию «Сохранить конфигурацию из программатора в ПЛК». После этого режим постоянного цикла становится режимом цикла по умолчанию.

Цикл ПЛК в режиме STOP

Когда ПЛК находится в режиме **STOP**, прикладная программа не выполняется. Связь с программатором и интеллектуальными модулями не прекращаются. Также в режиме **STOP** продолжается опрос сбойных модулей и реконфигурация модулей. Кроме того, операционная система ПЛК использует большие величины кванта времени, чем в режиме **RUN** (обычно около 50 мс на каждую стадию цикла). Вы можете также включить или отключить сканиро-

вание В/В. Сканирование В/В выполняется, если параметр ***IOScan-Stop*** в окне дополнительных настроек ЦП установлен в **YES**.

Режимы окна связи

По умолчанию для окна связи с программатором установлен «ограниченный» режим. Это значит, что запросы, требующие более 6 мс на обработку, выполняются в течение нескольких циклов ПЛК, чтобы ни в одном из циклов окно не длилось более 6 мс. Для ЦП моделей 313, 323 и 331 ограничение размера окна в режиме **RUN** составляет 12 мс. Текущий режим окна может быть изменен в окне “Sweep Control” программы Logicmaster. Инструкции, изменяющие текущий режим окна связи приведены в руководстве *Logicmaster 90™ среда программирования ПЛК Series 90™-30/20/Micro Руководство пользователя* (GFK-0466).

Примечание

Если режим окна изменен на «Ограниченный», то такие дополнительные модули как PCM и GBC меньше влияют на полное время цикла ПЛК, однако, обработка из запросов при этом происходит медленнее.

Ключ на ЦП моделей 35x и 36x: Смена режима и защита Flash

Каждый модуль ЦП серий 35x и 36x имеет на передней панели переключатель под ключ, позволяющий защитить Flash–память от перезаписи. Если ключ установлен в положение **ON**/**RUN**, то содержимое Flash–памяти не может быть изменено, пока ключ не будет переведен в положение OFF.

Начиная с версии 7 ЦП 351 и 352, этот переключатель выполняет другую функцию: он позволяет переводить ПЛК в режим **STOP**, в режим **RUN** и очищать сообщения о не фатальных сбоях, как описано в следующем параграфе.

Начиная с версии 8 ЦП 35x и 36x, этот переключатель выполняет дополнительные функции защиты памяти: он может включать два дополнительных режима защиты (см. параграф «Дополнительная защита памяти в ЦП версии 8 и выше»).

Если ключ установлен в положение **ON**/**RUN**, Вы можете изменить системную дату и время только программно. Ручной переносной программатор не даст Вам возможности изменить время, пока защита ключом не будет снята.

Использование ключа в версиях 7 и выше

В отличие от функции защиты Flash–памяти в ранних версиях ЦП, описываемые здесь расширенные функции защиты не будут активизированы, пока Вы не включите параметр «**RUN**/**STOP** Key Switch» в окне конфигурации ЦП.

Действие поворота ключа равнозначно подаче обычной команды **RUN**, при этом ПЛК не может быть переведен ключом в режим **RUN**, если он находится в режиме **STOP**/**FAULT**. Однако в режиме **STOP/FAULT** Вы можете с помощью ключа очистить все не фатальные сбои и вновь перевести ПЛК в режим **RUN**.

Если в таблице сбоев есть записи о ***не фатальных*** сбоях (сбои, не приводящие к переходу ПЛК в режим **STOP**/**FAULT**), то ПЛК будет переведен в режим **RUN** первым же поворотом ключа от положения Stop к положению Run, при этом таблица сбоев НЕ очищается.

Если в таблице сбоев есть записи о ***фатальных*** сбоях (ЦП перешел в режим **STOP**/**FAULT**), то первый перевод ключа из положения **STOP** в положение **RUN** приводит к миганию индикатора **RUN** с частотой 2 Гц в течение 5 секунд. Мигание индикатора - признак того, что в таблице записаны фатальные сбои. В этом случае ПЛК не переходит в режим **RUN**, не смотря на поворот ключа.

Очистка таблицы сбоев с помощью ключа

Если Вы переведете ключ из положения **RUN** в положение **STOP**, а затем обратно в положение **RUN**, пока длится пятисекундное мигание индикатора **RUN**, таблица сбоев очищается и ПЛК переходит в режим **RUN**. С этого момента индикатор горит непрерывно. При переключениях необходимо удерживать ключ в каждом положении (**RUN**, **STOP** и снова **RUN**) не менее ½ секунды.

Примечание

Если 5 секунд истекут (индикатор **RUN** прекратит мигать), ПЛК останется в исходном режиме **STOP**/**FAULT**, таблица сбоев не очистится. Если Вы переведете ключ из положения **STOP** в положение **RUN** еще раз, система будет вести себя так же, как и при первом переключении.

В следующей таблице приведены сведения о влиянии параметров конфигурации ЦП (R/S Switch и IOScan-Stop) и положения ключа на состояние ПЛК.

Параметр "R/S Key Switch" в конфигурации ЦП	Положение ключа	Параметр "IO-Scan-Stop" в конфигурации ЦП	Состояние ПЛК
OFF	X	X	Разрешены все режимы функционирования ПЛК.
ON	ON/RUN	X	Разрешены все режимы функционирования ПЛК.
ON	OFF/STOP	X	ПЛК запрещен переход в режим RUN.
ON	Переход из положения OFF/STOP в ON/RUN	X	ПЛК переходит в режим RUN, если нет фатальных сбоев, иначе индикатор RUN мигает в течение 5 секунд.
ON	Переход из положения ON/RUN в OFF/STOP	NO	ПЛК переходит в режим STOP–NO IO
ON	Переход из положения ON/RUN в OFF/STOP	YES	ПЛК переходит в режим STOP–IO

X= не имеет значения

Дополнительная защита памяти в ЦП версии 8 и выше

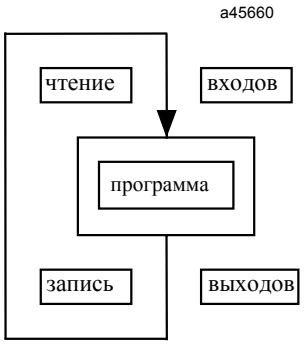
В ЦП версии 8 и выше ключ выполняет все описанные выше функции, а также с помощью дополнительного параметра в конфигурации ЦП позволяет защитить данные в ОЗУ от изменения из внешнего программного обеспечения. Данная функция блокирует два типа изменений: запрещается изменение конфигурации и прикладной программы; запрещается запись и подстановки в памяти данных. Функция активизируется полем Mem Protect окна конфигурации ЦП 35x или 36x в программе Logicmaster. Значение по умолчанию – Disabled.

Раздел 2: Организация программы пользователя и данных

Общий размер программы для ПЛК Series 90-30 приведен в следующей таблице.

Модели ЦП	Размер памяти под программу (Кбайт)
CPU311	6
CPU313, CPU323	12
CPU331	16
CPU340	32
CPU341	80
CPU350	80 (для версии 9 и более поздней) 32 (для более ранних версий)
CPU351, CPU352, CPU360, CPU363, CPU364	240 (для версии 9 и более поздней) 80 (для более ранних версий)

Начиная с версии 9, размер памяти для ЦП серий 351, 352 и 36x может быть настроен пользователем. (Более детальная информация находится в руководстве *Logicmaster 90™. Среда программирования ПЛК Series 90™-30/20/Micro. Руководство пользователя* (GFK-0466K) часть 10 раздел 3 “Настраиваемая память в ЦП моделей 351 и старше”). Программа для ПЛК Series 90-20 с 211 моделью процессора должна иметь размер до 2 KB. Программа пользователя содержит логику, которая выполняется при запуске ПЛК. Максимальное число шагов (rungs), из которых может состоять каждый программный блок, ограничивается 3000; для ПЛК 90-30, максимальный размер блока 80 Кбайт для блоков написанных на языке СИ и 16 Кбайт для блоков написанных на языках LD и SFC. В некоторых SFC-блоках часть памяти используются внутренними блоками. ПЛК выполняет программу постоянно.



Для ознакомления с возможностями и ограничениями каждой модели ЦП обратитесь к руководствам *Программируемые контроллеры Series 90-30. Руководство пользователя,* GFK-0356, или *Программируемые контроллеры Series 90-20. Руководство пользователя*, GFK-0551.

Каждый программный блок имеет таблицу переменных, которая содержит описания переменных и ссылок, которые используются в программе пользователя.

Редактор блоков, содержит список подпрограмм, описанных в программе.

Блоки подпрограмм (только для ПЛК Series 90-30)

Программа может запустить выполнение любой подпрограммы. Подпрограмма должна быть описана в Редакторе, перед выполнением инструкции вызова «CALL». Максимальное количество используемых в программе подпрограмм, ограничено 64 и разрешено не более 64 инструкций CALL для каждого программного блока. Максимальный размер подпрограммы ограничен 16 КВ или 3000 шагами, но программа, включая все подпрограммы, должна умещаться в памяти ЦП, размер которой зависит от типа ЦП.

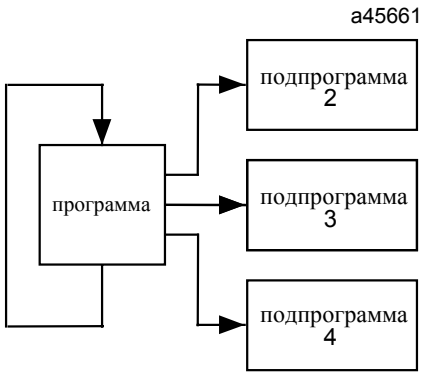
Примечание

Подпрограммы нельзя использовать в ПЛК Series 90-20 и 90-Micro.

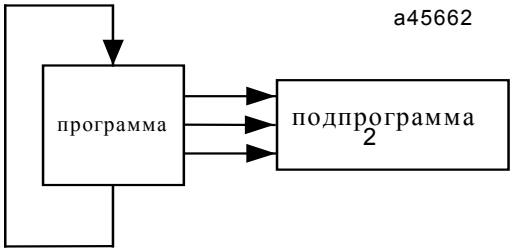
Использование подпрограмм является необязательным. Однако деление программы на подпрограммы может заметно упростить программирование, понимание алгоритма управления, и сокращает общее количество логических элементов в программе.

Примеры использования блоков подпрограмм

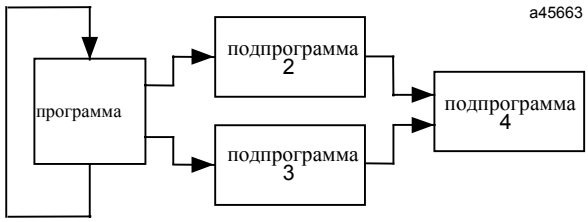
Например, программа содержит три блока подпрограмм, каждый из которых при необходимости может быть вызван из главной программы. В этом примере небольшой по размеру блок главной программы использует, в основном, блоки подпрограмм.



Подпрограмма может выполняться любое количество раз во время работы программы. Программный блок, который необходимо выполнить в программе несколько раз, может быть оформлен как подпрограмма. Каждый вызов такой подпрограммы обеспечит выполнение программного блока.



Подпрограмма, будучи вызвана из программы, также может вызывать другие подпрограммы, включая саму себя.



В программе возможно использование не более восьми вложений, иначе произойдет сбой “Application Stack Overflow”, который переведет ПЛК в режим **STOP/Fault**. Уровни вложенности считают, начиная с первого.

Как происходит вызов подпрограммы

Подпрограмма выполняется, при вызове ее из программы или из другой подпрограммы.

```
|
|%I0004                                                                %T0001
|──| |──────────────────────────────────────────────────────────────────( )─
|
|%I0006    |               |
|──| |─────|  CALL ASTRO   |─
|          | (подпрограмма) |
|          |_______________|
|
|%I0003  %I0010                                                        %Q0010
|──| |─────| |──────────────────────────────────────────────────────────( )─
|
```

Этот пример показывает использование инструкции CALL для вызова подпрограммы.

Периодические подпрограммы

Все модели ЦП старше 340 поддерживают периодические подпрограммы. Запомните следующие ограничения:

1. Таймеры (TMR, ONDTR, и OFDTR) не будут правильно работать в теле периодической подпрограммы. Использование блока DOIO в теле периодической подпрограммы, чей диапазон ссылок включает ссылки, назначенные для интеллектуальные модулей B/B (высокоскоростные счетчики, Power Mate APM, модули Genius и т.д.) приведет к потере связи ЦП с данным модулем. Контакты FST_SCN и LST_SCN (%S1 и %S2) будут иметь неопределенное значение в течение выполнения периодической подпрограммы. Периодическая подпрограмма не может вызывать другие подпрограммы и не может быть вызвана из других подпрограмм.

2. Латентность периодической подпрограммы (то есть максимальный промежуток времени, между заложенным временем выполнения и временем за которое она действительно была выполнена) составляет примерно 0.35 миллисекунд, если в основной стойке отсутствуют модули PCM, CMM, или ADC. Если модули присутствуют в основной стойке, то латентность может достигать значения в 2.25 миллисекунды, даже если эти модули не сконфигурированы или не используется. По этой причине, использование периодической подпрограммы в приложениях, использующих модули PCM, не рекомендуется.

Пользовательские ссылки

Данные, используемые в прикладных программах, сохраняются или в регистрах или в дискретных ссылках.

Таблица 2-4. Регистровые ссылки

Тип	Описание
%R	Ссылка используется для адресации к регистрам памяти, где хранятся данные, результаты вычислений и данные, полученные от счетчиков и таймеров.
%AI	Префикс %AI регистр аналогового ввода. После префикса следует адрес ссылки (например, %AI0015). Регистр аналогового ввода хранит значения одного аналогового входа или другое значение.
%AQ	Префикс %AQ регистр аналогового вывода. После префикса следует адрес ссылки (например, %AQ0056). Регистр аналогового вывода Регистр аналогового вывода хранит значение одного аналогового выхода или другое значение.

Примечание

Все регистровые ссылки удерживаются, пока ЦП включен.

Таблица 2-5. Дискретные ссылки

Тип	Описание
%I	Префикс %I представляет ссылки дискретных входов. После префикса следует адрес в таблице ввода (например, %I00121). Ссылки, начинающиеся с %I, располагаются в статусной таблице ввода, которая хранит данные, полученные от всех модулей ввода за последний цикл. Каждому модулю дискретного ввода назначается начальный адрес с помощью конфигурационного ПО или ручного программатора. До тех пор пока начальный адрес не будет назначен модулю, информация от него поступать не будет. Данные в памяти %I могут быть как удерживаемыми, так и не удерживаемыми. *
%Q	Префикс %Q указывает на ссылки дискретных выходов. Функция проверки обмоток в Logicmaster 90-30/20/Micro проверяет многократное использование ссылок типа %Q с обмотками реле или выходами функций. Начиная с версии 3, Вы можете выбирать режим проверки использования обмоток (`SINGLE`, `WARN MULTIPLE` или `MULTIPLE`). После префикса следует адрес в таблице вывода (например, %Q00016). Ссылки, начинающиеся с %Q, располагаются в статусной таблице вывода, которая хранит состояния всех выходов, как последние установленные приложением значения. Эти значения посылаются в модули вывода во время сканирования выходов. Каждому модулю дискретного вывода назначается начальный адрес с помощью конфигурационного ПО или ручного программатора. До тех пор пока начальный адрес не будет назначен модулю, информация ему посылаться не будет. Данные в памяти %Q могут быть как удерживаемыми, так и не удерживаемыми. *
%M	Префикс %M указывает на внутренние ссылки. Функция проверки обмоток проверяет многократное использование ссылок типа %M с обмотками реле или выходами функций. Начиная с версии 3, Вы можете выбирать режим проверки использования обмоток (`SINGLE`, `WARN MULTIPLE` или `MULTIPLE`). За дополнительной информацией обратитесь к руководству GFK-0466. Данные в памяти %M могут быть как удерживаемыми, так и не удерживаемыми. *

* Удерживаемость зависит от типа обмотки. Подробнее см. «Способность удерживать данные» на стр. 2-20.

%T	Префикс %T указывает на временные ссылки. Так как эти ссылки никогда не проверяются на многократное использование с обмотками, они могут использоваться много раз в одной программе, даже при включенной проверке обмоток. Ссылки %T типа могут применяться для предотвращения конфликтов использования обмоток во время выполнения функций вырезать/вставить и записать в файл/включить. Так как данный тип памяти предназначен для временного хранения данных, то при выключении питания или переходах **RUN-STOP-RUN** данные теряются. Также этот тип памяти не может быть применен для обмоток с удержанием.
%S	Префикс %S указывает на системную область памяти. Эти ссылки используются для доступа к специальным данным ПЛК, таким как таймеры, информация о сканировании В/В и информации о сбоях. Системные ссылки включают %S, %SA, %SB, и %SC Ссылки %S, %SA, %SB, и %SC могут использоваться в любых контактах. %SA, %SB, and %SC могут использоваться в удерживающих обмотках –(M)–.
%G	Префикс %G указывает на ссылки глобальных данных. Используются для обмена данным между несколькими ПЛК. Ссылки %G типа могут использоваться в контактах и удерживающих обмотках, так как данный тип памяти с удержанием. Ссылки %G типа не могут использоваться в не удерживающих обмотках.

Переходы и подстановки

Пользовательские ссылки %I, %Q, %M, и %G имеют биты переходов и подстановок. Ссылки %T, %S, %SA, %SB, и ссылки %SC имеют биты переходов, но не имеют подстановочных бит. Биты переходов используются ЦП в счетчиках и обмотках. Счетчики и обмотки в свою очередь используют разный тип битов перехода. Биты перехода для счетчиков сохраняются в назначенной им ссылке.

Начиная с 331 модели ЦП, значения подстановочных бит могут быть заданы пользователем. Когда значения подстановочных битов заданы, связанные с ними ссылки не могут быть изменены программой или устройством ввода данных; они могут быть изменены только по команде программатора. Модели ЦП 323, 321, 313, и 311, а также Micro ЦП не поддерживают подстановочные дискретные ссылки.

Способность удерживать данные

Данные считаются удержанными, если они сохранены контроллером, когда он остановлен. ПЛК Series 90 сохраняет программу, таблицы сбоев и результаты диагностики, значения подстановок и выходные значения, регистровые данные (%R, %AI, %AQ), битовые данные (%I, %SC, %G, биты сбоев и резервные биты), данные в памяти %Q и %M (кроме случаев использования обмоток без удержания), и данные размером в слово, сохраненные в памяти %Q и %M. Данные из памяти %T не сохраняются. Хотя, как заявлено выше, битовые данные в памяти %SC удерживаются, по умолчанию, данные в памяти %S, %SA, и %SB не удерживаются.

Память %Q и %M не удерживается, (то есть обнуляется при включении питания, когда ПЛК переходит из режима STOP в режим RUN) всякий раз, когда она используются с не удерживаемыми обмотками. К ним относятся: обычные обмотки —( )—, инверсные обмотки —(/)— , обмотки SET —(S)—, и обмотки RESET —(R)—.

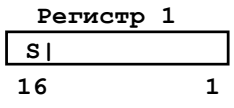
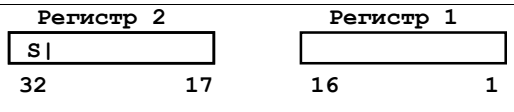
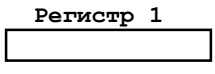
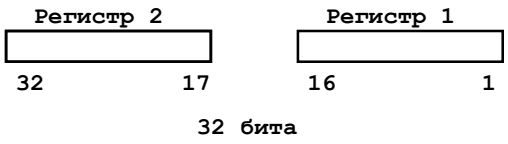
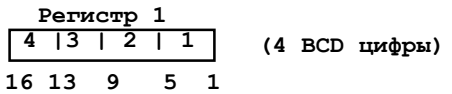
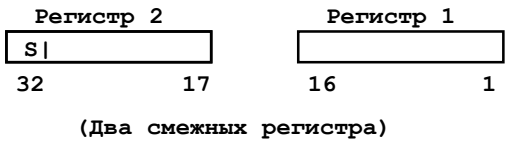
Когда память %Q или %M используется с удерживающими обмотками, или в качестве блоков вывода, значения сохраняются в памяти при переключениях режимов **RUN-STOP-RUN**. Удерживающие обмотки включают в себя: удерживающие обмотки —(M)—, инверсные

удерживающие обмотки —(/M)—, удерживающие обмотки SET —(SM)—, и удерживающие обмотки RESET—(RM)—.

То, как в последний раз была установлена %Q или %M ссылка, определяет, будет ли %Q или %M ссылка удерживаемой или нет. Например, если ссылка %Q0001 в последний раз была установлена как ссылка удерживающей обмотки, данные, по адресу %Q0001 будут удержаны. Однако, если ссылка %Q0001 в последний раз была установлена как ссылка не-удерживающей обмотки, данные, по адресу %Q0001 не будут удержаны.

Типы данных

Таблица 2-6. Типы данных

Тип	Имя	Описание	Формат данных
INT	Знаковое целое	Знаковые целые числа, использующие 16 разрядные регистры. Бит S – это знак. Диапазон значений типа INT представлен –32 768 до +32 767.	
DINT	Двойное знаковое целое	Данные этого типа хранятся в 32-х разрядной памяти (это два последовательных 16-ти разрядных регистра) 32-ой бит - это знак. Диапазон значений DINT от –2 147 483 648 до +2 147 483 647.	
BIT	Бит	Наименьшая единица памяти. Она имеет только два возможных значения: 1 или 0.	
BYTE	Байт	Байт – это тип данных, длиной 8 бит. Диапазон значений от 0 до 255 (от 0 до FF в шестнадцатеричном представлении).	
WORD	Слово	Для слова используется 16 бит, однако вместо бит в ячейке данных представлено число, биты независимы друг от друга. Каждый бит представлен своим собственным двоичным состоянием (1 или 0), и биты не рассматриваются вместе как целое число. Действительный диапазон значений слова от 0 до FFFF.	
DWORD	Двойное слово	Двойное слово – это два последовательных регистра, аналогично типу DINT, только без знакового бита.	
BCD-4	Четырехзначный двоично-десятичный код	Число в данном формате использует 16-ти разрядный регистр. Каждая цифра BCD использует четыре бита и может держать число от 0 до 9. Диапазон значений от 0 до 9999.	
REAL	Floating Point	Действительные числа используют 32 последовательных бита (обычно две последовательных 16-ти битовых ячейки памяти). Диапазон чисел, которые могут быть представлены в данном формате, составляет от ±1,401298E-45 до ±3,402823E+38.	

S = Знаковый бит (0 = число положительное, 1 = число отрицательное).

Системные ссылки

Системные ссылки в ПЛК Series 90 расположены в памяти %S, %SA, %SB и %SC. Для каждой такой ссылки определено имя.

Примечание

Биты, расположенные в памяти %S предназначены только для чтения. Для записи данных вы можете использовать память %SA, %SB и %SC.

Приведенный ниже список содержит системные ссылки, которые могут быть использованы в прикладной программе. Во время написания программы можно использовать как ссылки, так и имена. В третьей части данного руководства более подробно описаны возможные сбои и действия по их устранению.

Вы не можете использовать эти зарезервированные имена для других целей.

Таблица 2-7. Системные ссылки

Ссылка	Имя	Определение
%S0001	FST_SCN	Устанавливается в 1. только на время первого цикла ЦП.
%S0002	LST_SCN	Сбрасывается из 1 в 0 когда текущий цикл, является последним.
%S0003	T_10MS	Контакт таймера 0.01 секунды.
%S0004	T_100MS	Контакт таймера 0.1 секунды.
%S0005	T_SEC	Контакт таймера 1.0 секунда.
%S0006	T_MIN	Контакт таймера 1.0 минута.
%S0007	ALW_ON	Всегда установлен в ON (1).
%S0008	ALW_OFF	Всегда установлен в OFF (0).
%S0009	SY_FULL	Устанавливается в 1, когда таблица сбоев ПЛК заполняется целиком. Сбрасывается, когда записи удаляются из таблицы или таблица очищается целиком.
%S0010	IO_FULL	Устанавливается в 1, когда таблица сбоев В/В заполняется целиком. Сбрасывается, когда записи удаляются из таблицы или таблица очищается целиком.
%S0011	OVR_PRE	Устанавливается в 1, когда существуют подстановки в памяти %I, %Q, %M или %G.
%S0013	PRG_CHK	Устанавливается в 1, когда включена фоновая проверка программы.
%S0014	PLC_BAT	Устанавливается в 1, указывая на разряженную батарею для ЦП версии 4 или старше. Обновляется каждый цикл ЦП.

Таблица 2-7. Ссылки состояния системы - продолжение

Ссылка	Имя	Определение
%S0017	SNPXACT	Хост SNP-X подключен к ЦП.
%S0018	SNPX_RD	Хост SNP-X читает данные из ЦП.
%S0019	SNPX_WT	Хост SNP-X записывает данные в ЦП.
%S0020		Устанавливается в 1, когда функции отношений, использующих тип REAL, выполняется успешно. В случае NaN (не число) сбрасывается в 0.
%S0032		Зарезервировано.
%SA0001	PB_SUM	Устанавливается в 1, когда вычисленная контрольная сумма пользовательской программы не совпадает с предыдущей. Если сбой произошел из-за временной ошибки, бит сбрасывается при повторной записи программы в ЦП. Если сбой произошел из-за проблем с аппаратной частью, ЦП должен быть заменен.
%SA0002	OV_SWP	Устанавливается в 1, когда ПЛК обнаруживает, что предыдущий цикл выполнялся дольше чем, время установленное пользователем. Сбрасывается, когда ЦП обнаруживает, что цикл не превышает заданное время. Он также сбрасывается при выполнении перехода ЦП из режима **STOP** в режим **RUN.** Достоверен только когда ЦП находится в режиме **COSTANT SWEEP**.
%SA0003	APL_FLT	Устанавливается в 1, когда произошел сбой программного обеспечения ПЛК. Сбрасывается, когда ПЛК выполняет переход из режима **STOP** в режим **RUN**.
%SA0009	CFG_MM	Устанавливается в 1, когда обнаруживается несоответствие системной конфигурации при включении ПЛК или при сохранении конфигурации. Сбрасывается, если при включении ПЛК все несоответствия устранены, или при записи правильной конфигурации.
%SA0010	HRD_CPU	Устанавливается в 1, когда обнаруживается проблема с аппаратным обеспечением модуля ЦП. Сбрасывается путем замены модуля ЦП.
%SA0011	LOW_BAT	Устанавливается в 1, когда падает напряжение на батарее. Сбрасывается при замене батареи.
%SA0014	LOS_IOM	Устанавливается в 1, когда модуль В/В прерывает связь с модулем ЦП ПЛК. Сбрасывается заменой модуля и повторным включением ПКЛ.
%SA0015	LOS_SIO	Устанавливается в 1, когда дополнительный модуль прерывает связь с модулем ЦП ПЛК. Сбрасывается заменой модуля и повторным включением ПКЛ.
%SA0019	ADD_IOM	Устанавливается в 1, когда модуль В/В добавляется в стойку. Сбрасывается повторным включением ПКЛ, при условии отсутствия несоответствий в конфигурации.
%SA0020	ADD_SIO	Устанавливается в 1, когда дополнительный модуль добавляется в стойку. Сбрасывается повторным включением ПКЛ, при условии отсутствия несоответствий в конфигурации.
%SA0027	HRD_SIO	Устанавливается в 1, когда обнаружен сбой аппаратного обеспечения дополнительного модуля. Сбрасывается заменой модуля и повторным включением ПКЛ
%SA0031	SFT_SIO	Устанавливается в 1, когда произошел неисправимый сбой программного обеспечения дополнительного модуля. Сбрасывается повторным включением ПКЛ, при условии отсутствия несоответствий в конфигурации.
%SB0010	BAD_RAM	Устанавливается в 1, когда модуль ЦП обнаруживает нарушение целостности ОЗУ (RAM-памяти) при включении питания. Сбрасывается, когда ЦП обнаруживает наличие работоспособной памяти.

Таблица 2-7. Ссылки состояния системы - продолжение

Ссылка	Имя	Определение
%SB0011	BAD_PWD	Устанавливается в 1, когда неправильно введен пароль доступа. Сбрасывается, когда таблица сбоев ПЛК очищена.
%SB0013	SFT_CPU	Устанавливается в 1, когда модуль ЦП обнаруживает неисправимую ошибку в программном обеспечении. Сбрасывается, когда таблица сбоев ПЛК очищена.
%SB0014	STOR_ER	Устанавливается в 1, когда происходит ошибка при операции сохранения. Сбрасывается при успешном завершении операции сохранения.
%SC0009	ANY_FLT	Устанавливается в 1, когда происходит любой сбой. Сбрасывается, когда обе таблицы сбоев пусты.
%SC0010	SY_FLT	Устанавливается в 1, когда происходит любой сбой, который является причиной записи его в таблицу сбоев ПЛК. Сбрасывается, когда таблица сбоев ПЛК пуста.
%SC0011	IO_FLT	Устанавливается в 1, когда происходит любой сбой, который является причиной записи его в таблицу сбоев В/В. Сбрасывается, когда таблица сбоев В/В пуста.
%SC0012	SY_PRES	Установлен в 1, пока в таблице сбоев ПЛК есть хоть одна запись. Сбрасывается, когда таблица сбоев ПЛК пуста.
%SC0013	IO_PRES	Установлен в 1, пока в таблице сбоев В/В есть хоть одна запись. Сбрасывается, когда таблица сбоев В/В пуста.
%SC0014	HRD_FLT	Устанавливается в 1 при наличии аппаратного сбоя. Сбрасывается, когда обе таблицы сбоев пусты.
%SC0015	SFT_FLT	Устанавливается в 1 при наличии программного сбоя. Сбрасывается, когда обе таблицы сбоев пусты.

Примечание: Все остальные (не перечисленные здесь) ссылки памяти %S являются служебными и не могут быть использованы в прикладной программе пользователя.

Структура функциональных блоков

Каждый шаг программы может содержать одну или более инструкций. Это могут быть простые реле или более сложные функции.

Формат релейной логики

Система команд ПЛК включает несколько типов релейных функций. С помощью этих функциях строятся основные блоки программы. Примерами являются нормально открытый релейный контакт и инверсная обмотка. Каждый из этих контактов и обмоток имеет один вход и один выход. Вместе, они обеспечивают правильную работу логической схемы.
Каждому контакту или обмотке необходимо назначить ссылку, которая вводится при выборе данного контакта. Для контакта, ссылка представляет собой место в памяти, которая определяет прохождение потока энергии через контакт. Пример: если ссылка %I0122 установлена в 1, поток энергии, будет проходить сквозь этот релейный контакт.

```
%I0122
 -| |-
```

Для обмотки, ссылка представляет собой ячейку памяти, значение которой управляется поступлением энергии на обмотку. Пример: если поток энергии поступает на левую сторону обмотки, ссылка %Q0004 устанавливается в 1.

```
%Q0004
-( )-
```

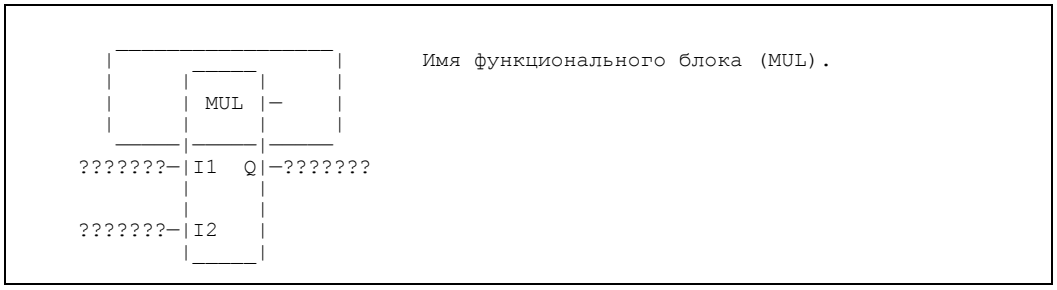
Как программное обеспечение для программирования, так и ручной программатор, имеют функцию проверки обмотки, которая проверяет многократные использования памяти %Q или %M с обмотками или функциональными выходами.

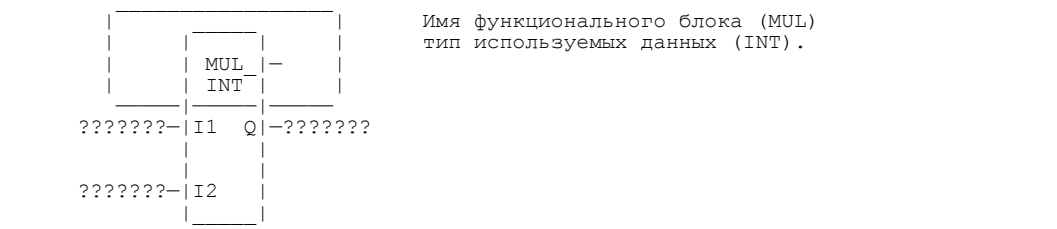
Формат программных функциональных блоков

Некоторые программные функции очень просты, подобно функции MCR, которая представлена сокращенным названием функции в скобках:

```
-[ MCR ]-
```

Другие функции более сложные. Они могут иметь несколько мест, где Вы можете ввести информацию, необходимую функции.
Типичный функциональный блок, проиллюстрированный ниже – умножение (MUL); параметры блока изменяются в зависимости от типа функционального блока. Многие его части одинаковы для большинства других программных функций. Верхняя часть функционального блока показывает название функции. Там также может находиться информация об используемом типе данных; в нашем примере это целом число со знаком.



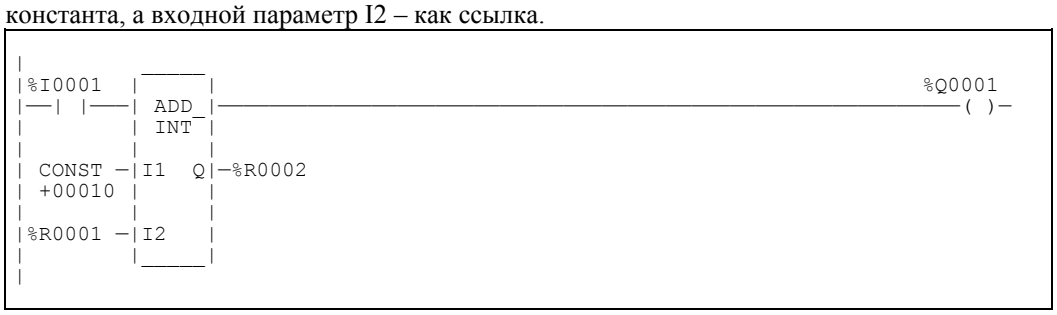


Большинство программных функций позволяют Вам выбрать тип данных для функции после выбора самой функции. Например, тип данных для функции MUL может быть изменен на DINT (целое знаковое число с двойной точностью). Дополнительная информация относительно типов данных была приведена ранее в этой главе.

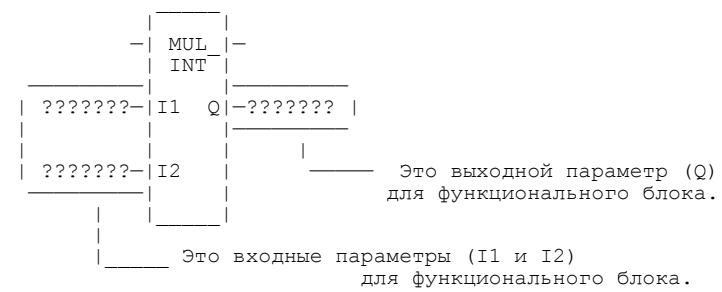
Параметры функциональных блоков

Каждая линия, подходящая ***слева*** к функциональному блоку, представляет собой ***вход*** для этой функции. Есть два типа входных значений, которые можно передать в функциональный блок: константы и ссылки. Константы – точное значение параметра. Ссылка – адрес ячейки памяти, содержащей значение.

В следующем примере, входной параметр I1 поступает в функциональный блок ADD как константа, а входной параметр I2 – как ссылка.

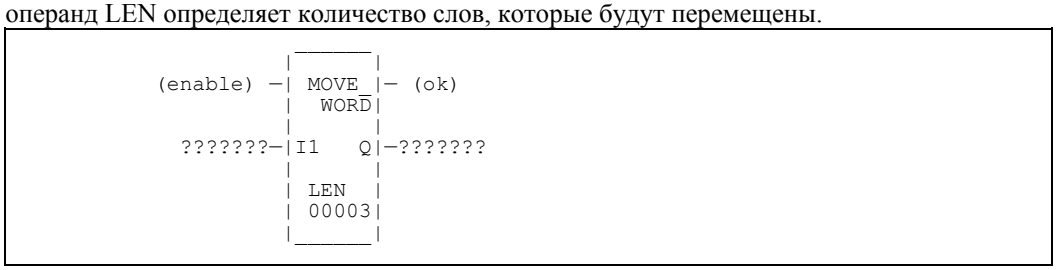


Каждая линия, выходящая с ***правой*** стороны функционального блока, является ***выходом*** этого блока. Возможен только один тип для выходного значения функционального блока – ссылка. Выходное значение не может быть записано в константу. Вместо расположенных слева от функционального блока вопросительных знаков, Вы будете либо сразу вводить данные, либо будете вводить ссылку на область памяти, где эти данные расположены, либо переменную, которая указывает на ссылку, которая в свою очередь указывает на данные. Вместо расположенных справа от функционального блока вопросительных знаков, Вы будете вводить ссылку на область памяти, куда следует поместить выходное значение, либо переменную, которая указывает на ссылку, которая в свою очередь указывает на область памяти, куда следует поместить выходное значение.

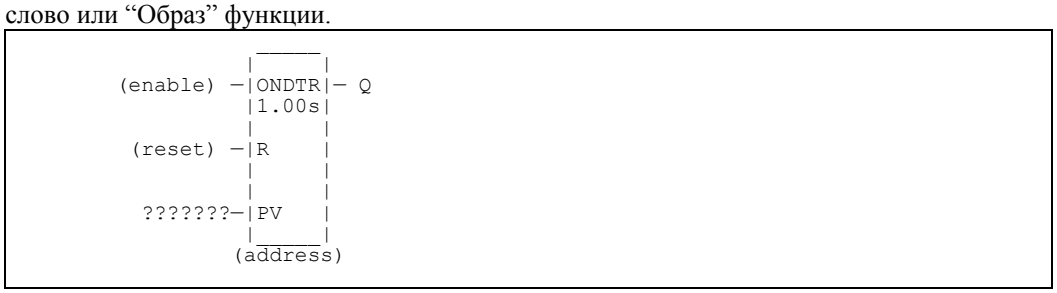


Большинство функциональных блоков не изменяют входные значения; вместо этого, они размещают результат операции в выходной ссылке.

Для функций, которые работают с массивами, может быть выбран размер массива. Пример: операнд LEN определяет количество слов, которые будут перемещены.

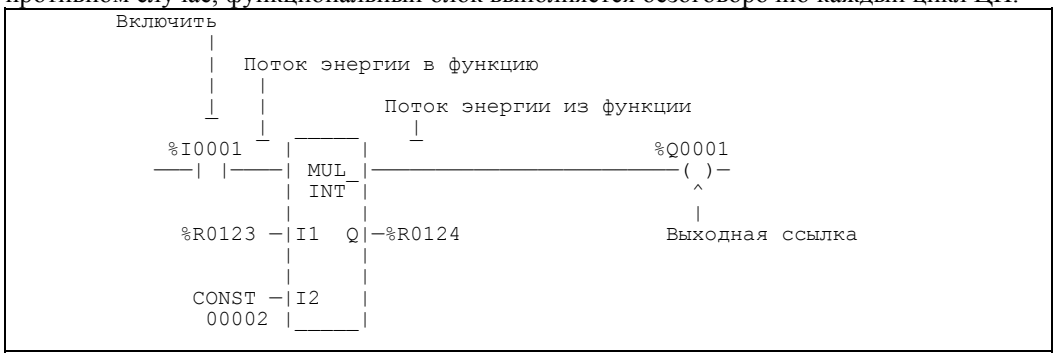


Функциональные блоки: таймер, счетчик, BITSEQ, и ID требуют адрес для размещения трех слов (регистры), которые сохраняют текущее значение, заданное значение, и управляющее слово или “Образ” функции.



Поток энергии в функцию и из функции

Поток энергии в функциональный блок поступает через верхний левый вход. Часто для управления потоком энергии к функциональному блоку используется сигнал включения; в противном случае, функциональный блок выполняется безоговорочно каждый цикл ЦП.



Примечание

Функциональные блоки не могут быть подключены непосредственно к шине питания. Вы можете использовать специальную системную переменную %S7, ALW_ON (всегда ON) вместе с нормально открытым контактом, присоединенным к шине, чтобы вызывать функцию на каждом цикле.

Поток энергии выходит из правого верхнего выхода функционального блока. Дальше он может поступать к другому блоку или к обмотке (не обязательно). Если блок выполняется успешно, то поток энергии поступает на выход данного блока.

Раздел 3: Последовательность включения и отключения ПЛК

Есть два возможных способа включения контроллеров Series 90-30: “холодное” включение и “теплое ” включение. Обычно ЦП использует “холодное” включение. Однако, в модели ЦП 331 и выше, если время, прошедшее между выключением и последующим включением менее пяти секунд, используется “теплое” включение.

Включение

“Холодное” включение питания состоит из нескольких шагов. При “теплом” включении шаг №1 пропускается.

1. ЦП диагностирует сам себя. Оперативная память, питающаяся от батареи питания, проверяется на наличие достоверных данных.

2. Если имеется EPROM, EEPROM, или флэш-память, и при включении определяется необходимость использования PROM, то содержимое PROM копируется в оперативную память. Если не имеется EPROM, EEPROM, или флэш-памяти, то в оперативную память ничего не записывается.

3. ЦП опрашивает каждый слот системы, чтобы определить наличие сетевых плат.

4. Сравнение аппаратной конфигурации с программной конфигурацией. Любые несоответствия рассматриваются как сбои и регистрируются. Например, если модуль определен в программной конфигурации, а в аппаратной конфигурации присутствует другой модуль, это рассматриваются как сбой и регистрируются.

5. Если не установлена программная конфигурация, то ЦП устанавливает конфигурацию по умолчанию.

6. ЦП устанавливает связь с другими микропроцессорными модулями в системе.

7. На последнем шаге включения, определяется режим цикла на основе конфигурации ЦП. Если установлен режим **RUN**, выполняется процедура “Переход из режима из **STOP** в **RUN**”, показанная на следующей странице на Рисунке 2-5.

Примечание

Шаги 2 до 7 не используется в ПЛК Series 90 Micro. Для получения информации о включении и выключении Series Micro обратитесь к главе 5, руководства пользователя ПЛК Series 90 Micro (GFK-1065) “Последовательность включения и выключения”.

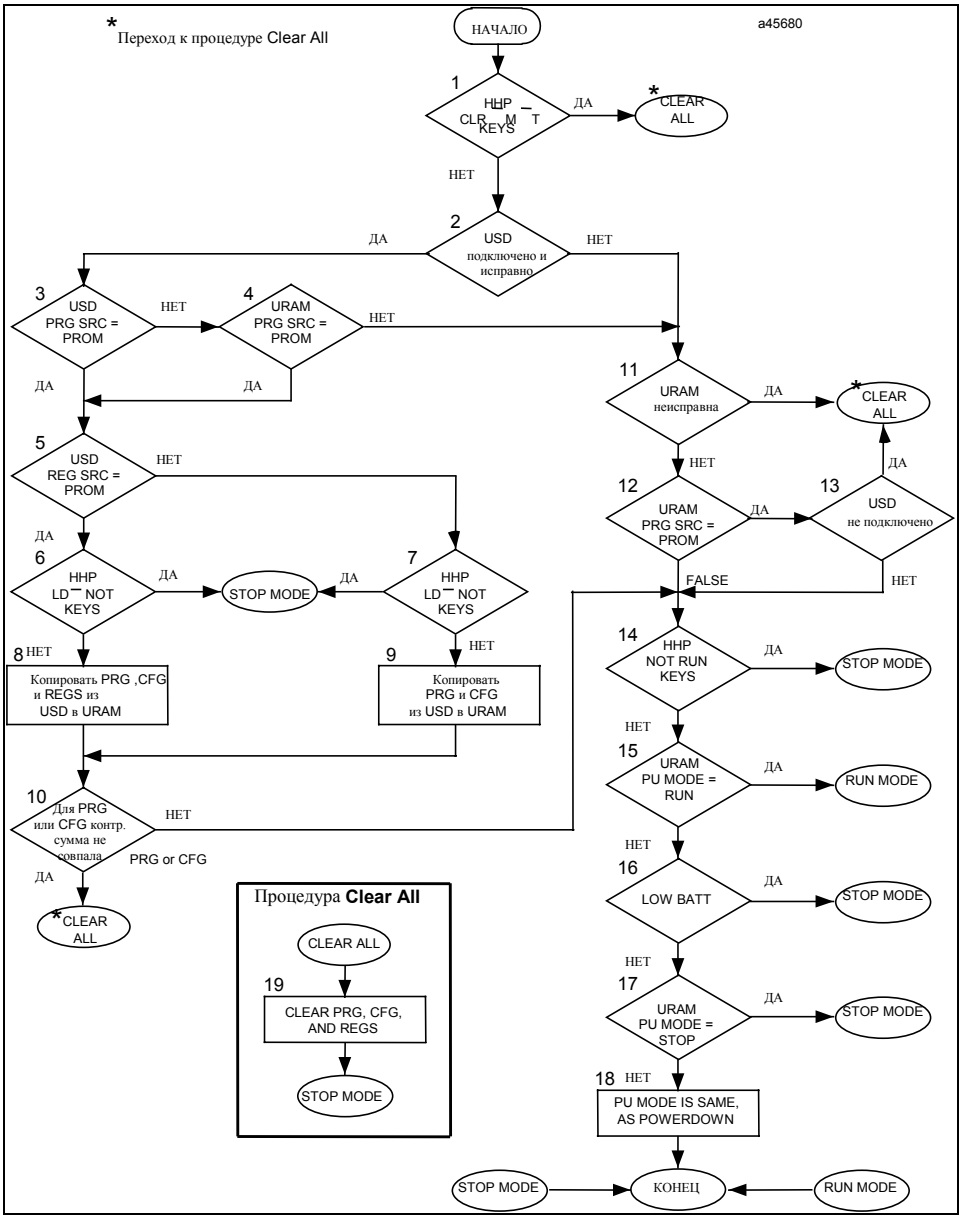


Рисунок 2-5. Последовательность включения

До команды НАЧАЛО в блок-схеме включения, ЦП проводит диагностику используемых периферийных устройств и оперативной памяти. После завершения диагностики внутренние структуры данных инициализируются. ЦП определяет работоспособность ОЗУ. Если ОЗУ не работоспособно, то пользовательские программы и конфигурации стираются и устанавливаются значения по умолчанию, стираются регистры пользователя.

Обозначения на блок-схеме:

PRG = программа пользователя

CFG = конфигурация пользователя

REGS = регистры пользователя (%I, %Q, %M, %G, %R, %AI, и %AQ ссылки).

USD = запоминающее устройство пользователя, – EEPROM или флэш-память.

URAM = долговременное ОЗУ, которое содержит PRG, CFG, и REGS.

Описание блок-схемы:

(1) Нажаты ли клавиши <CLR> и <M_T> на HHP во время включения питания, чтобы очистить URAM?

(2) Подключено ли USD (может быть пропущено только в моделях, использующих EEPROM) и содержит ли оно корректную информацию?

(3) Если параметр PRG SRC в USD имеет значение «Prom», загрузить PRG и CFG из USD.

(4) Если параметр PRG SRC в URAM имеет значение «Prom», загрузить PRG и CFG из USD.

(5) Если параметр REG SRC в USD имеет значение «Prom», загрузить REGS из USD.

(6 и 7) Нажаты ли клавиши <LD> и <NOT> на HHP во время включения питания, чтобы предотвратить загрузку PRG, CFG, и REGS из USD?

(8) Копируются PRG, CFG, и REGS из USD в URAM.

(9) Копируются PRG, и CFG из USD в URAM.

(10) Корректны ли PRG или CFG, только что загруженные из USD?

(11) URAM неисправно? Эта ситуация может возникнуть вследствие разряда батареи. Могло возникнуть при обновлении firmware.

(12) Если параметр PRG SRC в URAM имеет значение «Prom», загрузить PRG и CFG из USD.

(13) Подключено ли USD? Применяется только к моделям, использующим EEPROM.

(14) Нажаты ли клавиши <**NOT**> и <**RUN**> на HHP во время включения питания? Приводит к включению в режиме STOP.

(15) Параметр PWR UP в URAM установлен в **RUN?**

(16) Батарея разряжена?

(17) Параметр PWR UP в URAM установлен в **STOP?**

(18) Восстанавливается режим работы, который был при отключении.

(19) Очистка PRG, CFG, и REGS.

Примечание

Первая часть блок-схемы на предыдущей странице не используется в ПЛК Series 90 Micro. Для получения информации о включении и выключении Series 90 Micro обратитесь к главе 5 руководства пользователя ПЛК Series 90 Micro (GFK-1065) «Последовательность включения и выключения».

Отключение

Отключение системы происходит, когда выходное напряжение источника питания 5В становится ниже 4,9В.

Раздел 4: Таймеры

Часы и таймеры представлены в ПЛК Series 90-30, следующими видами: часы времени работы, часы астрономического времени (модели ЦП 331, 340/341, 351/352 и в 28-канальном Micro), сторожевой таймер и таймер постоянного цикла. Три типа функциональных блоков включают: таймер задержки включения, таймер задержки выключения и сохраняющий таймер задержки включения (также вызывается сторожевой таймер). Имеется четыре контакта временных меток: 0.01 секунды, 0.1 секунды, 1.0 секунда и 1 минута

Таймер времени работы

Таймер времени работы использует 100-микросекундные метки для учета времени, прошедшего с момента включения питания для модуля ЦП. Этот таймер не сохраняет свое значение в случае отключения питания; его перезапуск происходит при каждом включении модуля ЦП. Каждую секунду происходит аппаратное прерывание работы ЦП для регистрации секундных меток времени. Эти секундные метки накапливаются в регистре, который переполняется приблизительно через 100 лет, после чего таймер начинает считать сначала.

Так как с таймером времени работы связана правильная работа операционной системы и функциональных блоков таймеров, то он не может быть сброшен пользовательской программой или программистом. Тем не менее, пользовательская программа может считывать текущее значение таймера времени работы, используя сервисный запрос #16.

Таймер астрономического времени

Этот таймер в 28-канальном ПЛК Micro и в ПЛК Series 90-30 (модель 331 и старше) поддерживается на аппаратном уровне. Таймер астрономического времени поддерживает семь функций измерения времени:

- Год (две цифры)
- Месяц
- День
- Час
- Минута
- Секунда
- День недели

Этот таймер питается от батареи и сохраняет значение времени при отключении питания. Однако? пока вы не инициализировали этот таймер, его значение не определено. Программа пользователя может считывать и устанавливать значение таймера, используя сервисный запрос #7. Также, значение этого таймера можно считывать и устанавливать с помощью конфигурационного ПО. Примечательно, что Ручной Программатор не позволит Вам изменить значение данного таймера, если включена защита ключом на модуле ЦП. Таймер астрономического времени предназначен для управления переходом из месяца в месяц и из года в год. Поправка на високосный год выполняется автоматически, пока не наступит 2079 год.

Сторожевой таймер

Сторожевой таймер в ПЛК Series 90-30 необходим для предотвращения катастрофических ситуаций, которые возникают вследствие недопустимо большого времени цикла. Уставка времени для сторожевого таймера составляет 200 миллисекунд (500 миллисекунд в моделях ЦП 35x и 36x); это фиксированное значение, которое не может быть изменено. Сторожевой таймер всегда стартует со значения ноль в начале каждого цикла.

Для моделей 331 и младше, если сторожевой таймер превышает значение уставки, светодиод OK гаснет; ЦП сбрасывается и выключается; выходы переходят в состояние по умолчанию. Прекращается связь ЦП со всем оборудованием, и все микропроцессоры на всех платах останавливаются. Для восстановления, необходимо вновь подать питание на стойку, содержащую ЦП. В ЦП Series 90-20, 90 Micro и Series 90-30 модели 340 и старше, срабатывание сторожевого таймера приводит к сбросу модуля ЦП, выполнению процедуры запуска, генерации сообщения о срабатывании сторожевого таймера и переходу ПЛК в режим **STOP**.

Таймер времени простоя

Таймер времени простоя используется для определения времени, в течение которого ПЛК был выключен. Когда ПЛК выключен, этот таймер сбрасывается в 0 и начинает отсчет времени. Соответственно, когда ПЛК включается, таймер останавливается, и его значение запоминается. Сервисный запрос № 29, описанный в главе 12, дает возможность прочитать значение этого времени.

Примечание

Эта функция поддерживается только в модулях ЦП 331 или старше ПЛК Series 90-30.

Таймер постоянного цикла

Этот таймер контролирует длительность цикла, когда ПЛК Series 90-30 работает в режиме **CONSTANT SWEEP TIME**. В этом режиме каждый цикл выполняется за определенное время. Обычно, для большинства приложений, нет нужды устанавливать режим постоянного времени цикла. Заданное время цикла может быть установлено программистом, и может быть любым значением от 5 миллисекунд до значения сторожевого таймера (по умолчанию 100 миллисекунд).

Если время цикла истекает прежде, чем цикл окончится, и предыдущий цикл не превысил значение постоянной времени, то ПЛК размещает в таблице сбоев соответствующее сообщение об ошибке. С началом следующего цикла, ПЛК устанавливает ссылку OV_SWP в 1. Эта ссылка обращается в 0, если ПЛК находится не в режиме **CONSTANT SWEEP TIME** или время последнего цикла не было превышено.

Контакты временных меток

В ПЛК Series 90 имеется четыре контакта (дискретных сигнала) для временных меток, которые работают с gthbjljv 0.01 секунда, 0.1 секунда, 1.0 секунда и 1 минута. Состояния этих контактов нельзя изменить в течение выполнения цикла. Эти контакты обеспечивают импульсы, определенной ширины. Контакты сохраняют свое значение в ссылках T_10MS (0.01 секунды), T_100MS (0.1 секунды), T_SEC (1.0 секунда), и T_MIN (1 минута).

Следующая временная диаграмма представляет периоды включения/выключения этих контактов.

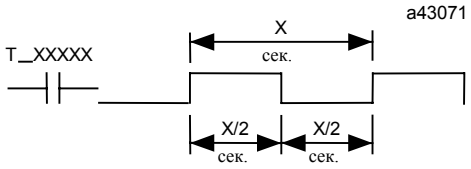


Рисунок 2-6. Временная диаграмма контактов временных меток

Раздел 5: Разграничение доступа к системе

Разграничение доступа в ПЛК Series 90-30, Series 90-20 и Micro необходимо для предотвращения несанкционированного изменения данных в ПЛК. Существует четыре уровня доступа к ПЛК. Первый уровень доступен всегда, он дает право на чтение данных ПЛК, но не разрешает изменять пользовательскую программу. Другие три уровня предоставляют доступ только после ввода пароля, который определяется для каждого уровня.

Каждый следующий уровень доступа дает больше прав, чем предыдущий. Каждый последующий уровень сочетает в себе права, определенные для данного уровня плюс права и возможности предыдущих уровней. Уровни и их возможности описаны ниже:

Уровень доступа	Права и возможности
Уровень 1	На данном уровне могут быть прочитаны любые данные, исключая пароли. В эти данные входят все данные, расположенные в памяти (%I, %Q, %AQ, %R, и прочие), таблицы сбоев, и все типы программных блоков (данные, значения и константы). Изменить какие-либо значения в ПЛК нельзя.
Уровень 2	Данный уровень позволяет производить запись данных в память (%I, %R, и прочие).
Уровень 3	Данный уровень позволяет производить запись данных в пользовательскую программу только в режиме **STOP**.
Уровень 4	Данный уровень является установленным по умолчанию для систем, которые не имеют установленного пароля. Этот уровень является установленным по умолчанию для систем с паролями, назначенными для самого низшего уровня. Данный уровень является самым высоким и позволяет производить чтение и запись данных в любую область памяти в двух режимах **RUN** и **STOP**. (Конфигурация не может быть изменена в режиме **RUN**)

Пароли

Для каждого уровня доступа назначается один пароль. (Отсутствие пароля возможно только для первого уровня). Каждый пароль может быть уникальным; однако, допускается использование одинаковых паролей для различных уровней доступа. Пароль представляют собой четыре ASCII символа; они могут быть введены и изменены только с помощью ПО для программирования или Ручного Программатора.

Уровень доступа действителен только на время сеанса связи между ПЛК и программатором. При этом может не выполняться никаких действий, однако связь не должна прерываться. Если связь прерывается более чем на 15 минут, автоматически устанавливается наивысший из незащищенных паролем уровней доступа.

После установления связи с ПЛК, ПО для программирования производит запрос статуса защиты для каждого уровня доступа к ПЛК. Затем ПО автоматически устанавливает наивысший из незащищенных паролем уровней доступа. Таким образом, ПО получает доступ к наивысшему незащищенному паролем уровню доступа без дополнительных запросов. При подключении ручного программатора к ПЛК, автоматически устанавливается наивысший из незащищенных паролем уровней доступа.

Запросы на изменение уровня доступа

Программист может изменить уровень доступа к ПЛК, выбрав необходимый ему уровень и введя требуемый пароль. Текущий уровень не изменяется, если пароль, введенный программистом, не совпадает с паролем, находящимся в таблице паролей, сохраненной в ПЛК. Если вы попытаетесь получить доступ к информации расположенной в ПЛК или модифицировать ее, используя Ручной Программатор без соответствующего уровня доступа, то Программатор сгенерирует сообщение об ошибке.

Блокирование/разблокирование подпрограмм

Блоки подпрограмм могут быть заблокированы и разблокированы с использованием ПО для программирования. Существуют два типа блокировки:

Тип блокировки	Описание
View	Однажды блокировав, вы не можете просматривать подпрограмму.
Edit	Однажды блокировав, вы не можете редактировать подпрограмму.

Ранее заблокированные подпрограммы можно разблокировать в редакторе, если блокировки не постоянные.

Функции поиска или поиска и замены могут быть произведены над заблокированной на просмотр подпрограммой. Если цель поиска найдена в заблокированных на просмотр подпрограммах, появиться одно из следующих предупреждающих сообщений:

```
Found in locked block <block_name>   (Continue/Quit)

Найден в заблокированном блоке <Имя_блока> (Продолжить/выход)
```

или

```
Cannot write to locked block <block_name>  (Continue/Quit)

Невозможно записать в заблокированный блок <Имя_блока> (Продолжить/выход)
```

Вы можете либо продолжить, либо прервать поиск.

Папки, которые содержат заблокированные подпрограммы, могут быть очищены или удалены. Если папка содержит заблокированные подпрограммы, то эти блоки остаются заблокированными, при выполнении над ними операций Copy, Backup и Restore.

Постоянно заблокированные подпрограммы

В дополнение к обычным блокировкам (VIEW LOCK и EDIT LOCK), существуют постоянные блокировки (PERMANENT LOCK). Если используется PERMANENT VIEW LOCK, то все действия внутри тела подпрограммы запрещены. Если используется PERMANENT EDIT LOCK, то невозможно редактировать подпрограмму.

Внимание

Постоянная блокировка отличается от обычной блокировки тем, что она не может быть удалена.

Однажды установив PERMANENT EDIT LOCK, Вы можете изменить ее только на PERMANENT VIEW LOCK. Установленную блокировку PERMANENT VIEW LOCK нельзя изменить на другой тип блокировки.

Раздел 6: Система В/В ПЛК Series 90-30, 90-20, и Micro

Система В/В ПЛК осуществляет взаимодействие между ПЛК 90-30 и периферийными устройствами. Модули В/В вставляются прямо в слот базовой платы ПЛК или в слоты на плате расширения для модели ЦП 331 или старше. Модели ЦП 331, 340 и 341 поддерживают до 49 модулей В/В (5 стоек). Модели 351 и 352 поддерживают до 79 модулей В/В (8 стоек). ПЛК Series 90-30 модели 311 или модели 313 с 5-слотовой базовой платой поддерживают до 5 модулей В/В; модель 323 с 10-слотовой базовой платой поддерживают до 10 модулей В/В.

Структура системы В/В для ПЛК 90-30 приведена ниже.

Система В/В ПЛК

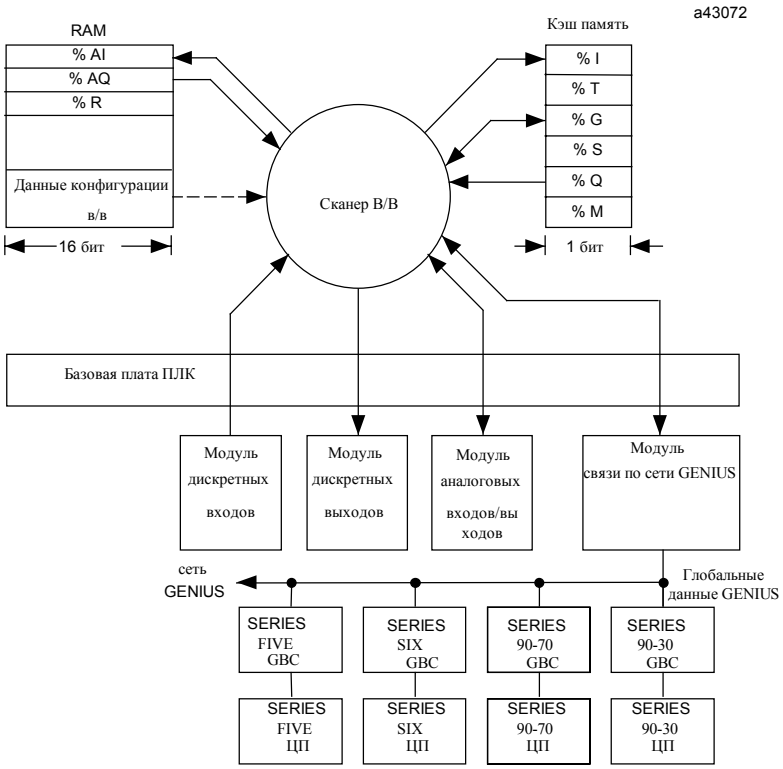


Рисунок 2-7. Структура системы В/В в ПЛК Series 90-30

Примечание

Диаграмма отображает вышесказанное, применительно к системе В/В ПЛК 90-30. интеллектуальные и дополнительные модули не представлены; они используются в окне связи с системой. Информацию относительно системы ввода/вывода ПЛК Series 90-20, смотрите в *Программируемые контроллеры Series 90™-20 Руководство пользователя* (GFK-0551). Информацию относительно системы В/В ПЛК 90-Micro, смотрите в *Программируемые контроллеры Series 90™ Micro Руководство пользователя* (GFK-1065).

Модули B/B Series 90-30

Существует пять типов модулей B/B для ПЛК Series 90-30: модули дискретного ввода, модули дискретного вывода, модули аналогового ввода, модули аналогового вывода и дополнительные модули. В следующей таблице представлены каталожные номера, число каналов B/B и краткое описание каждого модуля B/B Series 90-30.

Примечание

Все модули B/B, представленные ниже, доступны в то время, когда печаталось данное руководство. По вопросам применения консультируйтесь с вашим локальным дистрибьютором GE Fanuc или с коммерческим представительством GE. В руководстве *Series 90-30 I/O Module Specifications Manual*, GFK-0898, изложены технические характеристики и схемы подключения каждого модуля B/B Series 90-30.

Таблица 2-8. Модули B/B Series 90-30

Номер в каталоге	Каналов	Описание	Руководство
		Модули дискретного ввода	
IC693MDL230	8	Ввод 120 VAC, изолированные	GFK-0898
IC693MDL231	8	Ввод 240 VAC, изолированные	GFK-0898
IC693MDL240	16	Ввод 120 VAC	GFK-0898
IC693MDL241	16	Ввод 24 VAC/DC	GFK-0898
IC693MDL630	8	Ввод 24 VDC Положительная логика	GFK-0898
IC693MDL632	8	Ввод 125 VDC, положит./отриц. логика	GFK-0898
IC693MDL633	8	Ввод 24 VDC Отрицательная логика	GFK-0898
IC693MDL634	8	Ввод 24 VDC, положит./отриц. логика	GFK-0898
IC693MDL640	16	Ввод 24 VDC Положительная логика	GFK-0898
IC693MDL641	16	Ввод 24 VDC Отрицательная логика	GFK-0898
IC693MDL643	16	Ввод 24 VDC Положительная логика , FAST	GFK-0898
IC693MDL644	16	Ввод 24 VDC Отрицательная логика, FAST	GFK-0898
IC693MDL645	16	Ввод 24 VDC, положит./отриц. логика	GFK-0898
IC693MDL646	16	Ввод 24 VDC, положит./отриц. логика, FAST	GFK-0898
IC693MDL652	32	Ввод 24 VDC, положит./отриц. логика	GFK-0898
IC693MDL653	32	Ввод 24 VDC, положит./отриц. логика, FAST	GFK-0898
IC693MDL654	32	Ввод 5/12 VDC (TTL), положит./отриц логика	GFK-0898
IC693MDL655	32	Ввод 24 VDC, положит./отриц. логика	GFK-0898
IC693ACC300	8/16	Имитатор входов	GFK-0898

Таблица 2-9. Модули B/B Series 90-30 Продолжение

Номер в каталоге	Канал	Описание	Руководство
		Модули дискретного вывода	
IC693MDL310	12	Вывод 120 VAC, 0.5A	GFK-0898
IC693MDL330	8	Вывод 120/240 VAC, 2A	GFK-0898
IC693MDL340	16	Вывод 120 VAC, 0.5A	GFK-0898
IC693MDL390	5	Вывод 120/240 VAC Изолированный, 2A	GFK-0898
IC693MDL730	8	Вывод 12/24 VDC Положительная логика , 2A	GFK-0898
IC693MDL731	8	Вывод 12/24 VDC Отрицательная логика , 2A	GFK-0898
IC693MDL732	8	Вывод 12/24 VDC Положительная логика, 0.5A	GFK-0898
IC693MDL733	8	Вывод 12/24 VDC Отрицательная логика, 0.5A	GFK-0898
IC693MDL734	6	Вывод 125 VDC Положит./отрицательная логика, 2A	GFK-0898
IC693MDL740	16	Вывод 12/24 VDC Положительная логика, 0.5A	GFK-0898
IC693MDL741	16	Вывод 12/24 VDC Отрицательная логика, 0.5A	GFK-0898
IC693MDL742	16	Вывод 12/24 VDC Положительная логика, 1A	GFK-0898
IC693MDL750	32	Вывод 12/24 VDC Отрицательная логика	GFK-0898
IC693MDL751	32	Вывод 12/24 VDC Положительная логика, 0.3A	GFK-0898
IC693MDL752	32	Вывод 5/24 VDC (TTL) Отрицательная логика, 0.5A	GFK-0898
IC693MDL753	32	Вывод 12/24 VDC Положит./отрицательная логика, 0.5A	GFK-0898
IC693MDL930	8	Реле, N.O., 4A, Изолированный	GFK-0898
IC693MDL931	8	Реле, BC, Изолированный	GFK-0898
IC693MDL940	16	Реле, N.O., 2A	GFK-0898
		Модули B/B	
IC693MDR390	8/8	24 VDC вводов, релейные выводы	GFK-0898
IC693MAR590	8/8	120 VAC вводов, релейные выводы	GFK-0898
		Аналоговые модули	
IC693ALG220	4	Аналоговый ввод, Напряжение	GFK-0898
IC693ALG221	4	Аналоговый ввод, Ток	GFK-0898
IC693ALG222	16	Аналоговый ввод, Напряжение	GFK-0898
IC693ALG223	16	Аналоговый ввод, Ток	GFK-0898
IC693ALG390	2	Аналоговый вывод, Напряжение	GFK-0898
IC693ALG391	2	Аналоговый вывод, Ток	GFK-0898
IC693ALG392	8	Аналоговый вывод, Ток/ Напряжение	GFK-0898
IC693ALG442	4/2	Аналоговый модуль B/B, Напряжение/Ток	GFK-0898

Таблица 2-10. Модули B/B Series 90-30 Продолжение

Номер в каталоге	Описание	Руководство
	Дополнительные модули	
IC693APU300	Высокоскоростной счетчик	GFK-0293
IC693APU301	Модуль Power Mate APM, 1-ось–Режим слежения	GFK-0781
IC693APU301	Модуль Power Mate APM, 1-ось–Стандартный режим	GFK-0840
IC693APU302	Модуль Power Mate APM, 2-оси–Режим слежения	GFK-0781
IC693APU302	Модуль Power Mate APM, 2-оси–Стандартный режим	GFK-0840
IC693MCS001/2	Модуль Power Mate J Motion Control System (1 и 2 оси)	GFK-1256
IC693APU305	Процессор B/B	GFK-1028
IC693CMM321	Модуль интерфейса Ethernet	GFK-1084
IC693ADC311	Сопроцессор алфавитно-цифрового дисплея	GFK-0521
IC693BEM331	Контроллер сети Genius	GFK-1034
IC693BEM320	Модуль интерфейса I/O Link (slave)	GFK-0631
IC693BEM321	Модуль интерфейса I/O Link (master)	GFK-0823
IC693CMM311	Коммуникационный сопроцессор	GFK-0582
IC693CMM301	Коммуникационный модуль Genius	GFK-0412
IC693CMM302	Улучшенный коммуникационный модуль Genius	GFK-0695
IC693PCM300	PCM, 160K Bytes (35Кбайт User MegaBasic Program)	GFK-0255
IC693PCM301	PCM, 192K Bytes (47Кбайт User MegaBasic Program)	GFK-0255
IC693PCM311	PCM, 640K Bytes (190Кбайт User MegaBasic Program)	GFK-0255

Формат данных B/B

Дискретные входные и выходные сигналы запоминаются, как биты в кэш памяти. Информация аналоговых входов и выходов хранится как слова в области памяти RAM, специально выделенной для этого.

Условия по умолчанию для модулей вывода

При включении каналы вывода дискретных модулей Series 90-30 устанавливаются в состояние «выключено». Они сохраняют заданную по умолчанию конфигурацию выходов до первого сканирования выходов ПЛК. Модули аналогового вывода могут быть сконфигурированы с помощью перемычек на съемном клеммнике модуля в любой момент. Модули аналогового вывода могут питаться от внешнего источника, и продолжать работу по установленной конфигурации даже при отсутствии питания на ПЛК.

Диагностические данные

Диагностика доступна через биты в памяти %S, которые служат индикацией неисправности модуля B/B или несоответствия в конфигурации B/B. Диагностическая информация не доступна для индивидуальных каналов B/B. Более подробно информация по обработке ошибок изложена в главе 3 “Сбои и их исправление”.

Глобальные данные (Global Data)

Genius Global Data

ПЛК Series 90-30 поддерживают обмен данными между ЦП, используя глобальные данные Genius (Genius Global Data). Контроллер шины Genius IC693BEM331 версии 5 и старше, а также улучшенный модуль связи Genius IC693CMM302 могут публиковать до 128 байт данных своего ПЛК для чтения другими ПЛК или компьютером. Они также могут получать до 128 байт данных от каждого из 30 других узлов сети Genius. Данные могут быть переданы или получены в любой тип памяти, не только в память %G. Обычный модуль связи Genius, IC693CMM301, ограничен установленными в %G адресами и может обмениваться только 32 битами данных с узлами, имеющими сетевой адрес (SBA#) от 16 до 23. Этот модуль не может быть использован, как улучшенный GCM (Genius Controller Module), который обеспечивает более чем в 100 раз большую производительность.

Глобальными данными могут обмениваться только контроллеры Series 5, Series 6 и Series 90, подключенные к одному сегменту шины Genius.

Ethernet Global Data

Модуль ЦП 364 (версии 9.0 и старше) поддерживает подключение к сети Ethernet через любой из двух (но не оба) встроенных Ethernet-разъема. Подключение возможно через разъемы AAUI и 10BaseT. Только ЦП модели 364 (9.10 или старше) Series 90-30 поддерживает Ethernet Global Data (EGD).

Модель ЦП 364 поддерживает Ethernet Global Data, которые подобно глобальным данным Genius позволяют одному устройству передавать данные на одно или несколько других устройств в сети. EGD не поддерживается программным обеспечением Logicmaster 90 (необходим Windows-совместимый пакет для программирования ПЛК).

Модули В/В ПЛК Series 90-20

Следующие модули В/В доступны для использования в ПЛК Series 90-20. Каждый модуль, представленный в таблице, имеет каталожный номер, количество каналов В/В и краткое описание. Модули В/В устанавливаются на базовой плате с блоком питания. Для получения технической информации и схем подключения каждого модуля, обратитесь к руководству *Программируемые контроллеры Series 90-20 Руководство пользователя*, GFK-0551.

Номер в каталоге	Краткое описание	Каналы
IC692MAA541	базовый модуль питания и В/В ввод 120 VAC / вывод 120 VAC / питание 120 VAC	16 Вх/12 Вых
IC692MDR541	базовый модуль питания и В/В ввод 24 VDC / вывод реле / питание 120 VAC	16 Вх/12 Вых
IC692MDR741	базовый модуль питания и В/В ввод 24 VDC / вывод реле / питание 240 VAC	16 Вх/12 Вых

IC692CPU211	Модуль ЦП, Модель 211	-

Конфигурирование и программирование

Конфигурирование – это процесс назначения, как адресов, так и других характеристик, модулям в стойке. Процесс конфигурирования можно выполнить в любое время, перед программированием или после него, используя ПО для программирования или ручной программатор; тем не менее, конфигурирование рекомендуется делать первым. Если этого не было сделано, обратитесь к руководству *Программное обеспечение Logicmaster 90-30/20/Micro Руководство пользователя*, GFK-0466, чтобы решить можно ли начинать программирование.

Программирование состоит в создании прикладной программы для ПЛК. Так как контроллеры Series 90-30 и 90-20 и 90 Micro имеют одинаковую систему команд, то все три серии ПЛК могут быть запрограммированы с использованием пакета для программирования Logicmaster 90-30. Главы с 4 по 12 данного руководства описывают программные инструкции, которые могут быть использованы при написании программы для ПЛК Series 90-20 и 90-30.

Если программное обеспечение Logicmaster 90-30/20/Micro еще не установлено, пожалуйста, обратитесь к руководству GFK-0466, для получения подробных инструкций по установке. Это руководство также объясняет, как создавать, редактировать, загружать и печатать программы.

Глава
3

Сбои и их устранение

Эта глава описывает процедуру поиска неисправностей, возникающих в системах управления на базе ПЛК Series 90-30, 90-20 и Micro. В ней раскрываются описания сбоев, которые фиксируются в таблице сбоев ПЛК (PLC fault table), и категории сбоев, которые фиксируются в таблице сбоев системы ввода/вывода (I/O fault table).

Каждое разъяснение сбоя в данной главе содержит описание сбоя в таблице сбоев ПЛК или категорию сбоя в таблице сбоев системы ввода/вывода. Найдите в данной главе описание сбоя (категорию сбоя), совпадающее с отображенным на экране программатора описанием (категорией) сбоя. Описание сбоя содержит причину его возникновения и инструкции по устранению.

Глава 3 содержит следующие разделы:

Раздел 1: Обработка сбоев

Примечание

Эта информация по обработке сбоев применяется к системам программирования Logicmaster 90-30/20/Micro.

Сбои в системах на базе ПЛК 90-30, 90-20 или 90 Micro происходят при возникновении условий, изменяющих параметры их работы. Эти условия могут дать как к отрицательный, так и положительный эффект: например при потере модуля В/В или стойки, ПЛК может лишится способности управлять агрегатом или процессом, а при добавлении нового модуля в режиме online появится возможность его немедленного использования. Также эти условия могут действовать как предупреждающие, например, предупреждать о низком уровне напряжения на батарее, которая защищает память, и она должна быть заменена.

Обработчик тревог

Условия, изменяющие параметры работы системы называются сбоями. Если сбой произошел и был обработан ЦП, он называется тревогой. Программное обеспечение в ЦП, которое обрабатывает эти условия, называется Обработчик Тревог. Интерфейс пользователя с Обработчиком Тревог осуществляется посредством программного обеспечения для программирования. Все обнаруженные сбои записываются в таблицу сбоев и отображаются либо в таблице сбоев ПЛК, либо в таблице сбоев В/В, по принадлежности сбоя.

Классы сбоев

ПЛК Series 90-30, 90-20 и Micro определяет несколько классов сбоев. Они включают в себя внутренние сбои, внешние сбои и оперативные сбои.

Класс сбоя	Пример
Внутренние сбои	Нет ответа от модуля. Низкое напряжение батареи. Ошибка контрольной суммы.
Внешние сбои В/В	Потеря стойки или модуля. Добавление модуля или стойки.
Оперативные сбои	Сбой связи. Ошибка конфигурации. Отказ в доступе.

Примечание

Для получения информации по ПЛК Series 90 Micro, обратитесь к *ПЛК Series 90 Micro Руководство пользователя* (GFK-1065).

Реакция системы на сбой

Сбои аппаратного обеспечения требуют выключения системы, в тех случаях, когда сбой не допустим. Собственно система ПЛК допускает сбои модулей В/В, но для прикладной программы или контролируемого процесса они не допустимы. Оперативные сбои обычно являются допустимыми. Сбои ПЛК Series 90-30, 90-20 и Micro имеют два атрибута:

Атрибут	Описание
Принадлежность к таблице сбоев	Таблица сбоев В/В Таблица сбоев ПЛК
Действие сбоя	Фатальное Диагностическое Информационное

Таблицы сбоев

В ПЛК поддерживаются две таблицы сбоев для регистрации сбоев: таблица сбоев В/В для регистрации сбоев, происходящих в системе В/В ПЛК, и таблица сбоев ПЛК для регистрации всех остальных сбоев. Таблица 3-1 содержит: группы сбоев, действие сбоев, в какую таблицу занесена запись о сбое, и “имя” системной переменной в памяти %S.

Таблица 3-1. Общая классификация сбоев

Группа сбоев	Действие сбоя	Таблица сбоев	Имя системной переменной			
Потеря или отсутствие модуля В/В	Диагностическое	В/В	io_flt	any_flt	io_pres	los_iom
Потеря или отсутствие дополнительного модуля	Диагностическое	ПЛК	sy_flt	any_flt	sy_pres	los_sio
Несоответствие системной конфигурации	Фатальное	ПЛК	sy_flt	any_flt	sy_pres	cfg_mm
Сбой конфигурации ПЛК	Фатальное	ПЛК	sy_flt	any_flt	sy_pres	hrd_cpu
Ошибка контрольной суммы программы	Фатальное	ПЛК	sy_flt	any_flt	sy_pres	pb_sum
Низкое напряжение батареи	Диагностическое	ПЛК	sy_flt	any_flt	sy_pres	low_bat
Заполнена таблица сбоев ПЛК	Диагностическое	—	sy_full			
Заполнена таблица сбоев модулей В/В	Диагностическое	—	io_full			
Ошибка приложения	Диагностическое	ПЛК	sy_flt	any_flt	sy_pres	apl_flt
Нет пользовательской программы	Информационное	ПЛК	sy_flt	any_flt	sy_pres	no_prog
Повреждена память	Фатальное	ПЛК	sy_flt	any_flt	sy_pres	bad_ram
Отказ в доступе; неправильный пароль	Диагностическое	ПЛК	sy_flt	any_flt	sy_pres	bad_pwd
Сбой ПО ПЛК	Фатальное	ПЛК	sy_flt	any_flt	sy_pres	sft_cpu
Сбой при записи в ПЛК	Фатальное	ПЛК	sy_flt	any_flt	sy_pres	stor_er
Превышено время цикла	Диагностическое	ПЛК	sy_flt	any_flt	sy_pres	ov_swp
Неизвестный сбой ПЛК	Фатальное	ПЛК	sy_flt	any_flt	sy_pres	
Неизвестный сбой модуля В/В	Фатальное	В/В	io_flt	any_flt	io_pres	

Действия сбоев

Сбои бывают фатальные, диагностические и информационные.

При возникновении фатального сбоя происходит его регистрация в соответствующей таблице, устанавливаются диагностические переменные, и система останавливается. При диагностических сбоях происходит регистрация в соответствующей таблице, и устанавливаются диагностические переменные. При информационных сбоях происходит только регистрация.

Все возможные действия сбоев приведены в следующей таблице.

Таблица 3-2. Действия сбоев

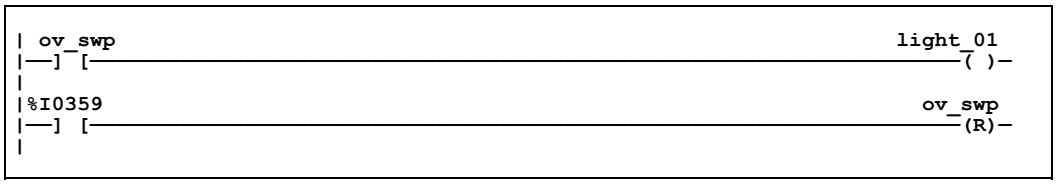
Действие сбоя	Реакция CPU
Фатальное	Запись сбоя в таблицу. Установка ссылок на сбой. Переход в режим **STOP**.
Диагностическое	Запись сбоя в таблицу. Установка ссылок на сбой.
Информационное	Запись сбоя в таблицу.

Когда обнаружен сбой, ЦП выполняет специальные действия по ликвидации этого сбоя. Действия по ликвидации сбоев не требуют дополнительной настройки в ПЛК Series 90-30, 90-20, или 90 Micro.

Ссылки на сбои

Все ссылки на сбои в ПЛК Series 90-30 – это итоговые ссылки одного типа. Их назначение – показать, какой сбой произошел. Ссылки на сбои сохраняются до тех пор, пока ПЛК не будет очищен или до тех пор, пока они не будут сброшены программой пользователя.

Ниже приведен пример установки и сброса бита, ссылающегося на сбой. Ячейка light_01 устанавливается в 1, когда выполнится условие OV_SWP; ячейка light_01 и переменная OV_SWP обращены в 1 до тех пор, пока не будет замкнут контакт %I0359.



Общие сведения о ссылках на сбои

Обработчик тревог формирует состояния 128 системных дискретных бит в памяти %S. Эти ссылки на сбои могут быть использованы для указания, где произошел сбой и к какому типу он относится. Ссылки на сбои располагаются в памяти %S, %SA, %SB, и %SC, и все они

имеют имена. Эти ссылки доступны для использования в прикладной программе как инициаторы запроса. Смотрите главу 2, «Принципы работы системы» для получения списка системных ссылок.

Дополнительные эффекты сбоев

Два сбоя, описанные выше, могут оказывать дополнительное воздействие на систему. Эти воздействия описаны в следующей таблице.

Побочное воздействие	Описание
Ошибка программного обеспечения ЦП ПЛК	Когда регистрируется сбой программы ЦП, ПЛК Series 90-30 или 90-20 немедленно переходит в специальный режим **ERROR SWEEP**. Для выхода из этого режима существует только один метод: отключение и повторное включение питания ПЛК.
Ошибка при записи в ПЛК	В течение процесса записи (запись блоков программы и других данных начинается со специальной команды начала записи и заканчивается командой конца записи), если связь с программирующим устройством, выполняющим запись данных, прервана или запись прекратилась по причине любой другой ошибки, регистрируется этот сбой. До тех пор пока этот сбой присутствует в системе, контроллер не может быть переведен в режим **RUN**.

Отображение таблицы сбоев ПЛК

Экран таблицы сбоев ПЛК отображает поизошедшие в ПЛК сбои. Например: ввод некорректного пароля, несоответствие аппаратного обеспечения контроллера его конфигурации, ошибки четности и коммуникационные ошибки.

ПО для программирования ПЛК, может отображать эту таблицу в любом режиме. Если ПО находится в режиме **OFFLINE**, сбои в таблице не отображаются. В режиме **ONLINE** или **MONITOR**, сбои отображаются в таблице. В режиме **ONLINE** таблица сбоев может быть очищена (выполнение этой процедуры ограничивается паролем).

После очистки таблицы, те сбои, которые по-прежнему присутствуют в системе, заново не регистрируются в ней (например, сбой “Low Battery”).

Отображение таблицы сбоев В/В

Таблица сбоев В/В отображает на экране сбои модулей В/В, такие как ошибки схемы подключения, конфликты адресов, аварии схемы, и отказы шины В/В.

ПО для программирования ПЛК, может отображать эту таблицу в любом режиме. Если ПО находится в режиме **OFFLINE**, сбои в таблице не отображаются. В режиме **ONLINE** или **MONITOR**, сбои отображаются в таблице. В режиме **ONLINE** таблица сбоев может быть очищена (выполнение этой процедуры ограничивается паролем). После очистки таблице, те сбои, которые по-прежнему присутствуют в системе, заново не регистрируются в ней.

Доступ к дополнительной информации о сбое

Таблицы сбоев содержат только базовую информацию относительно произошедшего сбоя. Дополнительная информация по каждому сбою может быть получена с помощью ПО для программирования ПЛК. Кроме того, ПО обеспечивает представление кода сбоя в шестнадцатеричном формате.

Последняя запись, «Correction», предназначена для разъяснения действий необходимых для устранения каждого сбоя. Заметьте, что действия по устранению некоторых сбоев содержат в себе следующую фразу:

Display the PLC Fault Table on the Programmer. Contact GE Fanuc Field Service, giving them all the information contained in the fault entry.

Второе предложение означает, что вы должны предоставить Службе Поддержки (Field Service) как текстовую запись о сбое, взятую непосредственно из таблицы, так и шестнадцатеричный код этого сбоя. Для устранения сбоя в дальнейшем следуйте инструкциям Службы Поддержки.

Раздел 2: Объяснение таблицы сбоев ПЛК

Каждое объяснение сбоя содержит описание сбоя и инструкции по его устранению. Во многих случаях описания сбоев, регистрируемых в таблице, совпадают, но имеют различные причины возникновения. В связи с этим, код ошибки, отображаемый в разделе дополнительной информации по сбою, используется для распознавания сбоев, в ситуации, когда описания сбоев одинаковы. Код ошибки – это одна или две шестнадцатеричных цифры в пятой группе чисел – показан в нижеследующем примере.

```
01   000000   01030100   0902   0200   000000000000
                                 |
                                 |_____  Код сбоя   (первые две 16-тиричные
                                                     цифры в пятой группе)
```

Некоторые сбои могут происходить из-за ошибок произвольного доступа к памяти ЦП. Причиной таких сбоев может быть непредвиденное выключение питания системы при низком заряде батареи, которого не хватит для поддержки питания памяти. Чтобы избежать чрезмерного дублирования инструкций, когда испорченная память может стать причиной ошибки, устранение сопровождается следующим текстом:

Perform the corrections for Corrupted Memory.

Это означает:

1. Если система выключена, замените батарею. Напряжение батареи могло стать недостаточным для поддержания памяти.
2. Замените модуль ЦП ПЛК, так как он мог быть поврежден.

Следующая таблица поможет вам быстро найти в данном тексте детальное объяснение сбоев. Каждый пункт представлен так, как Вы его видите на экране программатора.

Действия по исправлению сбоев

При возникновении **фатальных сбоев** ПЛК переходит в режим **STOP** в конце того цикла, который вызвал этот сбой. **Диагностические** сбои регистрируются, и устанавливаются диагностические переменные. **Информационные** сбои только регистрируется в таблице сбоев ПЛК.

Потеря или отсутствие дополнительного модуля

Сбои данной группы возникают, когда модули PCM, CMM, или ADC не отвечают на запрос ЦП. Ошибка появляется после включения питания, если модуль отсутствует, или во время работы, если модуль не отвечает. Это **диагностический** сбой.

Код ошибки:	1, 42
Наименование:	Option Module Soft Reset Failed (Сбой при перезапуске модуля)
Описание:	ЦП ПЛК не имеет возможности перезапустить процедуру коммуникации.
Устранение:	(1) Повторите попытку сброса через некоторое время. (2) Замените модуль. (3) Выключите систему. Проверьте, как модуль PCM размещен в стойке и все ли кабеля подключены. (4) Замените кабели.
Код ошибки:	Все остальные
Наименование:	Module Failure During Configuration (Сбой модуля во время конфигурации)
Описание:	ПО ПЛК генерирует эту ошибку, когда происходит сбой модуля при включении или при сохранении конфигурации.
Устранение:	(1) Выключите систему. Измените месторасположение модуля в стойке.

Сброс, добавление или появление дополнительного модуля

Сбои группы **Reset of, Addition of, or Extra Option Module** возникают, когда дополнительный модуль (PCM, ADC и тому подобные) находясь в режиме online, сбрасывается, или, находясь в стойке, не имеет правильной конфигурации. Данный сбой относиться к группе **диагностических**. Три байта из данных выделенных для этого сбоя обеспечивают дополнительную информацию о сбое.

Устранение:	(1) Обновите конфигурацию данного модуля. (2) Удалите модуль из системы.

Несоответствие системной конфигурации

Сбои группы **Configuration Mismatch** возникают, когда модуль, находящийся в слоте не соответствует настройками, указанным в конфигурации. Это **фатальный** сбой.

Код ошибки:	1
Наименование:	System Configuration Mismatch
Описание:	ПО ПЛК (системный конфигуратор) генерирует этот сбой, когда модуль, находящийся в слоте, не соответствует модулю, который должен находится в этом слоте согласно конфигурационному файлу.
Устранение:	Установите несоответствие и переконфигурируйте модуль или стойку.
Код ошибки:	6
Наименование:	System Configuration Mismatch
Описание:	Это та же ошибка, что и с кодом 1; ПО ПЛК генерирует этот сбой, когда модуль, находящийся в слоте, не соответствует модулю, который должен находится в этом слоте согласно конфигурационному файлу.
Устранение:	Установите несоответствие и переконфигурируйте модуль или стойку.
Код ошибки:	18
Наименование:	Unsupported Hardware
Описание:	Присутствие PCM или подобного модуля в стойках с ЦП модели 311, 313 или 323.
Устранение:	Физически ликвидировать эту ситуацию можно удалением PCM или подобного модуля или установкой модуля ЦП, который поддерживает данные модули.
Код ошибки:	26
Наименование:	Module busy–config not yet accept by module
Описание:	Модуль не может принять новую конфигурацию в данное время, потому что занят обработкой другого процесса.
Устранение:	Разрешите модулю завершить текущую операцию и сохраните конфигурацию еще раз.
Код ошибки:	51
Наименование:	END Function Executed from SFC Action
Описание:	Данный сбой происходит при размещении функции END в блоке SFC или в блоке, вызываемом блоком SFC.
Устранение:	Удалите функцию END из блока SFC или из блока, который вызывается блоком SFC.

Сбой ПО дополнительного модуля

Сбои группы **Option Module Software Failure** возникают, когда в модулях PCM или ADC происходит ошибка невозможности восстановления ПО модуля. Эта группа сбоев относится к **фатальным сбоям**.

Код ошибки:	Любой
Наименование:	COMMREQ Frequency Too High
Описание:	Функция COMMREQs посылает данные быстрее, чем процесс может их обработать.
Устранение:	Измените программу в ПЛК.

Ошибка при проверке контрольной суммы программы

Сбои группы **Program Block Checksum Failure** возникают, в случае обнаружения ЦП ПЛК ошибочных условий в полученной контроллером программе. Также данный сбой возникает, если ЦП ПЛК обнаруживает несоответствие контрольной суммы при проверке памяти во время включения контроллера или во время работы контроллера. Эта группа относится к **фатальным сбоям**.

Код ошибки:	Любой
Наименование:	Program Block Checksum Failure
Описание:	ПО ПЛК генерирует эту ошибку, в случае если программа повреждена.
Устранение:	(1) Очистите память ПЛК и запишите программу заново. (2) Свяжитесь с центром поддержки GE Fanuc и получите более подробную информацию.

Сигнал низкого напряжения на батарее

Сбои группы **Low Battery Signal** возникают, когда ЦП ПЛК регистрирует низкое напряжение в блоке питания контроллера или в самом модуле, например в PCM. Этот сбой принадлежит группе **диагностических**.

Код ошибки:	0
Имя:	Failed Battery Signal
Описание:	Модуль ЦП (или другой модуль с батареей) имеет севшую батарею.
Устранение:	Замените батарею. Не вынимайте блок питания из стойки.
Код ошибки:	1
Имя:	Low Battery Signal
Описание:	Батарея в ЦП или в другом модуле имеет пониженное напряжение.
Устранение:	Замените батарею. Не вынимайте блок питания из стойки.

Превышение установленного времени цикла ПЛК

Сбои группы **Constant Sweep Time Exceeded** возникают, когда ЦП ПЛК работает в режиме **CONSTANT SWEEP**, и время цикла ПЛК превысило установленную величину. Дополнительные данные сбоя, в первых двух байтах, содержат фактическое время цикла, а в следующих восьми – имя программы. Данный сбой относится к группе **диагностических**.

Устранение:	(1)	Увеличьте значение постоянной времени цикла.
	(2)	Удалите часть блоков из программы.

Сбой приложения

Сбои группы **Application Fault** возникают в случае, если ЦП ПЛК обнаруживает сбой в пользовательской программе. Данный сбой относится к группе **диагностических**, за исключением тех ошибок, которые вызваны переполнением стека подпрограмм, которые отнесены к группе **фатальных**.

Код ошибки:	7
Имя:	Subroutine Call Stack Exceeded
Описание:	Вызовы подпрограммы ограничены 8 уровнями. Подпрограмма может вызывать другую подпрограмму, которая в свою очередь вызывает следующую подпрограмму пока общее их количество не достигнет восьми.
Устранение:	Модифицируйте программу, таким образом, чтобы количество вложенных подпрограмм не превышало восьми
Код ошибки:	1B
Имя:	CommReq Not Processed Due To PLC Memory Limitations
Описание:	Коммуникационные запросы с нулевым временем ожидания поступают быстрее, чем могут быть обработаны (один запрос в течение одного цикла). В такой ситуации, когда коммуникационные запросы поступают на вход, для которого ПЛК выделил памяти меньше, чем необходимо, то такие запросы не обрабатываются.
Устранение:	Создайте несколько запросов или попробуете уменьшить требуемое запросу количество памяти.
Код ошибки:	5A
Имя:	User Shut Down Requested
Описание:	Системное ПО ПЛК (функциональные блоки) генерирует это информационную тревогу, когда сервисный запрос №13 (User Shut Down) выполняется в программе.
Устранение:	Исправления не требуется. Это информационная тревога.

Отсутствие пользовательской программы

Сбои группы **No User Program Present** возникают, когда ЦП ПЛК отдает команду на перевод ПЛК из режима **STOP** в режим **RUN** *или* на запись в ПЛК, при отсутствии пользовательской программы в памяти ПЛК. ЦП ПЛК обнаруживает отсутствие пользовательской программы во время включения контроллера. Данный сбой отнесется к группе **информационных**.

Устранение:	Загрузите пользовательскую программу перед переводом ПЛК в режим **RUN**.

Нарушение целостности пользовательской программы при включении

Сбои группы **Corrupted User Program on Power-Up** возникают, если ЦП ПЛК обнаруживает нарушение целостности пользовательской памяти (RAM). ЦП ПЛК перейдет в режим **STOP,** пока не будет заново загружена пользовательская программа и конфигурация. Данный сбой отнесется к группе **фатальных**.

Код ошибки:	1
Имя:	Corrupted User RAM on Power-Up
Описание:	ПО ПЛК генерирует эту ошибку, когда обнаруживает нарушение целостности пользовательской памяти при включении контроллера.
Устранение:	(1) Перезагрузите конфигурацию, пользовательскую программу и ссылки. (2) Замените батарею в модуле ЦП. (3) Замените карту расширенной памяти в модуле ЦП. (4) Замените модуль ЦП.
Код ошибки:	2
Имя:	Illegal Boolean OpCode Detected
Описание:	ПО ПЛК генерирует эту ошибку, когда обнаруживает неправильную инструкцию в пользовательской программе.
Устранение:	(1) Перезагрузите пользовательскую программу. (2) Замените карту расширенной памяти в модуле ЦП. (3) Замените модуль ЦП.

Неправильный пароль

Сбои группы **Password Access Failure** возникают, когда ЦП ПЛК получает запрос на изменение уровня привилегий и пароль, включенный в тело запроса, указан неверно для данного уровня. Данный сбой отнесен к группе **информационных**.

Устранение:	Повторите запрос с корректным паролем.

Ошибки системного программного обеспечения ЦП ПЛК

Сбои группы **PLC CPU System Software Failure** генерируются ПО ЦП ПЛК Series 90-30, 90-20 или Micro PLC. Они возникают во время выполнения различных системных операций. Если произошел **фатальный** сбой, ЦП ПЛК **незамедлительно** переходит в специальный режим – **ERROR SWEEP**. Запрещены любые действия, когда ПЛК находится в этом режиме. Существует только один способ ликвидации этого сбоя – выключить и вновь включить ПЛК. Данный сбой относится к группе **фатальных**.

Код ошибки:	с 1 по B
Имя:	User Memory Could Not Be Allocated
Описание:	Системное ПО ПЛК (менеджер памяти) генерирует эти ошибки, когда запрос, посылаемый ПО менеджеру памяти на выделение/освобождение блока памяти в пользовательской части ОЗУ, не разрешен. Эти ошибки не должны возникать в производственных системах!
Устранение:	Выведете таблицу сбоев ПЛК на дисплей программатора и свяжитесь с центром технической поддержки GE Fanuc для передачи им всей информации по данному сбою.
Код ошибки:	D
Имя:	System Memory Unavailable
Описание:	Системное ПО ПЛК (сканнер B/B) генерирует эту ошибку, когда на его запрос блока системной памяти поступает отказ от менеджера памяти, по причине недостатка свободной памяти. Если ошибка произошла в группе инструкций DO I/O, то этот сбой – *информационный*. Если ошибка возникает при инициализации или автоконфигурации – это *фатальный* сбой.
Устранение:	Выведете таблицу сбоев ПЛК на дисплей программатора и свяжитесь с центром технической поддержки GE Fanuc для передачи им всей информации по данному сбою.
Код ошибки:	E
Имя:	System Memory Could Not Be Freed
Описание:	Системное ПО ПЛК (сканнер B/B) генерирует эту ошибку, если запрос на освобождение блока системной памяти не был исполнен. Этот сбой может возникнуть только во время выполнения блока инструкций DO I/O.
Устранение:	(1) Выведете таблицу сбоев ПЛК на дисплей программатора и свяжитесь с центром технической поддержки GE Fanuc для передачи им всей информации по данному сбою. (2) Выполните восстановление поврежденной памяти.
Код ошибки:	10
Имя:	Invalid Scan Request of the I/O Scanner
Описание:	Системное ПО ПЛК (сканнер B/B) генерирует эту ошибку, когда операционная система или функциональный блок DO I/O не производит ни полного ни частичного сканирования поступающих запросов. Эти ошибки не должны возникать в производственных системах!
Устранение:	Выведете таблицу сбоев ПЛК на дисплей программатора и свяжитесь с центром технической поддержки GE Fanuc для передачи им всей информации по данному сбою.
Код ошибки:	13
Имя:	PLC Operating Software Error
Описание:	Системное ПО ПЛК генерирует эту ошибку, когда в работе ПО ПЛК имеются проблемы. Эти ошибки не должны возникать в производственных системах!
Устранение:	(1) Выведете таблицу сбоев ПЛК на дисплей программатора и свяжитесь с центром технической поддержки GE Fanuc для передачи им всей информации по данному сбою. (2) Выполните восстановление поврежденной памяти.

Код ошибки:	14, 27
Имя:	Corrupted PLC Program Memory
Описание:	Системное ПО ПЛК генерирует эту ошибку, когда в работе ПО ПЛК имеются проблемы. Эти ошибки не должны возникать в производственных системах!
Устранение:	(1) Выведете таблицу сбоев ПЛК на дисплей программатора и свяжитесь с центром технической поддержки GE Fanuc для передачи им всей информации по данному сбою. (2) Выполните восстановление поврежденной памяти.
Код ошибки:	С 27 до 4E
Имя:	PLC Operating Software Error
Описание:	Системное ПО ПЛК генерирует эту ошибку, когда в работе ПО ПЛК имеются проблемы. Эти ошибки не должны возникать в производственных системах!
Устранение:	Выведете таблицу сбоев ПЛК на дисплей программатора и свяжитесь с центром технической поддержки GE Fanuc для передачи им всей информации по данному сбою.
Код ошибки:	4F
Имя:	Communications Failed
Описание:	Системное ПО ПЛК (обработчик сервисных запросов) генерирует эту ошибку, когда оно выполняет запрос, требующий фоновой коммуникации, но получает отказ.
Устранение:	(1) Проверьте шину на наличие неисправностей (2) Замените интеллектуальный модуль для связи напрямую.
Код ошибки:	50, 51, 53
Имя:	System Memory Errors
Описание:	Системное ПО ПЛК генерирует эти ошибки, когда его запрос на выделение памяти блокируется из-за того, что память вся занята или содержит ошибки.
Устранение:	(1) Выведете таблицу сбоев ПЛК на дисплей программатора и свяжитесь с центром технической поддержки GE Fanuc для передачи им всей информации по данному сбою. (2) Выполните восстановление поврежденной памяти.
Код ошибки:	52
Имя:	Backplane Communications Failed
Описание:	Системное ПО ПЛК (обработчик сервисных запросов) генерирует эту ошибку, когда оно выполняет запрос, требующий фоновой коммуникации, но получает отказ.
Устранение:	(1) Проверьте шину на наличие неисправностей (2) Замените интеллектуальный модуль для связи напрямую. (3) Проверьте правильность присоединения программатора.
Код ошибки:	Все остальные
Имя:	PLC CPU Internal System Error
Описание:	Это внутренняя системная ошибка, которая проявляется во всех остальных случаях. Эта ошибка не должна возникать в производственных системах.
Устранение:	Выведете таблицу сбоев ПЛК на дисплей программатора и свяжитесь с центром технической поддержки GE Fanuc для передачи им всей информации по данному сбою.

Ошибки связи во время записи в ПЛК

Сбои группы **Communications Failure During Store** возникают во время записи программных блоков и других данных в ПЛК. Последовательность команд для записи программных блоков и данных начинается со специальной команды начала записи и заканчивается командой конца записи. Если взаимосвязь с программирующим устройством, выполняющим запись данных, прервана или запись прекратилась по причине любой другой ошибки, этот сбой регистрируется. До тех пор пока этот сбой присутствует в системе, контроллер не может быть переведен в режим **RUN**.

Этот сбой не устраняется автоматически при включении питания; пользователь должен выполнить специальные действия для устранения этого сбоя. Данный сбой относится к группе **фатальных**.

Устранение:	Очистите таблицу сбоев и повторите попытку загрузки в ПЛК программы или конфигурации.

Раздел 3: Объяснение таблицы сбоев модулей B/B

Таблица сбоев B/B регистрирует данные о сбоях в трех классификациях:

- Категория сбоя.
- Тип сбоя.
- Описание сбоя.

Описываемые ниже сбои не имеют ни типа сбоя, ни группы.

Каждое объяснение сбоя содержит описание сбоя и инструкции по его устранению. Во многих случаях описания сбоев, регистрирующихся в таблице, имеют различные причины возникновения. В связи с этим, код ошибки, отображаемый в разделе дополнительной информации по сбою, доступ к которой можно получить, нажав CTRL-F, используется для распознавания сбоев, в ситуации, когда описания сбоев одинаковы (для получения более подробной информации относительно использования комбинации CTRL-F, смотрите приложение B, *«Интерпретация таблиц сбоев»* в этом руководстве). Код ошибки – это одна или две шестнадцатеричных цифры в пятой группе чисел – показан в нижеследующем примере.

```
02  1F0100  00030101FF7F  0302  0200   84000000000003
                                |
                                |_____  категория сбоя (первые две
                                  шестнадцатеричных цифры в пятой группе)
```

Следующая таблица поможет вам быстро найти в данном тексте детальное объяснение сбоев. Каждый пункт представлен так, как Вы его видите на экране программатора.

Потеря модуля B/B

Сбои категории **Loss of I/O Module** может появляться в модулях дискретного и аналогового B/B Series 90-30. Типов и описаний для сбоев данной категории не существует. Этот сбой является **диагностическим**.

Описание:	Системное ПО ПЛК генерирует эту ошибку, когда обнаруживает, что модуль не отвечает на команды от ЦП, или, когда конфигурация указывает на неправильное месторасположение модуля B/B в слоте.
Устранение:	(1) Замените модуль. (2) Исправьте конфигурацию. (3) Выведете таблицу сбоев ПЛК на дисплей программатора и свяжитесь с центром технической поддержки GE Fanuc для передачи им всей информации по данному сбою.

Добавление модуля B/B

Сбои категории **Addition of I/O Module** может появляться в модулях дискретного и аналогового B/B Series 90-30. Типов и описаний для сбоев данной категории не существует. Этот сбой является **диагностическим**.

Описание:	СПО ПЛК генерирует эту ошибку, когда неисправный модуль B/B, начинает снова работать.
Устранение:	(1) Если модуль был удален или заменен или было выключено/включено питание стойки, то ничего делать не требуется. (2) Обновите конфигурацию или удалите этот модуль.
Описание:	СПО ПЛК генерирует эту ошибку, когда обнаруживает модуль B/B 30-ой модели в слоте, хотя согласно конфигурации данный слот должен быть пустым.
Устранение:	(1) Удалите этот модуль. (2) Обновите или перезагрузите конфигурацию вместе с дополнительным модулем.

Глава 4 Функции реле

В настоящей главе поясняется, как использовать контакты, обмотки и связи в звеньях многозвенной логики.

Использование контактов

Контакт используется для контроля состояния ссылки. Пропустит контакт поток энергии или нет, зависит от состояния или статуса контролируемой ссылки и типа контакта. Ссылка обозначается ON, если находится в состоянии 1, и OFF, если находится в состоянии 0.

Таблица 4-1. Типы контактов

Тип контакта	Отображение	Контакт пропускает энергию направо
Нормально разомкнутый	—| |—	Когда ссылка установлена в ON
Нормально замкнутый	—|/|—	Когда ссылка установлена в OFF
Обмотка продолжения	<+>———	Когда предшествующая обмотка продолжения установлена в ON

Использование обмоток

Обмотки используются для управления дискретными ссылками. Для управления потоком энергии в обмотку должна использоваться условная логика. Обмотки вызывают действие непосредственно; они не пропускают поток энергии вправо. Если в программе должна быть выполнена дополнительная логика, являющаяся следствием состояния обмотки, то для этой обмотки должна быть внутренняя ссылка или должна использоваться комбинация обмотки/контакта продолжения.

Обмотки всегда расположены в крайней правой позиции логической линии. Звено может содержать до восьми обмоток.

Тип используемой обмотки зависит от типа желаемого действия программы. Состояния обмоток с удержанием сохраняются при отключении питания, или когда ПЛК переходит из режима **STOP** к режиму **RUN**. Состояния обмоток без удержания устанавливаются в ноль при отключении питания, или когда ПЛК переходит из режима **STOP** к режиму **RUN**.

Таблица 4-2. Типы обмоток

Тип обмотки	Отображение	Энергия в обмотке	Результат
Обычная	---( )---	ON	Устанавливает ссылку в ON.
		OFF	Устанавливает ссылку в OFF.
Инверсная	---(/)---	ON	Устанавливает ссылку в OFF.
		OFF	Устанавливает ссылку в ON.
С удержанием	---(M)---	ON	Устанавливает ссылку в ON с удержанием.
		OFF	Устанавливает ссылку в OFF с удержанием.
Инверсная с удержанием	---(/M)---	ON	Устанавливает ссылку в OFF с удержанием.
		OFF	Устанавливает ссылку в ON с удержанием.
По переднему фронту	—(↑)—	OFF→ ON	Если ссылка установлена в OFF, устанавливает ее в ON на один цикл.
По заднему фронту	—(↓)—	ON→OFF	Если ссылка установлена в OFF, устанавливает ее в ON на один цикл
SET	---(S)---	ON	Устанавливает ссылку в ON, пока ---(R)--- не установит ее в OFF.
		OFF	Не изменяет состояния ссылки.
RESET	---(R)---	ON	Устанавливает ссылку в OFF, пока ---(S)--- не установит ее в ON.
		OFF	Не изменяет состояния ссылки.
SET с удержанием	---(SM)---	ON	Устанавливает ссылку в ON с удержанием, пока ---(RM)--- не установит ее в OFF.
		OFF	Не изменяет состояния обмотки.
RESET с удержанием	---(RM)---	ON	Устанавливает ссылку в OFF с удержанием, пока ---(SM)--- не установит ее в ON
		OFF	Не изменяет состояния обмотки
Обмотка продолжения	-----<+>	ON	Устанавливает следующий контакт продолжения в ON.
		OFF	Устанавливает следующий контакт продолжения в OFF.

Нормально разомкнутый контакт —| |—

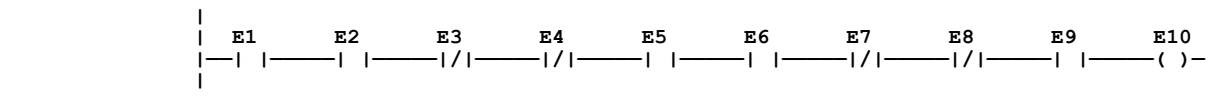
Нормально разомкнутый контакт действует как переключатель, который пропускает поток энергии, если связанная с ним ссылка установлена в состояние ON (1).

Нормально замкнутый контакт —|/|—

Нормально замкнутый контакт действует как переключатель, который пропускает поток энергии, если связанная с ним ссылка установлена в состояние OFF (0).

Пример

Следующий пример демонстрирует звено с десятью элементами, имеющими мнемонические имена с E1 по E10. Обмотка E10 установлена в ON, когда установлены в ON ссылки E1, E2, E5, E6 и E9, и установлены в OFF ссылки E3, E4, E7 и E8.

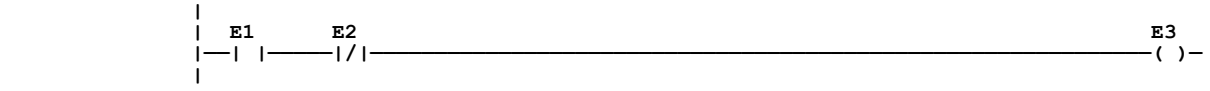


Обмотка —( )—

Обмотка устанавливает дискретную ссылку в ON, когда принимает поток энергии. Она не является удерживающей и поэтому не может использоваться со ссылками статуса системы (%SA, %SB, %SC, %G).

Пример

В следующем примере обмотка E3 установлена в ON, когда установлена в ON ссылка E1, и установлена в OFF ссылка E2.

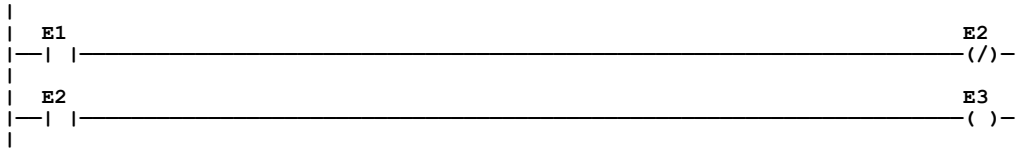


Инверсная обмотка —(/)—

Инверсная обмотка устанавливает дискретную ссылку в ON, когда НЕ принимает поток энергии. Она не является удерживающей и поэтому не может использоваться со ссылками статуса системы (%SA, %SB, %SC, %G).

Пример

В следующем примере обмотка E3 установлена в ON, когда установлена в OFF ссылка E1.



Обмотка с удержанием —(M)—

Как и обычная обмотка, обмотка с удержанием устанавливает дискретную ссылку в ON, когда она принимает поток энергии. Состояние обмотки с удержанием поддерживается при сбое в энергопитании. Следовательно, она не может использоваться со ссылками из заведомо не поддерживающейся памяти (%T).

Инверсная обмотка с удержанием —(/M)—

Инверсная обмотка с удержанием устанавливает дискретную ссылку в ON, когда она НЕ принимает поток энергии. Состояние обмотки с удержанием и отрицанием поддерживается при сбое в энергопитании. Следовательно, она не может использоваться со ссылками из заведомо не поддерживающейся памяти (%T).

Обмотка, срабатывающая по переднему фронту—(↑)—

Если ссылка, связанная с обмоткой, срабатывающей по переднему фронту, установлена в OFF, то, принимая поток энергии, обмотка устанавливает ее в ON до следующего исполнения, данной обмотки. (Если звено, содержащее обмотку, пропущено в последующих циклах, она остается в ON.) Эта обмотка может использоваться в качестве одновибратора.

Каждая ссылка в прикладной программе должна использоваться в качестве обмотки с переходом однократно, чтобы сохранить одноразовую сущность этой обмотки.

Обмотки с переходом могут использоваться либо со ссылками из сохраняемой памяти, либо со ссылками из не сохраняемой памяти (%Q, %M, %T, %G, %SA, %SB, %SC).

Обмотка, срабатывающая по заднему фронту —(↓)—

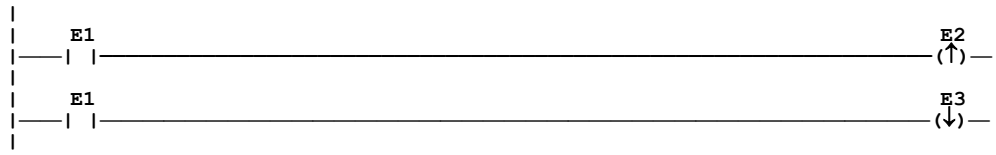
Если ссылка, связанная с этой обмоткой, установлена в OFF, то, когда обмотка прекращает принимать поток энергии, она устанавливает ссылку в ON, до тех пор, пока обмотка не будет исполнена следующий раз.

Каждая ссылка в прикладной программе должна использоваться в качестве обмотки с переходом однократно, чтобы сохранить одноразовую сущность этой обмотки.

Обмотки с переходом могут использоваться либо со ссылками из сохраняемой памяти, либо со ссылками из не сохраняемой памяти (%Q, %M, %T, %G, %SA, %SB, %SC).

Пример

В следующем примере, когда ссылка E1 переходит из OFF в ON, обмотки E2 и E3, принимают поток энергии, переключая E2 в ON на один цикл. Когда E1 переходит из ON в OFF, поток энергии отключается от E2 и E3 переключением обмотки E3 в ON на один цикл.



Обмотка SET —(S) —

Обмотки SET и RESET являются обмотками без удержания, которые можно использовать для удержания состояния ссылки (например E1) в ON или OFF. Когда обмотка SET принимает поток энергии, ее ссылка остается в состоянии ON (независимо от того, принимает ли сама обмотка поток энергии), пока не переустановится другой обмоткой.

Обмотка SET записывает неопределенный результат в бит перехода для данной ссылки. (Обратитесь за дополнительной информацией к разделу *Переходы и подстановки* в Главе 2.)

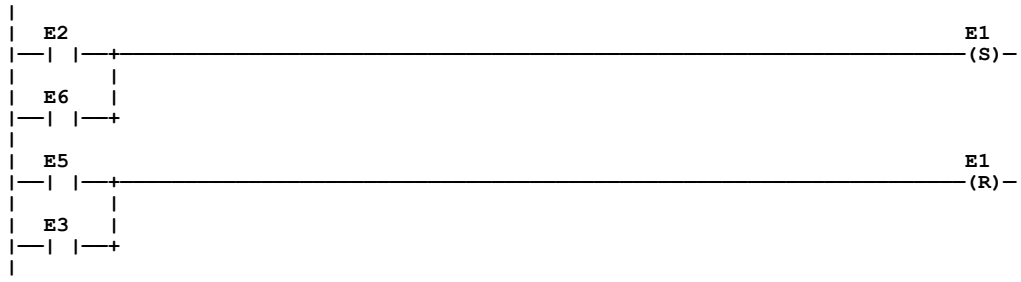
Обмотка RESET —(R)—

Обмотка RESET устанавливает дискретную ссылку в OFF, если принимает поток энергии. Ссылка остается в состоянии OFF, пока не переустановится другой обмоткой. Последняя разрешенная в паре SET или RESET обмотка имеет преимущество.

Обмотка RESET записывает неопределенный результат в бит перехода для данной ссылки. (Обратитесь за дополнительной информацией к разделу *Переходы и подстановки* в Главе 2.)

Пример

В следующем примере, обмотка, представленная E1, переходит в ON всякий раз, когда ссылки E2 или E6 находятся в состоянии ON. Обмотка, представленная E1, переходит в OFF всякий раз, когда ссылки E5 или E3 находятся в состоянии ON



Примечание

Когда уровень проверки использования обмоток соответствует SINGLE (Одиночный), вы можете использовать отдельную %M или %Q ссылку только с одной обмоткой, но вы можете использовать ее одновременно с одной обмоткой SET и одной обмоткой RESET. Когда уровень проверки обмотки соответствует WARN MULTIPLE (Множественный с предупреждением) или MULTIPLE (Множественный), то каждая ссылка может использоваться с несколькими обмотками, обмотками SET и обмотками RESET. При множественном использовании ссылка должна переключаться в ON либо обмоткой SET, либо обычной обмоткой, и должна переключаться в OFF либо обмоткой RESET, либо обычной обмоткой.

Обмотка SET с удержанием —(SM)—

Обмотки SET и RESET с удержанием аналогичны обмоткам SET и RESET, но удерживаются при сбое в энергопитании или когда ПЛК переходит из режима STOP в режим RUN. Обмотка SET с удержанием устанавливает дискретную ссылку в состояние ON, если обмотка принимает поток энергии. Ссылка остается ON, пока не переустановится обмоткой RESET с удержанием.

Обмотки SET с удержанием записывают неопределенный результат в бит перехода для данной ссылки. (Обратитесь за дополнительной информацией к разделу *Переходы и подстановки* в Главе 2.)

Обмотка RESET с удержанием —(RM)—

Эта обмотка устанавливает дискретную ссылку в состояние OFF, если обмотка принимает поток энергии. Ссылка остается OFF, пока не переустановится обмоткой SET с удержанием. Состояние обмотки удерживается при сбое в энергопитании или когда ПЛК переходит из режима STOP в режим RUN.

Обмотки RESET с удержанием записывают неопределенный результат в бит перехода для данной ссылки. (Обратитесь за дополнительной информацией к разделу *Переходы и подстановки* в Главе 2.)

Связи

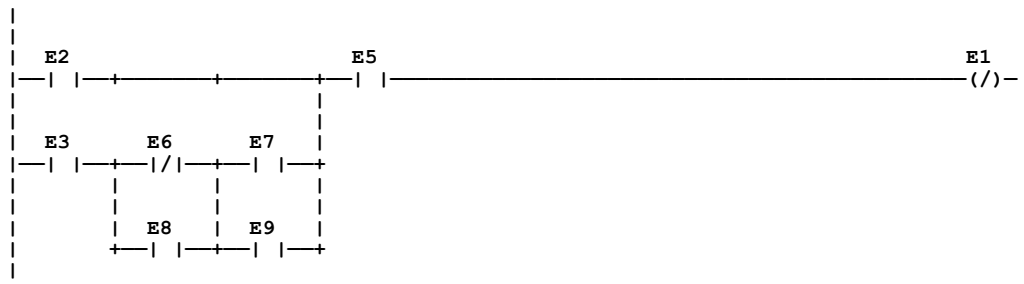
Горизонтальные и вертикальные связи используются как соединительные элементы строки многозвенной логики между функциями. Их назначением является осуществление логического потока (“энергии”) слева направо в логической последовательности.

Примечание

Вы не можете использовать горизонтальные связи для присоединения функции или обмотки к левой шине питания. Тем не менее, вы можете использовать системный бит ALW_ON (всегда On) в %S7 с нормально разомкнутым контактом, присоединенным к шине питания для вызова функции на каждом цикле.

Пример

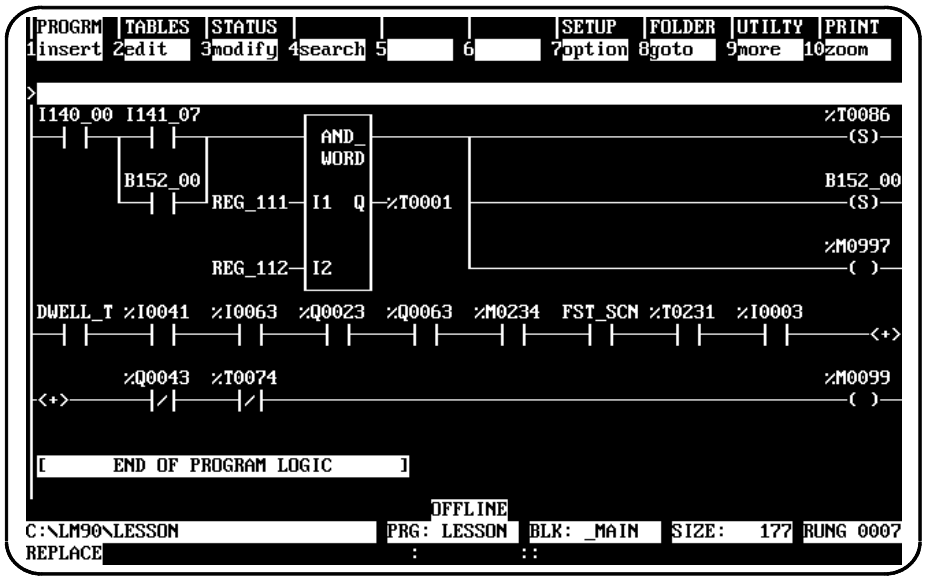
В следующем примере две горизонтальные связи соединяют контакты E2 и E5. Вертикальные связи использованы для соединения контактов E3, E6, E7, E8, и E9 с E2.



Обмотки продолжения (——<+>) и контакты продолжения (<+>——)

Обмотки продолжения (——<+>) и контакты продолжения (<+>——) используются для продолжения релейного логического звена сверх имеющегося лимита в 10 столбцов. Состояние последней исполненной обмотки продолжения является текущим состоянием, которое будет использовано при исполнении следующего контакта продолжения. Необходимо иметь обмотку продолжения, прежде чем логика исполнит контакт продолжения. Состояние контакта продолжения сбрасывается, когда ПЛК переходит из состояния STOP в состояние RUN, и поток прекращается, пока не будет установлена обмотка перехода после установки режима RUN.

На одном звене может быть только одна обмотка продолжения и контакт продолжения; контакт продолжения должен быть в столбце 1, обмотка продолжения должна быть в столбце 10. Ниже приведен пример обмотки продолжения и контакта продолжения.



Глава 5 Таймеры и счетчики

В настоящем разделе поясняется, как использовать таймеры задержки включения и таймеры-секундомеры, а также счетчики прямого и обратного счета. Данные, ассоциированные с этими функциями, удерживаются при включении/выключении питания.

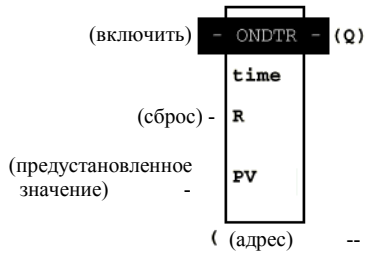
Обозначение	Функция	Страница
ONDTR	Таймер задержки включения с удержанием	5-3
TMR	Простой таймер задержки включения	5-6
OFDT	Таймер задержки выключения	5-9
UPCTR	Счетчик прямого счета	5-12
DNCTR	Счетчик обратного счета	5-14

Данные в памяти, необходимые таймерам и счетчикам

Каждый таймер или счетчик использует три слова (регистра) в памяти для запоминания следующей информации:

current value (CV)	word 1
preset value (PV)	word 2
control word	word 3

При вводе таймера или счетчика, вы должны ввести начальный адрес для этих трех слов (регистров) прямо под графическим представлением функции. Например:

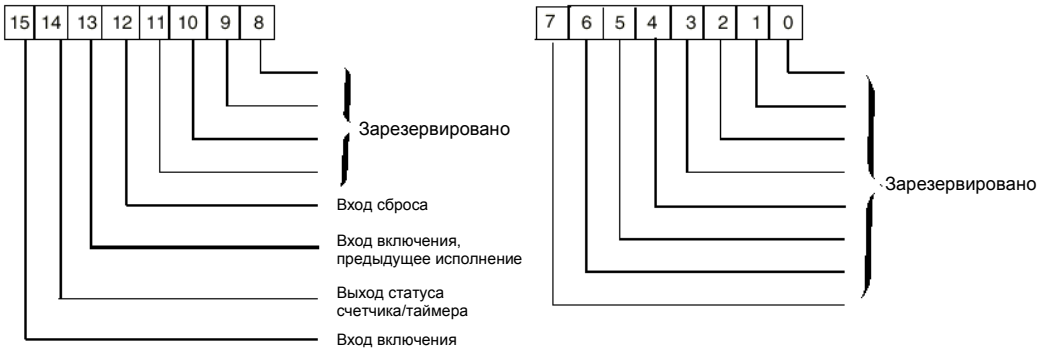


-- Введите здесь начальный адрес.

Примечание

Не используйте одни и те же регистры для различных блоков таймеров/счетчиков. Logicmaster **не проверяет** и не выдает предупреждения при наложении блоков регистров. Таймеры и счетчики не будут работать, если вы поместите текущее значение блока поверх предустановленного значения предыдущего блока.

Управляющее слово запоминает состояние булевых входов и выходов своего ассоциированного функционального блока, как видно из следующего формата:



Биты с 0 по 11 используются под точность таймеров и не используются в случае счетчиков.

Примечание

Будьте внимательны при использовании того же адреса PV в качестве второго слова в блоке из трех слов. Если PV не постоянно, оно обычно устанавливается не на место второго слова. Некоторые приложения выбирают для PV адрес второго слова, например, используя %R0102, когда нижний блок данных начинается в %R0101. Это позволяет приложению изменить PV при работе таймера или счетчика. Приложение может считать первое CV слово, или третье управляющее слово, но оно не сможет записать эти значения, или функция не будет работать.

Дополнительное примечание для операций с отдельными битами

При использовании функций Bit Test, Bit Set, Bit Clear или Bit Position, биты в слове нумеруются от 1 до 16, а ***НЕ*** от 0 до 15 как показано выше.

ONDTR - таймер задержки включения с удержанием

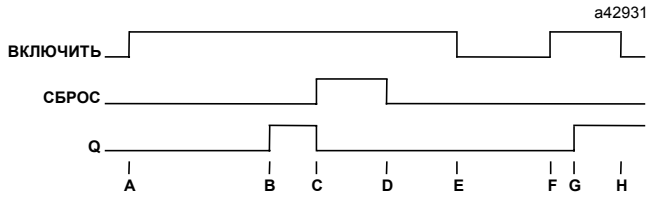
Таймер задержки включения с удержанием ONDTR дает приращение при приеме потока энергии и удерживает это значение при прекращении потока энергии. Время может быть подсчитано в десятых долях секунды (установлено по умолчанию), сотых долях или тысячных долях секунды. Диапазон составляет от 0 до ±32767 временных единиц. Состояние такого таймера сохраняется при пропадании энергии; при восстановлении энергии автоматической инициализации не происходит.

Когда ONDTR впервые принимает поток энергии, он начинает накапливать время (текущее значение). Когда такой таймер появляется в многозвенной логике, его текущее значение обновляется.

Примечание

Если многократное выполнение одного и того же таймера с одним и тем же ссылочным адресом разрешено в течение цикла ЦП, то текущие значения таких таймеров будут одинаковыми.

Когда текущее значение больше или равно предустановленному значению, выход Q возбуждается. В течении всего времени приема потока энергии таймер продолжает накопление, пока не будет достигнуто предельное значение. При достижении максимального значения оно удерживается, а выход Q остается возбужденным независимо от состояния разрешающего входа.



A = Уровень Enable (Включить) становится высоким, таймер начинает накопление

B = Значение CV достигает PV, Q становится высоким

C = Уровень Reset (Сброс) становится высоким, Q становится низким, накопленное время сбрасывается

D = Уровень Reset (Сброс) становится низким, таймер начинает новое накопление

E = Уровень Enable (Включить) становится низким, таймер останавливается. Накопленное время остается тем же самым.

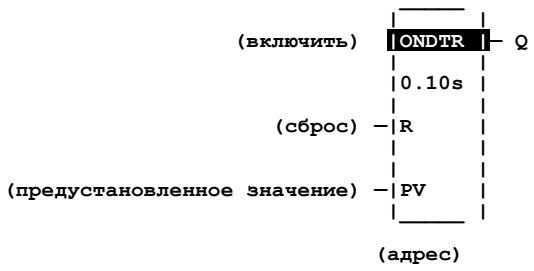
F = Уровень Enable (Включить) снова становится высоким. Таймер продолжает накопление.

G = CV становится равным PV; Q становится высоким. Таймер продолжает накапливать время пока уровень Enable (Включить) не станет низким, Reset (Сброс) станет высоким, или CV станет равным максимальному времени.

H = Уровень Enable (Включить) становится низким, таймер прекращает накапливать время.

Когда поток энергии к таймеру прекращается, текущее значение прекращает прирост и удерживается. Выход Q, если до этого был возбужден, остается в этом состоянии. Когда функция снова принимает поток энергии, текущее значение снова начинает прирастать, начиная с удержанного значения. Когда Reset принимает поток энергии, текущее значение

возвращается к нулю и снимается возбуждение выхода Q. В модулях ЦП 351 и 352, если уровень включения ONDTR-таймера PV=0 и Reset принимает поток энергии, то выход таймера устанавливается в OFF. Тем не менее, в модулях ЦП 311-341 при тех же условиях уровень выхода будет ON.



Параметры

Параметр	Описание
адрес	ONDTR использует три последовательных слова (регистра) %R памяти для хранения: Текущего значения CV = слово 1 Предустановленного значения PV = слово 2 Управляющего слова = слово 3 При вводе ONDTR вы должны ввести адрес для размещения этих трех последовательных слов (регистров) прямо под графическим представлением функции. **Замечание**. Не используйте этот адрес с другими командами. **Предупреждение**. Перекрытие ссылок приведет к ошибочной работе таймера.
включить	Когда принимает поток энергии, текущее значение таймера прирастает.
R	Когда R принимает поток энергии, сбрасывает текущее значение в 0.
PV	PV является значением для копирования в предустановленное значение таймера, когда он включен или сброшен.
Q	Выход Q обращается в 1, когда текущее значение больше или равно предустановленному значению.
время	Масштаб времени в десятых (0.1), сотых (0.01), и тысячных (0.001) долях секунды. Относится к значениям PV и CV.

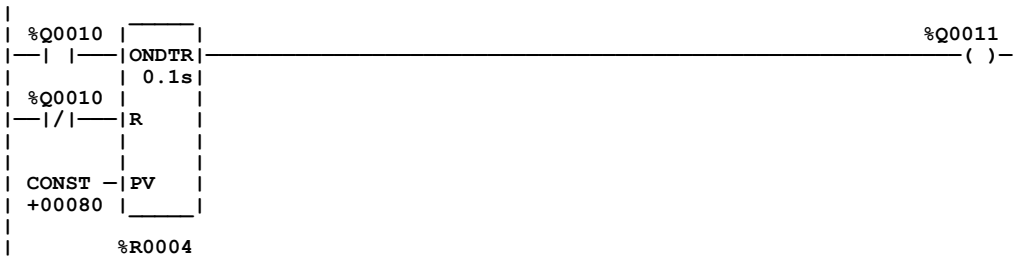
Допустимые типы памяти

Параметр	поток	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	const	нет
адрес								•				
включить	•											
R	•											
PV		•	•	•	•		•	•	•	•	•	•
Q	•											•

- Допустимая ссылка или место, где поток может проходить через данную функцию.

Пример

В следующем примере таймер задержки включения с удержанием используется для создания сигнала (%Q0011), который включается через 8 сек после включения %Q0010 и отключается при отключении %Q0010.



TMR – простой таймер задержки включения

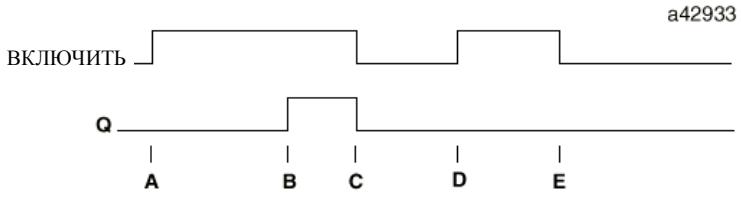
Простой таймер задержки включения TMR накапливает значение времени, когда принимает поток энергии и сбрасывается в нуль, когда поток прекращается. Время может быть подсчитано в десятых долях секунды (установлено по умолчанию), сотых долях или тысячных долях секунды. Диапазон составляет от 0 до ±32767 единиц, поэтому диапазон времени составляет от 0.001 до 3276.7 секунд. Состояние таймера сохраняется при пропадании энергии; при восстановлении энергии автоматической инициализации не происходит.

Когда TMR принимает поток энергии, он начинает накапливать время (текущее значение). Когда такой таймер появляется в многозвенной логике, его текущее значение обновляется, отображая общее прошедшее время, в течение которого таймер был включен со времени последнего сброса.

Примечание

Если многократное выполнение одного и того же таймера с одним и тем же ссылочным адресом разрешено в течение цикла ЦП, то текущие значения таких таймеров будут одинаковыми.

Обновление текущего значения производится пока логический уровень включения остается в состоянии ON. Когда текущее значение больше или равно предустановленному значению, функция начинает пропускать направо поток энергии. Таймер продолжает накопление времени, пока не будет достигнуто максимально возможное значение. При переходе параметра включения из состояния ON в состояние OFF, таймер прекращает накопление, и текущее значение сбрасывается к нулю.



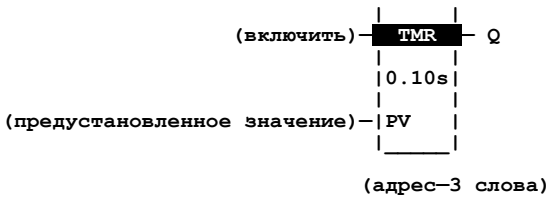
A = Уровень Enable (Включить) становится высоким, таймер начинает накопление.

B = Текущее значение достигает величины PV, Q становится высоким, таймер продолжает накапливать время.

C = Уровень Enable (Включить) становится низким, уровень Q становится низким, таймер прекращает накапливать время и текущее значение сбрасывается.

D = Уровень Enable (Включить) снова становится высоким. Таймер начинает накопление времени.

E = Уровень Enable (Включить) становится низким до достижения текущим значением предустановленного, уровень Q остается низким, таймер прекращает накапливать время и сбрасывается к нулю.



Параметры

Параметр	Описание
адрес	TMR использует три последовательных слова (регистра) %R памяти для запоминания: Текущего значения CV = слово 1 Предустановленного значения PV = слово 2 Управляющего слова = слово 3 При вводе TMR вы должны ввести адрес для размещения этих трех последовательных слов (регистров) прямо под графическим представлением функции. **Замечание**. Не используйте этот адрес с другими командами. **Предупреждение**. Перекрытие ссылок приведет к ошибочной работе таймера.
включить	Когда принимает поток энергии, текущее значение таймера прирастает. Когда TMR не включен, текущее значение сбрасывается в нуль и выход Q отключается.
PV	PV является значением для копирования в предустановленное значение таймера, когда он включен или сброшен.
Q	Выход Q обращается в 1, когда TMR включен и текущее значение больше или равно предустановленному значению.

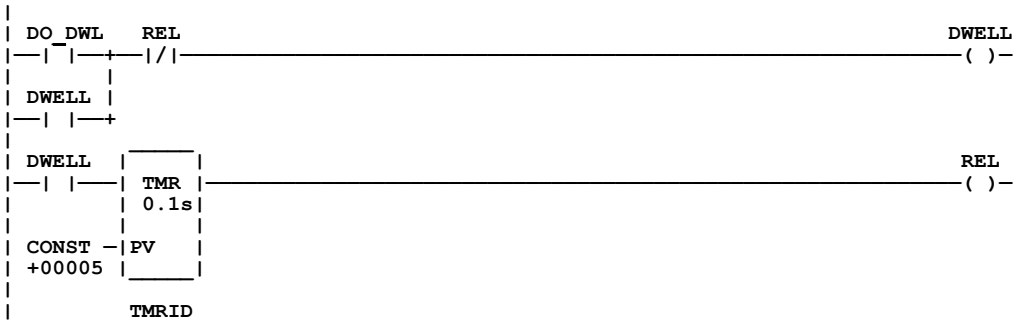
Допустимые типы памяти

Параметр	поток	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	пост.	нет
адрес								•				
включить	•											
PV		•	•	•	•		•	•	•	•	•	•
Q	•											•

- Допустимая ссылка или место, где поток может проходить через данную функцию.

Пример

В следующем примере таймер задержки включения TMRID используется для управления временем включения обмотки DWELL. Когда нормально разомкнутый контакт DO_DWL включается, обмотка DWELL запитывается. Контакт обмотки DWELL поддерживает питание DWELL (при отпускании контакта DO_DWL) и запускает таймер TMRID. Когда TMRID достигает предустановленного значения, равного половине секунды, обмотка REL запитывается, прекращая защелкнутое состояние обмотки DWELL. Контакт DWELL прерывает поток энергии к TMRID, сбрасывая его текущее значение и снимая питание с обмотки REL. Цепь готова к следующей временной активации контакта DO_DWL.



OFDT – таймер задержки выключения

Таймер задержки выключения OFDT накапливает время, когда прекращается поток энергии, и сбрасывается к нулю, когда поток возобновляется. Время может быть подсчитано в десятых долях секунды (установлено по умолчанию), сотых или тысячных долях секунды. Диапазон составляет от 0 до +32767 единиц времени. Состояние таймера сохраняется при пропадании энергии; при восстановлении энергии автоматической инициализации не происходит.

Когда OFDT впервые принимает поток энергии, он пропускает его направо, и текущее значение (current value – CV) устанавливается в нуль. (OFDT использует первое слово [регистр] в качестве места хранения CV). Выход остается активным до тех пор, пока функция принимает поток энергии. Когда функция прекращает прием потока энергии слева, она продолжает пропускать его направо, а таймер начинает накапливать время в текущем значении.

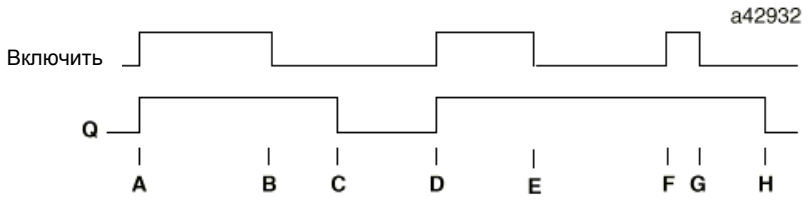
Примечание

Если многократное выполнение одного и того же таймера с одним и тем же ссылочным адресом разрешено в течение цикла ЦП, то текущие значения таких таймеров будут одинаковыми.

OFDT не пропускает поток энергии, если предустановленное значение равно нулю или отрицательное.

Каждый раз, когда функция инициализируется установкой логического уровня разрешения в состоянии OFF, текущее значение обновляется для отображения времени, прошедшего с момента отключения таймера. Когда текущее значение больше или равно предустановленному значению PV, функция прекращает пропускать направо поток энергии. Когда это происходит, таймер прекращает накопление времени (см. ниже пункт C).

Когда функция снова принимает поток энергии, текущее значение сбрасывается в нуль.



A = Уровни Enable (Включить) и Q становятся высокими, таймер сбрасывается в нуль (CV=0).

B = Уровень Enable (Включить) становится низким, таймер начинает накапливать время.

C = Текущее значение CV достигает предустановленного PV, уровень Q становится низким, таймер прекращает накопление времени..

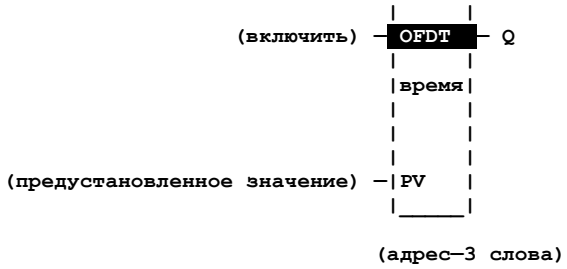
D = Уровень Enable (Включить) становится высоким, таймер сбрасывается.

E = Уровень Enable (Включить) становится низким, таймер начинает накопление времени.

F = Уровень Enable (Включить) снова становится высоким. Таймер сбрасывается.

G = Уровень Enable (Включить) становится низким, таймер начинает накапливать время.

H = CV достигает значения PV; уровень Q становится низким, таймер прекращает накапливать время.



Если OFDT используется в программном блоке, который не вызывается при каждом цикле, то таймер накапливает время между вызовами к этому программному блоку до тех пор, пока он не будет сброшен. Это означает, что он функционирует как таймер, работающий в программе со значительно меньшей скоростью цикла, чем таймер в главном программном блоке. Для программных блоков, не активируемых в течение длительного времени, этот таймер следует программировать так, чтобы сделать возможным такое «подхватывание». Например, если таймер в каком-нибудь программном блоке сбрасывается, и этот программный блок не вызывается (остается не активным) в течение четырех минут, то когда этот программный блок будет вызван, уже будут накоплены четыре минуты времени. Это время будет придано данному таймеру при его включении, если только он не будет перед этим сброшен.

Параметры

Параметр	Описание
адрес	OFDT использует три последовательных слова (регистра) памяти %R для удержания: Текущего значения (CV) = слово 1 Предустановленного значения (PV) = слово 2 Управляющего слова = слово 3 При вводе OFDT вы должны ввести адрес для размещения этих трех последовательных слов (регистров) прямо под графическим представлением функции. **Замечание:** Не используйте этот адрес с другими командами. **Предупреждение:** Перекрытие ссылок приведет к ошибочной работе таймера.
включить	Когда принимает поток энергии, текущее значение таймера прирастает.
время	Время P1 задает тип единицы времени (миллисекунда и т.п.), используемый регистрами.
PV	PV является значением для копирования в предустановленное значение таймера, когда он включен или сброшен. Для регистровой (%R) ссылки PV, параметр PV задается как второе слово адресного параметра. Например, параметр адреса %R0001 будет использовать %R0002 в качестве параметра PV.
Q	Выход Q обращается в 1, когда текущее значение меньше предустановленного значения. Значение Q запоминается при отключении энергии; при включении энергии автоматическая инициализация не выполняется.

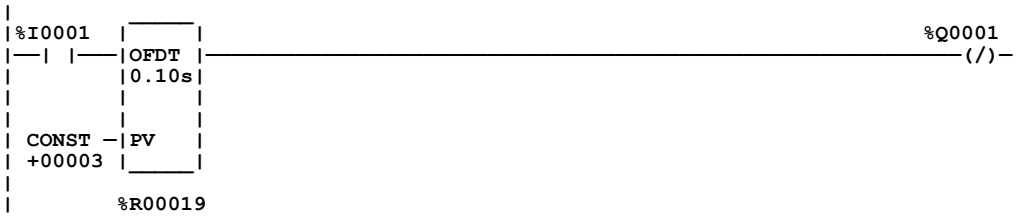
Допустимые типы памяти

Параметр	поток	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	пост.	нет
адрес								•				
включить	•											
PV	•	•	•	•	•		•	•	•	•	•	•
Q	•											•

• Допустимая ссылка или место, где поток может проходить через данную функцию.

Пример

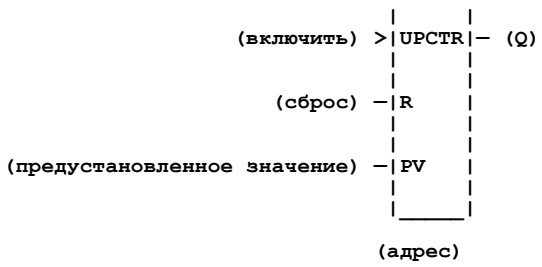
В следующем примере таймер OFDT используется для отключения выхода (%Q0001) каждый раз, когда вход (%I0001) включается. Выход повторно включается через 0.3 сек после отключения входа.



UPCTR – счетчик прямого счета

Функция счетчика прямого счета UPCTR используется для прямого счета до назначенного значения. Диапазон счета составляет от 0 до 32767 единиц. Когда счетчик устанавливается в состояние ON (Включено), текущее значение счетчика устанавливается в нуль. Каждый раз, когда включающий вход переходит из состояния OFF к состоянию ON, текущее значение прирастает на единицу. Текущее значение может прирастать после достижения предустановленного значения. Выход устанавливается в состояние ON, когда текущее значение больше или равно предустановленному.

Состояние такого счетчика сохраняется при пропадании энергии; при восстановлении энергии автоматической инициализации не происходит.



Параметры

Параметр	Описание
адрес	UPCTR использует три последовательных слова (регистра) памяти %R для запоминания: ▪ Текущего значения (CV) = слово 1 ▪ Предустановленного значения (PV) = слово 2 ▪ Управляющего слова = слово 3 При вводе UPCTR вы должны ввести адрес для размещения этих трех последовательных слов (регистров) прямо под графическим представлением функции. **Замечание:** Не используйте этот адрес с другими счетчиками или командами во избежание сбоев в работе. **Предупреждение:** Перекрытие ссылок приведет к ошибочной работе таймера.
включить	При положительном переходе текущее значение счетчика увеличивается на 1.
R	Когда R принимает поток энергии, оно сбрасывает текущее значение на нуль.
PV	PV является значением для копирования в предустановленное значение счетчика, когда он включен или сброшен.
Q	Выход Q возбуждается, когда текущее значение больше или равно предустановленному значению PV.

Допустимые типы памяти

Параметр	поток	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	пост.	нет
адрес								•				
включить	•											
R	•											
PV		•	•	•	•		•	•	•	•	•	•
Q	•											•

• Допустимая ссылка или место, где поток может проходить через данную функцию.

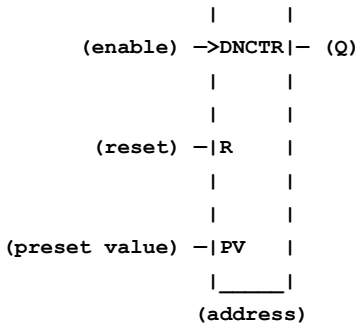
Пример

В следующем примере при каждом переходе входа %I0012 из состояния OFF в состояние ON счетчик прямого счета PRT_CNT увеличивается на 1; внутренняя обмотка %M0001 запитывается когда будет накоплено 100 единиц. Когда %M0001 в состоянии ON, накопленное значение сбрасывается в нуль.

DNCTR – счетчик обратного счета

Функция счетчика обратного счета DNCTR используется для обратного счета от назначенного значения. Минимальное предустановленное значение равно 0, максимальное составляет 32767 единиц. Минимальное текущее значение равно -32767. При сбросе, текущее значение счетчика устанавливается равным предустановленному значению. Каждый раз, когда включающий вход переходит из состояния OFF к состоянию ON, текущее значение уменьшается на единицу. Текущее значение может прирастать после достижения предустановленного значения. Выход устанавливается в состояние ON, когда текущее значение меньше или равно нулю.

Текущее значение такого счетчика сохраняется при пропадании энергии; при восстановлении энергии автоматической инициализации не происходит.



Параметры

Параметр	Описание
адрес	DNCTR использует три последовательных слова (регистра) памяти %R для запоминания: ▪ Текущего значения (CV) = слово 1 ▪ Предустановленного значения (PV) = слово 2 ▪ Управляющего слова = слово 3 При вводе DNCTR вы должны ввести адрес для размещения этих трех последовательных слов (регистров) прямо под графическим представлением функции. **Замечание:** Не используйте этот адрес с другими счетчиками или командами во избежание сбоев в работе. **Предупреждение:** Перекрытие ссылок приведет к ошибочной работе таймера.
включить	При положительном переходе текущее значение счетчика уменьшается на 1.
R	Когда R принимает поток энергии, оно сбрасывает текущее значение к предустановленному.
PV	PV является значением для копирования в предустановленное значение счетчика, когда он включен или сброшен.
Q	Выход Q возбуждается, когда текущее значение меньше или равно нулю.

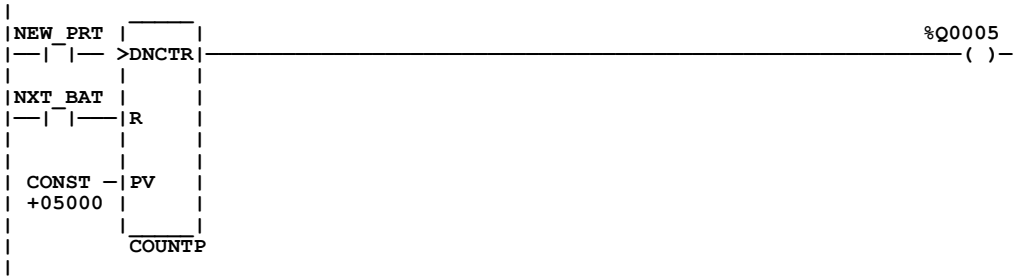
Допустимые типы памяти

Параметр	поток	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	пост.	нет
адрес								•				
включить	•											
R	•											
PV		•	•	•	•		•	•	•	•	•	•
Q	•											•

• Допустимая ссылка или место, где поток может проходить через данную функцию.

Пример

В следующем примере счетчик обратного счета, обозначенный как COUNTR, отсчитывает 500 новых деталей, после чего формирует выход %Q0005.

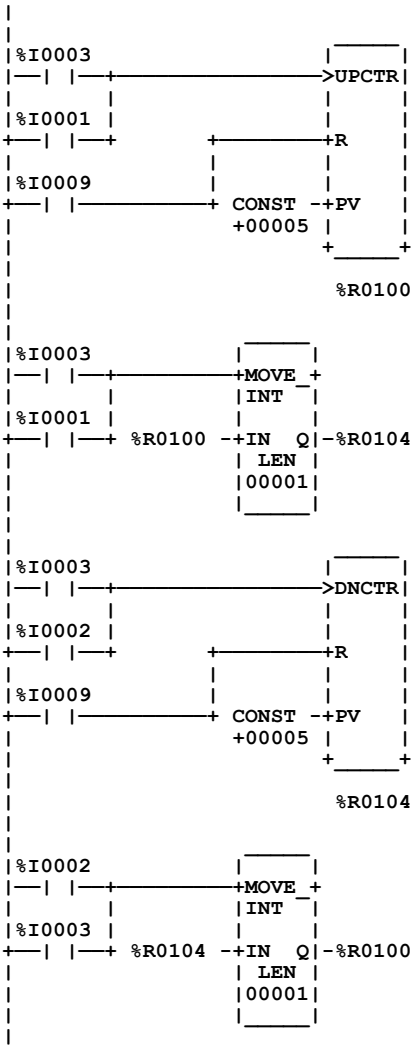


Пример

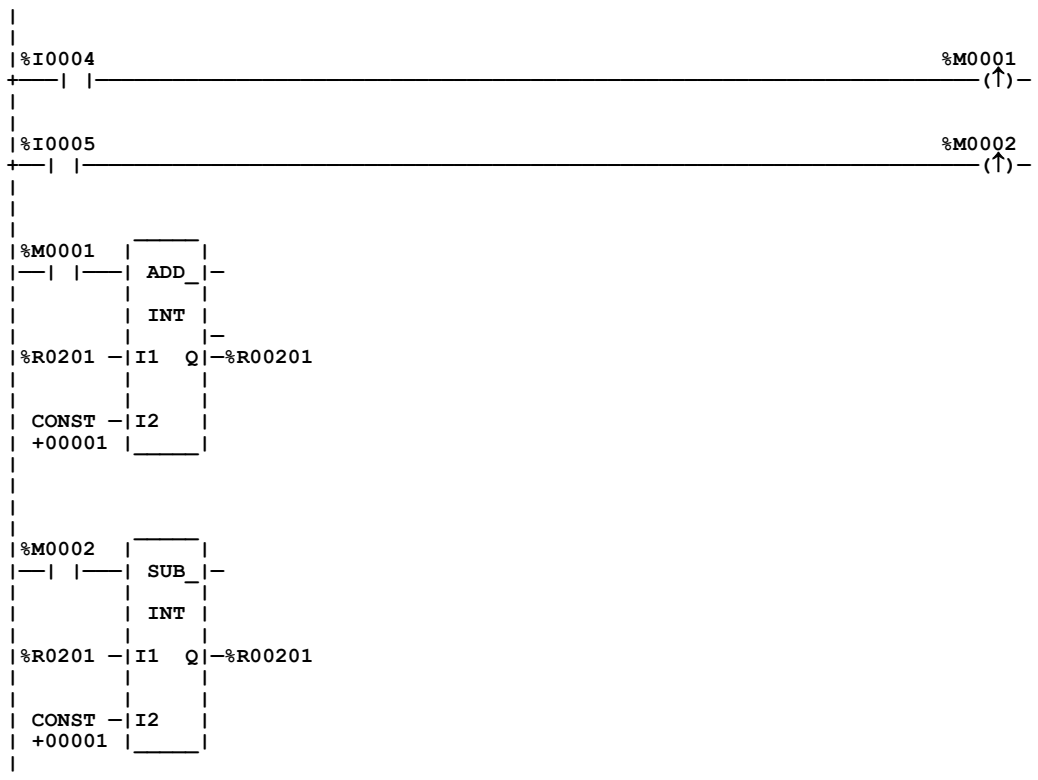
В следующем примере программируемый контроллер используется для отслеживания числа деталей, содержащихся в области временного хранения. Имеется два способа исполнения этой функции с использованием системы команд Series 90-30/20/Micro.

Первый способ заключается в использовании пары счетчиков прямого и обратного счета с совместно используемым регистром для накопления текущего значения.

Когда изделие поступает в область хранения, счетчик прямого счета прирастает на 1, увеличивая текущее значение числа изделий на складе на 1. Когда изделие покидает склад, счетчик обратного счета уменьшается на 1, уменьшая в свою очередь число хранимых изделий на 1. Для избежания конфликтов при совместном использовании регистра, оба счетчика используют разные адреса регистра. Когда в регистре изменяется значение, оно должно быть передано в регистр текущего значения другого счетчика.



Второй способ, показанный на следующем рисунке, использует функции ADD и SUB для наблюдения за складом.



Глава
6

Математические функции

В настоящем разделе рассмотрены математические функции системы команд контроллеров Series 90-30/20/Micro:

Сокращение	Функция	Описание	Стр.
ADD	Сложение	Суммирует два числа	6-2
SUB	Вычитание	Вычитает одно число из другого	6-2
MUL	Умножение	Перемножает два числа	6-2
DIV	Деление	Делит одно число на другое, получая частное	6-2
MOD	Деление по модулю	Делит одно число на другое, получая остаток	6-7
SQRT	Извлечение корня	Определяет квадратный корень целой или действительной величины	6-10
SIN, COS, TAN, ASIN, ACOS, ATAN	Тригонометрические функции†	Определяет соответствующую функцию от действительного значения на входе IN	6-12
LOG, LN, EXP, EXPT	Логарифмические/ Экспоненциальные функции †	Определяет соответствующую функцию от действительного значения на входе IN	6-14
RAD, DEG	Преобразование радиан †	Определяет соответствующую функцию от действительного значения на входе IN	6-16

† Тригонометрические функции, логарифмические/экспоненциальные функции, и функции преобразования радиан доступны только для ЦП модели 352.

Примечание

Функции деление и деление по модулю являются аналогичными друг другу, отличаясь тем, что первая определяет частное, а вторая - остаток.

Стандартные математические функции (ADD, SUB, MUL, DIV)

К этим математическим функциям относятся сложение, вычитание, умножение и деление. Когда функция принимает поток энергии, соответствующие действия выполняются над входными параметрами I1 и I2. Параметры должны относиться к данным одного типа. Выход Q также относится к данным того же типа.

Примечание

Функция DIV округляет к меньшему значению; она не округляет до ближайшего целого значения. Например, 24 DIV 4 = 4.

Математические функции оперируют следующими типами данных:

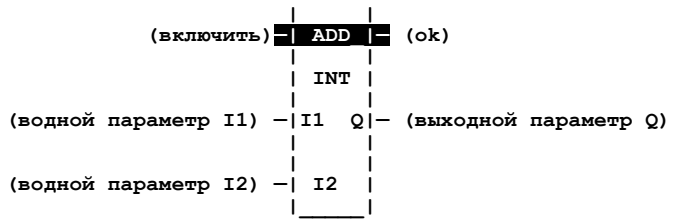
Тип данных	Описание
INT	Целое число со знаком
DINT	Целое число со знаком двойной разрядности
REAL	С плавающей запятой

Примечание

Тип REAL по умолчанию используется только в центральных процессорах типа 352.

По умолчанию данные относятся к целочисленному знаковому типу; но тип может быть изменен после выбора функции. Для получения более подробных сведений о типах данных обратитесь к главе 2, разделу 2 “ Организация программы пользователя и данных ”.

Если операции над данными типа INT и DINT приводят к переполнению, то выходная ссылка устанавливается к наибольшему допустимому значению для данного типа данных. Для чисел со знаками, знак будет указывать направление переполнения. Если операция не привела к переполнению (и на входе имеются данные допустимого формата), выход подтверждения OK устанавливается в состояние ON, в противном случае в состояние OFF. Если используются знаковые целые или целые двойной разрядности, то знак результата для функций DIV и MUL зависит от знаков I1 и I2.



Параметры

Параметр	Описание
включить	Когда функция включена, соответствующая ей операция исполняется.
I1	I1 содержит константу или ссылку для первой величины, используемой в операции. (I1 помещается слева в математическом выражении, например I1-I2.)
I2	I2 содержит константу или ссылку для второй величины, используемой в операции. (I2 помещается справа в математическом выражении, например I1-I2.)
OK	Выход OK запитывается когда операция выполнена без переполнения, если только сама эта операция не окажется недопустимой.
Q	Выход Q содержит результат операции.

Допустимые типы памяти

Параметр	поток	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	пост.	нет
включить	•											
I1		o	o	o	o		o	•	•	•	•†	
I2		o	o	o	o		o	•	•	•	•†	
OK	•											•
Q		o	o	o	o		o	•	•	•		

• Допустимая ссылка или место, где поток проходит через данную функцию.

o Допустимая ссылка только для данных типа INT; не допустима для DINT и REAL.

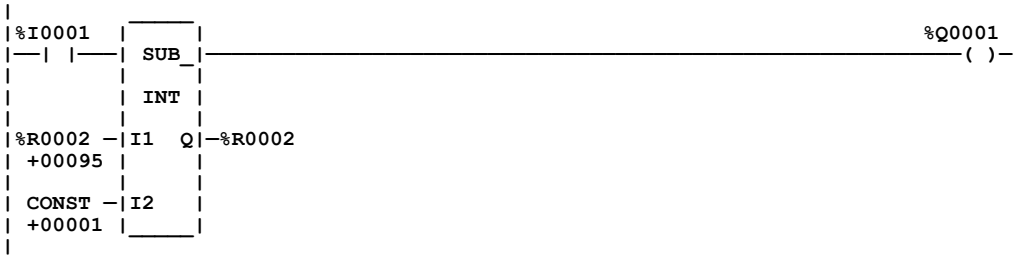
↑ Константы ограничены значениями в диапазоне от -32768 до 32767 для знаковых целочисленных операций двойной разрядности.

Примечание

Тип данных по умолчанию относится к типу INT для 16-битовых или однорегистровых операндов. Нажмите клавишу F10 для изменения выбранного типа к типу DINT, 32-разрядному двойному слову, или REAL (только для ЦП 352). Значения PLC типа INT занимают одиночный 16-битовый регистр %R, %AI% или AQ. Значения типа DINT требуют двух последовательных регистров с младшими 16 битами в первом слове и старшими 16 битами со знаком во втором слове. Значения типа REAL (только для ЦП 352) также занимают 32-битный двойной регистр со знаком в старшем бите, за которым следуют показатель и мантисса.

Пример

В следующем примере, когда вход %I0001 устанавливается, целочисленное содержание (%R0002) уменьшается на 1 и обмотка %Q0001 включается, при условии отсутствия переполнения при вычитании.



Математические функции и типы данных

Функция	Операция	Отображается как
ADD INT	Q(16 бит)=I1(16 бит)+I2(16 бит)	5 цифр десятичного числа со знаком
ADD DINT	Q(32 бит)=I1(32 бит)+I2(32 бит)	8 цифр десятичного числа со знаком
ADD REAL*	Q(32 бит)=I1(32 бит)+I2(32 бит)	7 цифр десятичного числа со знаком и точкой
SUB INT	Q(16 бит)=I1(16 бит)-I2(16 бит)	5 цифр десятичного числа со знаком
SUB DINT	Q(32 бит)=I1(32 бит)-I2(32 бит)	8 цифр десятичного числа со знаком
SUB REAL*	Q(32 бит)=I1(32 бит)-I2(32 бит)	7 цифр десятичного числа со знаком и точкой
MUL INT	Q(16 бит)=I1(16 бит)*I2(16 бит)	5 цифр десятичного числа со знаком
MUL DINT	Q(32 бит)=I1(32 бит)*I2(32 бит)	8 цифр десятичного числа со знаком
MUL REAL*	Q(32 бит)=I1(32 бит)*I2(32 бит)	7 цифр десятичного числа со знаком и точкой
DIV INT	Q(16 бит)=I1(16 бит)/I2(16 бит)	5 цифр десятичного числа со знаком
DIV DINT	Q(32 бит)=I1(32 бит)/I2(32 бит)	8 цифр десятичного числа со знаком
DIV REAL*	Q(32 бит)=I1(32 бит)/I2(32 бит)	7 цифр десятичного числа со знаком и точкой

* - только для ЦП 35x и 36x версии 9 и более поздних, или для ЦП 352 всех версий

Примечание

Тип входных и выходных данных должен быть одинаков. Функции MUL и DIV не поддерживают смешанный режим, имеющий место в контроллерах 90-70. Например, функция MUL INT двух 16-разрядных входов дает 16-разрядное произведение, а не 32-разрядное. Использование функции MUL DINT для 32-разрядного произведения требует, чтобы оба входа были 32-разрядными. Функция DIV INT делит 16-разрядный вход I1 на 16-разрядный вход I2 и производит 16-разрядный результат, тогда как функция

DIV DINT делит 32-разрядный вход I1 на 32-разрядный вход I2 и производит 32-разрядный результат.

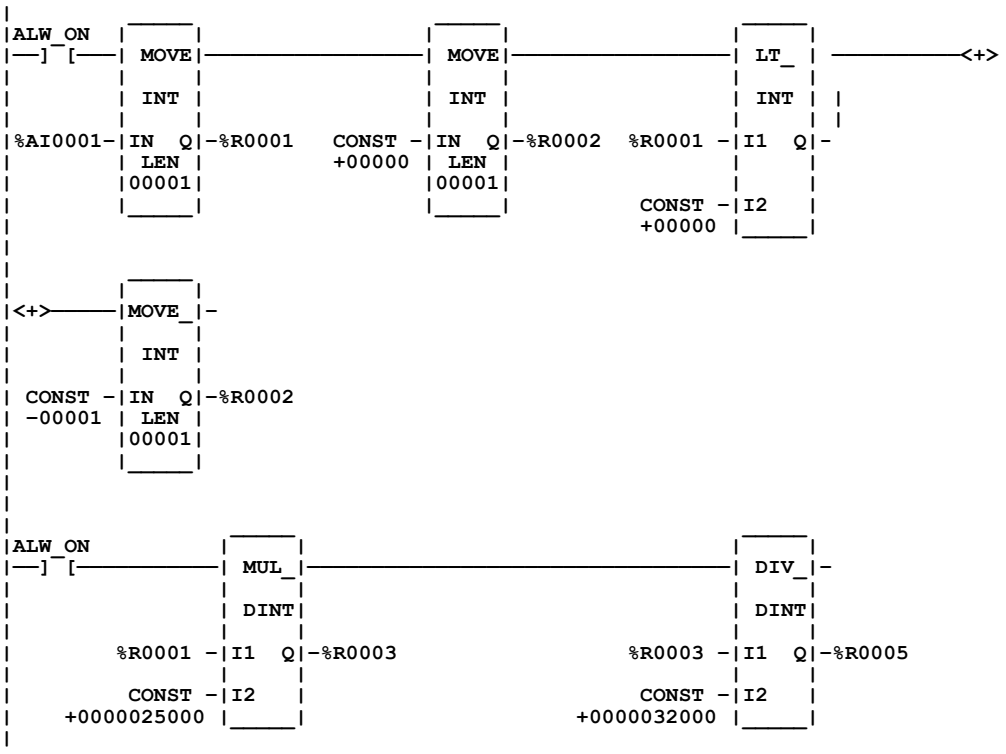
Эти функции пропускают энергетический поток, если не наступает переполнения. При переполнении результатом является наибольшее возможное значение с соответствующим знаком, при этом поток не пропускается.

Будьте внимательны и не допускайте переполнения при использовании функций MUL и DIV. Если вы хотите преобразовать величину типа INT в величину типа DINT, помните, что ЦП использует стандартный двоичный дополнительный код со знаком, присоединенным к старшему биту второго слова. Вы должны проверить знак младшего 16 разрядного слова и распространить его на второе 16 битное слово. Если старший значащий бит в 16 разрядном INT слове равен -1 (отрицательный), переместите -1, или в шестнадцатеричном представлении 0FFFFh, во второе слово. Преобразование от типа DINT к типу INT проще, так как младшее 16-битовое слово (первый регистр) является INT частью 32-разрядного слова DINT. Старшие 16 бит, или второе слово, должны быть либо 0 (положительными), или -1 (отрицательными) значениями, либо DINT число слишком велико, чтобы быть преобразованным к 16-разрядному представлению.

Пример

Обычно приложения масштабируют значения аналоговых входов с MUL операцией, после которой следует операция DIV или, возможно, ADD операция. В диапазоне до 32000, использование MUL INT ведет к переполнению. Использование %AI величины для MUL DINT также не будет работать, так как 32 битное значение I1 объединит 2 аналоговых входа одновременно. Вы должны переместить аналоговый вход в младшее слово двойного регистра, затем проверить знак и установить второй регистр в 0, если он положительный, и в -1, если он отрицательный. Используйте двойной регистр с функцией MUL DINT для 32 битного произведения и последующей функцией DIV.

Например, для масштабирования входа %AI1 величиной ±10 В к инженерным единицам в %R5 должна быть использована следующая логика.



Функция MOD (INT, DINT)

Функция деления по модулю используется для получения остатка при делении одной величины на другую для данных одного типа. Знак результата всегда совпадает со знаком входного параметра I1.

Функция MOD оперирует со следующими типами данных:

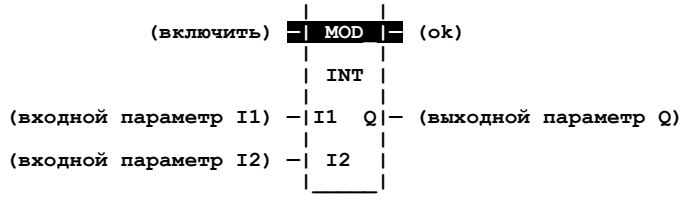
Тип данных	Описание
INT	Целое число со знаком
DINT	Целое число со знаком двойной разрядности

По умолчанию тип данных относится к целочисленному знаковому, но может быть изменен после выбора функции. Для получения более подробных сведений о типах данных обратитесь к главе 2, разделу 2 «Организация программы пользователя и данных».

Когда функция принимает поток энергии, она делит входной параметр I1 на входной параметр I2. Эти параметры должны относиться к данным одного типа. Выход Q вычисляется по формуле:

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

где функция DIV производит целочисленное значение. Выход Q относится к данным того же типа, что и входные параметры. Параметр OK всегда находится в состоянии ON, если функция принимает поток энергии, до тех пор, пока не будет предпринята попытка деления на 0. В этом случае параметр установится к значению OFF.



Параметры

Параметр	Описание
включить	Когда функция включена, соответствующая ей операция исполняется.
I1	I1 содержит константу или ссылку величины, которая делится на I2.
I2	I2 содержит константу или ссылку величины, на которую делится I1.
OK	Выход OK запитывается когда операция выполнена без переполнения.
Q	Выход Q содержит остаток от деления I1 на I2.

Допустимые типы памяти

Параметр	поток	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	пост.	нет
включить	•											
I1		o	o	o	o		o	•	•	•	•↑	
I2		o	o	o	o		o	•	•	•	•↑	
OK	•											•
Q		o	o	o	o		o	•	•	•		

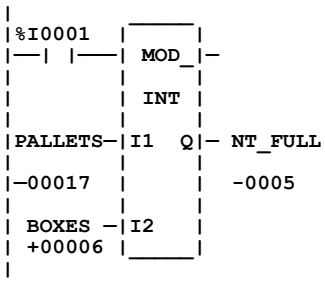
• Допустимая ссылка или размещение, когда поток может проходить функцию.

o Допустимая ссылка только для данных типа INT; не допустима для DINT и REAL.

↑ Константы ограничены значениями в диапазоне от -32768 до +32767 для знаковых целочисленных операций двойной разрядности.

Пример

В следующем примере, остаток целочисленного деления величин PALLETS и BOXES помещается в NT_FULL каждый раз, когда %I0001 находится в состоянии ON.



Функция SQRT (INT, DINT, REAL)

Функция извлечения корня используется для получения квадратного корня данной величины. Когда функция принимает поток энергии, значение выхода Q устанавливается равным целой части корня квадратного входной величины IN. Выход Q должен относиться к данным того же типа, что и IN.

Функция SQRT оперирует со следующими типами данных:

Тип данных	Описание
INT	Целое число со знаком
DINT	Целое число со знаком двойной разрядности
REAL	С плавающей запятой

Примечание

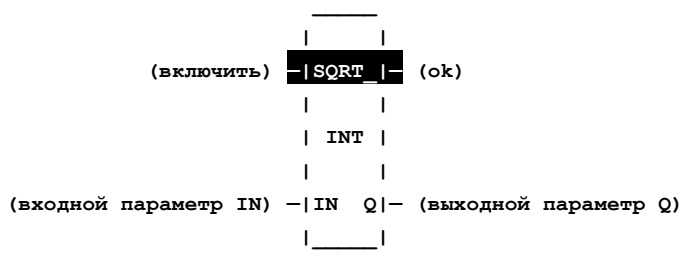
Данные типа REAL доступны только для процессора типа 352, а также 35x и 36x версии 9 и выше.

По умолчанию тип данных относится к целочисленному знаковому; но может быть изменен после выбора функции. Для получения более подробных сведений о типах данных обратитесь к главе 2, разделу 2 “Организация программы пользователя и данных.”

Параметр OK устанавливается в состояние ON, если операция выполнена без переполнения, если не возникает одна из следующих недопустимых REAL операций:

- IN<0
- IN в состоянии NaN (не число)

Иначе, параметр OK устанавливается в состояние OFF.



Параметры

Параметр	Описание
включить	Когда функция включена, соответствующая ей операция исполняется.
IN	IN содержит константу или ссылку величины, квадратный корень которой определяется. Если IN меньше нуля, функция не пропустит поток энергии.
OK	Выход OK запитывается когда операция выполнена без переполнения, если не было недопустимых операций.
Q	Выход Q содержит результат извлечения корня из IN.

Допустимые типы памяти

Параметр	поток	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	пост.	нет
включить	•											
IN		o	o	o	o		o	•	•	•	•↑	
OK	•											•
Q		o	o	o	o		o	•	•	•		

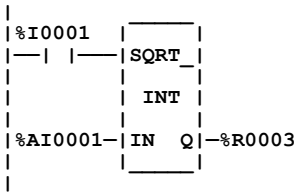
• Допустимая ссылка или размещение, когда поток может проходить функцию.

o Допустимая ссылка только для данных типа INT; не допустима для DINT и REAL.

↑ Константы ограничены значениями в диапазоне от -32768 до +32767 для знаковых целочисленных операций двойной разрядности.

Пример

В следующем примере, корень квадратный целого числа, расположенного в %AI001, помещается в результат, расположенный в %R0003, когда %I0001 находится в состоянии ON.



Тригонометрические функции SIN, COS, TAN, ASIN, ACOS, ATAN

Функции SIN, COS и TAN используются для вычисления значений соответствующих тригонометрических функций от входной величины. Когда функция принимает поток энергии, она вычисляет соответствующее тригонометрическое значение входа IN, заданного в радианах, и запоминает результат в выходе Q. Вход IN и выход Q являются числами с плавающей запятой.

Функции ASIN, ACOS и ATAN используются для вычисления соответствующих значений обратных тригонометрических функций данной входной величины. Когда функция принимает поток энергии, она вычисляет соответствующее значение обратной тригонометрической функции входа IN, и запоминает результат в выходе Q, заданном в радианах. Вход IN и выход Q являются числами с плавающей запятой.

Функции SIN, COS и TAN допускают широкий диапазон входных значений, где $-263 = 9.22*10^{18} < IN < +9.22*10^{18}$.

Функции ASIN и ACOS допускают узкий диапазон входных значений, такой, что $1 \le IN \le 1$. Получая достоверное значение в качестве параметра IN, функция ASIN_REAL вычислит результат Q следующего вида:

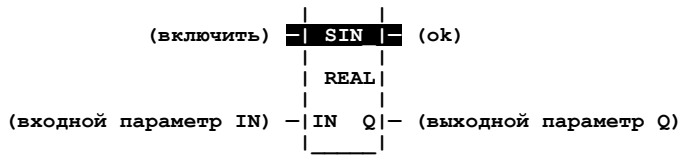
$$\mathbf{ASIN\ (IN)} = -\frac{\pi}{2} \le Q \le \frac{\pi}{2}$$

В свою очередь, функция ACOS_REAL вычислит результат Q следующего вида:

$$\mathbf{ACOS\ (IN)} = 0 \le Q \le \pi$$

Функция ATAN допускает широкий диапазон входных значений, таких, что $-\infty \le IN \le +\infty$. Получая значение из этого диапазона, функция ATAN_REAL вычислит следующий результат Q:

$$\mathbf{ATAN\ (IN)} = -\frac{\pi}{2} \le Q \le \frac{\pi}{2}$$



Примечание

Тригонометрические функции доступны только для процессора типа 352, а также 35x и 36x версии 9 и выше.

Параметры

Параметр	Описание
включить	Когда функция включена, соответствующая ей операция исполняется.
IN	IN содержит константу или ссылку величины, с которой выполняется операция.
OK	Выход OK запитывается, когда операция выполнена без переполнения, если не было недопустимых операций и/или вход IN равен NaN (не число).
Q	Выход Q содержит тригонометрическое значение IN.

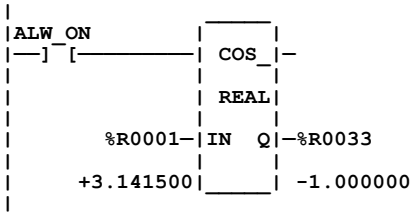
Допустимые типы памяти

Параметр	поток	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	пост.	нет
разрешить	•											
IN								•	•	•	•	
OK	•											•
Q								•	•	•		

• Допустимая ссылка или размещение, когда поток может проходить функцию.

Пример

В следующем примере, значение COS величины %R0001 помещается в результат, расположенный в %R0033.

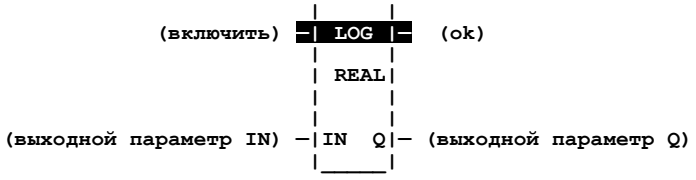


Логарифмические и экспоненциальные функции (LOG, LN, EXP, EXPT)

Функции LOG, LN и EXP имеют два входных и два выходных параметра. Когда функция принимает поток энергии, она выполняет соответствующую логарифмическую/ экспоненциальную операцию над действительным значением входа IN и передает результат на выход Q.

- Для функции LOG, на выход Q передается десятичный логарифм входа IN
- Для функции LN, на выход Q передается натуральный логарифм входа IN
- Для функции EXP, e возводится в степень, заданную входом IN
- Для функции EXPT, значение входа I1 возводится в степень, заданную значением I2, и передает результат на выход Q. (Эта функция имеет три входных параметра и два выходных)

Выход OK будет принимать поток энергии, кроме случаев, когда IN отрицательно или NaN (не число).



Параметры

Параметр	Описание
включить	Когда функция включена, соответствующая ей операция исполняется.
IN	IN содержит действительное значение, с которым выполняется операция.
OK	Выход OK обращается в 1, когда операция выполнена без переполнения, если не было недопустимых операций.
Q	Выход Q содержит логарифмическое/экспоненциальное значение IN.

Примечание

Функции LOG, LN, EXP и EXPT доступны только для процессора типа 352, а также 35x и 36x версии 9 и выше.

Примечание

Если входной параметр IN функции EXP имеет значение минус бесконечность (-∞), функция возвращает значение 0. В этом случае для ЦП модели 352 функция *не пропускает* поток энергии. Для всех остальных моделей ЦП Series 90-30 функция *пропускает* поток энергии, даже, если возвращает значение 0.

Допустимые типы памяти

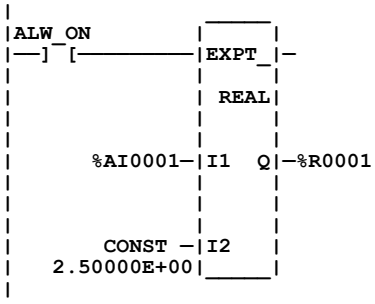
Параметр	поток	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	пост.	нет
включить	•											
IN*								•	•	•	•	
OK	•											•
Q								•	•	•		

• Допустимая ссылка или место, где поток проходит через данную функцию.

* Для функции EXPT вход IN замещается входными параметрами I1 и I2.

Пример

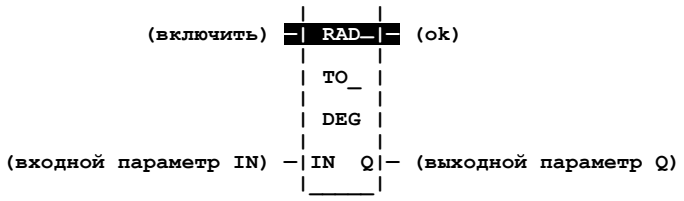
В следующем примере, значение величины %AI001 возводится в степень 2.5 и результат помещается в %R0001.



Преобразование радиан (RAD, DEG)

Когда эта функция принимает поток энергии, над действительными значениями входа IN производится соответствующее преобразование RAD_TO_DEG (радианы в градусы) или DEG_TO_RAD (градусы в радианы), и результат запоминается в выходе Q.

Выход OK будет принимать поток энергии, кроме случая, когда IN равен NaN (не число).



Параметры

Параметр	Описание
включить	Когда функция включена, соответствующая ей операция исполняется.
IN	IN содержит действительное значение, с которым выполняется операция.
OK	Выход OK устанавливается в 1, когда операция выполнена без переполнения, если IN не равен NaN (не число).
Q	Выход Q содержит преобразованное значение IN.

Примечание

Функции преобразования радиан доступны только для процессора типа 352, а также 35x и 36x версии 9 и выше.

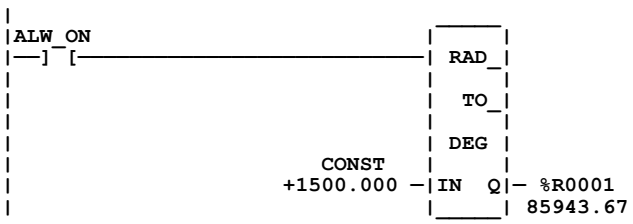
Допустимые типы памяти

Параметр	поток	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	пост.	нет
включить	•											
IN								•	•	•	•	
OK	•											•
Q								•	•	•		

• Допустимая ссылка или место, где поток проходит через данную функцию.

Пример

В следующем примере, значение +1500 преобразуется в градусы, и результат помещается в %R0001.



Глава 7

Функции отношений

Функции отношений используются для сравнения двух чисел. В настоящем разделе рассмотрены следующие функции отношений:

Сокращение	Функция	Описание	Стр.
EQ	Равно	Проверяет два числа на равенство	7-2
NE	Не равно	Проверяет два числа на неравенство	7-2
GT	Больше	Проверяет, является ли одно число большим другого	7-2
GE	Больше или равно	Проверяет, является ли одно число большим или равным другому	7-2
LT	Меньше	Проверяет, является ли одно число меньшим другого	7-2
LE	Меньше или равно	Проверяет, является ли одно число меньшим или равным другому	7-2
RANGE	Интервал	Определяет, входит ли число в заданный интервал (доступно только для ЦП выпуска 4.5 и старше)	7-4

Стандартные функции отношений (EQ, NE, GT, GE, LT, LE)

Функции отношений используются для определения отношений между двумя величинами. Когда функция принимает поток энергии, она сравнивает входной параметр I1 с входным параметром I2. Параметры должны относиться к одному типу данных. Функции отношений оперируют со следующими типами данных:

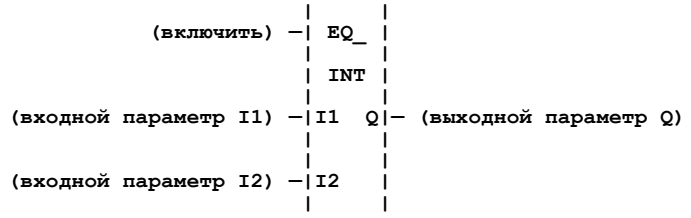
Тип данных	Описание
INT	Целое число со знаком
DINT	Целое число со знаком двойной разрядности
REAL	С плавающей запятой

Примечание

Тип данных REAL доступен только с ЦП 352. Кроме того, функция RANGE не приемлет данных типа REAL. Следует также отметить, что бит %S0020 устанавливается в состояние ON, когда функция отношения, использующая данные типа REAL, исполняется успешно. Он очищается, когда какой либо вход в состоянии NaN (не число).

По умолчанию тип данных относится к целочисленному знаковому. Чтобы сравнивать знаковые целые, знаковые целые двойной точности, или действительные величины, выберите новый тип данных после выбора функции. Для сравнения данных других типов, или данных различных типов, используйте сначала подходящую функцию преобразования (описанную в главе 11 “Функции преобразования”) для приведения данных к одному из перечисленных типов.

Если входные параметры соответствуют заданным соотношениям, выход Q устанавливается в состояние ON (1), в противном случае - в состояние OFF (0).



Параметры

Параметр	Описание
Включить	Когда функция включена, соответствующая ей операция исполняется.
I1	I1 содержит константу или ссылку первой величины, используемой в операции сравнения. (I1 помещается слева в математическом выражении, например I1<I2).
I2	I2 содержит константу или ссылку второй величины, используемой в операции сравнения. (I2 помещается справа в математическом выражении, например I1<I2).
Q	Выход Q возбуждается, когда входные параметры удовлетворяют заданному отношению.

Примечание

I1 и I2 должны быть допустимыми числами, то есть не могут быть NaN (не число).

Подробное описание

Функция	Описание
Равно	Когда функция включена, если значение на входе I1 равно значению на входе I2, выход Q возбуждается.
Не равно	Когда функция включена, если значение на входе I1 не равно значению на входе I2, выход Q возбуждается.
Больше чем	Когда функция включена, если значение на входе I1 больше чем значение на входе I2, выход Q возбуждается.
Больше чем или равно	Когда функция включена, если значение на входе I1 больше чем или равно значению на входе I2, выход Q возбуждается.
Меньше чем	Когда функция включена, если значение на входе I1 меньше чем значение на входе I2, выход Q возбуждается.
Меньше чем или равно	Когда функция включена, если значение на входе I1 меньше чем или равно значению на входе I2, выход Q возбуждается.

Допустимые типы памяти

Параметр	поток	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	пост.	нет
включить	•											
I1		o	o	o	o		o	•	•	•	•†	
I2		o	o	o	o		o	•	•	•	•†	
Q	•											•

- • Допустимая ссылка или место, где поток проходит через данную функцию.
- o Допустимая ссылка только для данных типа INT; не допустима для DINT и REAL.
- † Константы ограничены значениями в диапазоне от -32768 до +32767 для знаковых целочисленных операций двойной разрядности.

Пример

В следующем примере, два целых знаковых числа двойной разрядности, PWR_MDE и BIN_FUL сравниваются, когда устанавливается вход %I0001. Если PWR_MDE меньше или равно BIN_FUL, то обмотка %Q0002 устанавливается в 1.

Функция RANGE (INT, DINT, WORD)

Функция RANGE (Интервал) используется для определения, находится ли значение в интервале, заданном двумя числами.

Примечание

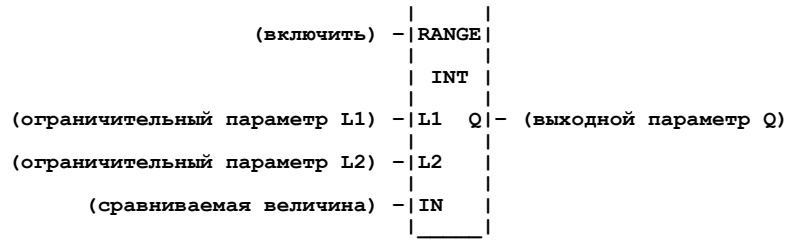
Эта функция **доступна только** для ЦП версии 4.41 и выше.

Функция RANGE оперирует со следующими типами данных:

Тип данных	Описание
INT	Целое число со знаком
DINT	Целое число со знаком двойной разрядности
WORD	Тип данных “Слово”

По умолчанию тип данных относится к целочисленному знаковому, и может быть изменен после выбора функции. Более подробная информация о типах данных имеется в главе 2, разделе 2 «Организация программы пользователя и данных».

Когда функция включена, блок функции RANGE будет сравнивать значение входного параметра IN с интервалом, заданным ограничительными параметрами L1 и L2. Если входной параметр находится в интервале, заданном ограничителями (включая их значения), то выходной параметр Q устанавливается в состояние ON (1), в противном случае в состояние OFF (0).



Примечание

Ограничительные параметры L1 и L2 представляют конечные точки диапазона. Дополнительных обозначений, присваивающих каждому параметру минимальное/максимальное или верхнее/нижнее содержательное значение, нет. Так, желаемый интервал от 0 до 100 можно задать, назначив параметру L1 значение 0, а параметру L2 значение 100, или наоборот.

Параметры

Параметр	Описание
включить	Когда функция включена, соответствующая ей операция исполняется.
L1	L1 содержит начальную точку интервала.
L2	L2 содержит конечную точку интервала.
IN	IN содержит значение, сравниваемое с заданным интервалом.
Q	Выход Q возбуждается, когда параметр IN находится в заданном интервале, включая граничные точки.

Допустимые типы памяти

Параметр	поток	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	пост.	нет
включить	•											
L1		o	o	o	o		o	•	•	•	•†	
L2		o	o	o	o		o	•	•	•	•†	
IN		o	o	o	o		o	•	•	•		
Q	•											•

• Допустимая ссылка или место, где поток проходит через данную функцию.

o Допустимая ссылка только для данных типа INT; не допустима для DINT и REAL.

† Константы ограничены значениями в диапазоне от -32768 до +32767 для знаковых целочисленных операций двойной разрядности

Пример 1

В следующем примере, вход %AI001 проверяется на нахождение в интервале, заданном двумя константами 0 и 100.

Таблица истинности функции RANGE				
Состояние включения %I0001	**Значение L1 Константа**	**Значение L2 Константа**	**Входное значение (IN) %AI001**	**Выходное состояние (Q) %Q0001**
ON	100	0	<0	OFF
ON	100	0	0-100	ON
ON	100	0	>100	OFF
OFF	100	0	Не имеет значения	OFF

Пример 2

В этом примере, вход %AI001 проверяется на вхождение в интервал, заданный двумя значениями регистров.

Таблица истинности функции RANGE				
Состояние включения %I0001	**Значение L1 %R0001**	**Значение L2 %R0002**	**Входное значение (IN) %AI001**	**Выходное состояние (Q) %Q0001**
ON	500	0	<0	OFF
ON	500	0	0-500	ON
ON	500	0	>5000	OFF
OFF	500	0	Не имеет значения	OFF

Глава 8

Функции битовых операций

Функции поразрядных (битовых) операций выполняют операции сравнения, логические операции и операции перемещения над итоговыми строками. Функции AND, OR, XOR и NOT работают с одиночным словом. Остальные функции операций с битами работают с несколькими словами при максимальной длине строки в 256 слов. Все функции поразрядных операций требуют применения типа данных WORD.

Хотя данные должны быть заданы с 16-битовыми приращениями, эти функции работают с данными, как с непрерывными строками битов, у которых первый бит первого слова является младшим значащим битом (LSB). Последний бит последнего слова является старшим значащим битом (MSB). Например, если вы задали три слова данных, начиная со ссылки %R0100, то функции будут оперировать с 48 непрерывными битами.

%R0100	**16**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1** ← **bit 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)

Примечание

Перекрытие областей адресов входных и выходных ссылок для функций, работающих с несколькими словами, может привести к непредсказуемым результатам.

В настоящем разделе описаны следующие функции бит-операций:

Сокращение	Функция	Описание	Стр.
AND	Логическое И	Если бит битовой строки I1 и соответствующий бит битовой строки I2 оба равны 1, то функция помещает 1 в соответствующее место выходной строки Q.	8-2
OR	Логическое ИЛИ	Если бит битовой строки I1 и/или соответствующий бит битовой строки I2 оба равны 1, то функция помещает 1 в соответствующее место выходной строки Q.	8-2
XOR	Логическое Исключающее ИЛИ	Если бит битовой строки I1 и соответствующий бит битовой строки I2 различны, то функция помещает 1 в соответствующее место выходной строки Q.	8-5
NOT	Логическое отрицание НЕ	Устанавливает каждый бит выходной строки Q в состояние, противоположное состоянию соответствующего бита битовой строки I1.	8-7
SHL	Сдвиг влево	Сдвигает все биты в слове или в строке слова влево на заданное число позиций.	8-9
SHR	Сдвиг вправо	Сдвигает все биты в слове или в строке слова вправо на заданное число позиций.	8-9
ROL	Циклический сдвиг влево	Циклически сдвигает все биты в строке влево на заданное число позиций.	8-12
ROR	Циклический сдвиг вправо	Циклически сдвигает все биты в строке вправо на заданное число позиций.	8-12
BTST	Проверка бита	Проверяет бит в битовой строке для определения, находится ли он в состоянии 1 или 0.	8-15
BSET	Установка бита	Устанавливает бит в битовой строке в 1.	8-17
BCLR	Очистка бита	Очищает бит, устанавливая его в 0.	8-17
BPOS	Позиция бита	Определяет положение бита битовой строки, установленного в 1.	8-19
MSKCMP	Маскированное сравнение	Сравнивает две различные битовые строки с возможностью маскирования выбранных битов (доступно для ЦП выпуска 4.5 и выше).	8-21

Функции AND и OR (WORD)

При каждом проходе, когда принимается поток энергии, функции AND и OR проверяют каждый бит битовой строки I1 и соответствующий бит битовой строки I2, начиная с младшего значащего бита в каждой.

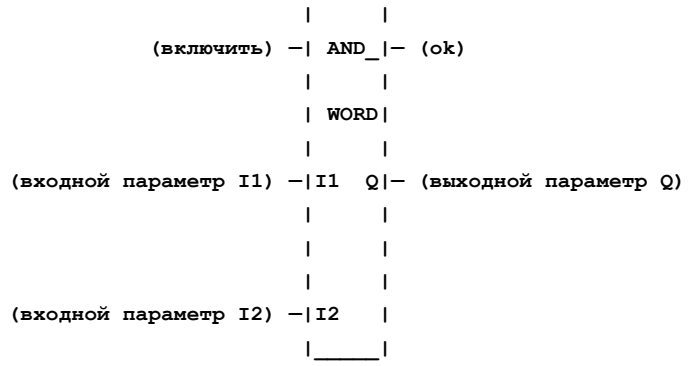
Если каждый из двух битов, проверяемых функцией AND, равен 1, то 1 помещается в соответствующую позицию выходной строки Q. Если какой либо или оба бита равны 0, то на соответствующее место выходной строки Q помещается 0.

Функция AND полезна для построения масок и экранов, когда пропускаются только определенные биты (те которые отличны от 1 битовой маски), а все остальные биты устанавливаются в 0. Функцию также можно использовать для очистки выбранной области памяти слов, путем применения функции AND к данным битам и битам другой строки, заведомо содержащей все 0. Задаваемые строки I1 и I2 могут перекрываться.

Если какой-либо или каждый из двух битов, проверяемых функцией OR, равен 1, то 1 помещается в соответствующую позицию выходной строки Q. Если оба бита равны 0, то на соответствующее место выходной строки Q помещается 0.

Функция OR полезна для комбинирования строк и управления многими выходами путем использования одной простой логической структуры. Функция эквивалентна двум параллельным контактам реле, умноженным на число битов битовой строки. Ее можно использовать для управления индикаторами непосредственно от состояний входов или реализации мигания индикаторов статуса.

Функции пропускают направо поток энергии, когда энергия принимается.



Параметры

Параметр	Описание
Включить	Когда функция включена, соответствующая ей операция исполняется.
I1	I1 содержит константу или ссылку первого слова первой строки..
I2	I2 содержит константу или ссылку второго слова второй строки.
OK	Выход возбуждается, когда возбуждается параметр Enable (Включить).
Q	Выход Q содержит результат операции.

Допустимые типы памяти

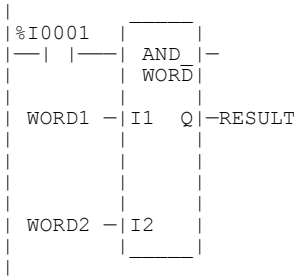
Параметр	поток	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	пост.	нет
включить	•											
I1		•	•	•	•	•	•	•	•	•	•	
I2		•	•	•	•	•	•	•	•	•	•	
Ok	•											•
Q		•	•	•	•	•†	•	•	•	•		

• Допустимая ссылка или место, где поток проходит через данную функцию.

† Только для %SA, %SB или %SC; %S использовать нельзя.

Пример

В следующем примере, каждый раз, когда устанавливается вход %I0001, 16 битовые строки, представленные мнемоническими именами WORD1 и WORD2, проверяются. Результат применения функции логическое И (AND) помещается в выходную строку RESULT (Результат).



WORD1	**0**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**1**	**0**	**0**	**0**
WORD2	**1**	**1**	**0**	**1**	**1**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**1**

RESULT	**0**	**0**	**0**	**1**	**1**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**

Функция XOR (WORD)

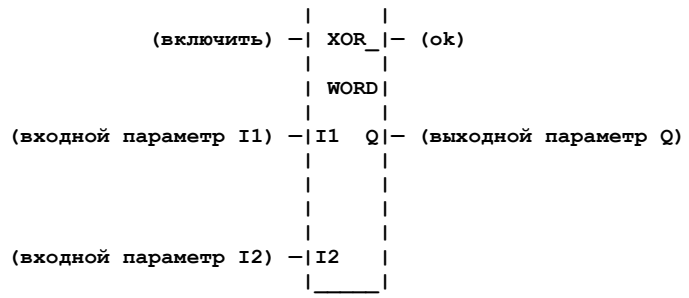
Функция XOR (ИСКЛЮЧАЮЩЕЕ ИЛИ) используется для сравнения каждого бита битовой строки I1 с соответствующим битом битовой строки I2. Если биты различны, то в соответствующую позицию выходной строки помещается 1.

При каждом проходе, когда принимается поток энергии, функция XOR проверяет каждый бит битовой строки I1 и соответствующий бит битовой строки I2, начиная с младшего значащего бита в каждой.

Если только один из двух битов, проверяемых функцией XOR, равен 1, то 1 помещается в соответствующую позицию выходной строки Q. Функция XOR пропускает направо поток энергии каждый раз, когда энергия поступает.

Если строка I2 и выходная строка Q начинаются у одной и той же ссылки, то 1, помещенная в строке I1, будет вызывать чередования состояния соответствующего бита в строке I2 при каждом цикле, пока поступает энергия. Более длинные циклы можно запрограммировать, возбуждая поток к функции дважды за желаемый интервал мигания; импульс потока энергии должен иметь продолжительность одного цикла (одноразовая обмотка или таймер с автосбросом).

Функция XOR полезна для быстрого сравнения двух битовых строк или для создания мигания группы битов с частотой один раз за два цикла.



Параметры

Параметр	Описание
включить	Когда функция включена, то соответствующая ей операция исполняется.
I1	I1 содержит константу или ссылку первого слова операции.
I2	I2 содержит константу или ссылку второго слова операции.
OK	Выход возбуждается, когда возбуждается параметр Enable (Включить).
Q	Выход Q содержит результат операции XOR, примененной к I1 и I2.

Допустимые типы памяти

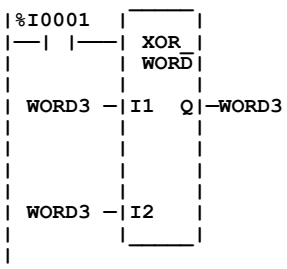
Параметр	поток	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	пост.	нет
включить	•											
I1		•	•	•	•	•	•	•	•	•	•	
I2		•	•	•	•	•	•	•	•	•	•	
OK	•											•
Q		•	•	•	•	•†	•	•	•	•		

- Допустимая ссылка или место, где поток проходит через данную функцию.

† Только для %SA, %SB или %SC; %S использовать нельзя.

Пример

В следующем примере, каждый раз, когда устанавливается вход %I0001, битовая строка, представленная мнемоническим именем WORD3, очищается (устанавливается на все нули).



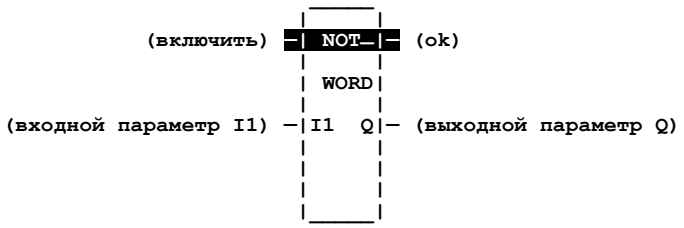
I1 (WORD3)	**0**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**1**	**0**	**0**	**0**
I2 (WORD3)	**0**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**1**	**0**	**0**	**0**

Q (WORD3)	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

Функция NOT (WORD)

Функция NOT (НЕ) используется для установки каждого бита выходной строки Q в состояние, противоположное состоянию соответствующего бита строки I1.

При каждом цикле все биты изменяются, делая выходную строку логическим дополнением строки I1. Функция пропускает направо поток энергии каждый раз, когда энергия поступает.



Параметры

Параметр	Описание
включить	Когда функция включена, то соответствующая ей операция исполняется.
I1	I1 содержит константу или ссылку слова для операции инвертирования.
OK	Выход возбуждается, когда возбуждается параметр Enable (Включить).
Q	Выход Q содержит результат операции инвертирования, примененной к I1.

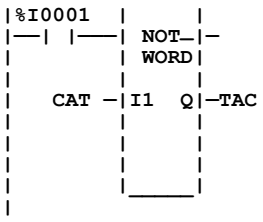
Допустимые типы памяти

Параметр	поток	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	пост.	нет
включить	•											
I1		•	•	•	•	•	•	•	•	•	•	
OK	•											•
Q		•	•	•	•	•†	•	•	•	•		

- • Допустимая ссылка или место, где поток проходит через данную функцию.
- † Только для %SA, %SB или %SC; %S использовать нельзя.

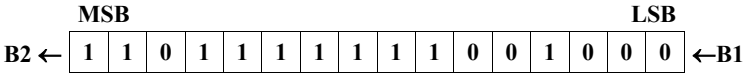
Пример

В следующем примере, каждый раз, когда устанавливается вход %I0001, битовая строка, представленная мнемоническим именем TAC, устанавливается в состояние, инверсное по отношению к битовой строке CAT.

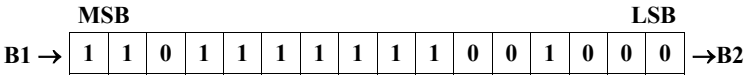


Функции SHL и SHR (WORD)

Функция SHL (СДВИГ ВЛЕВО) используется для сдвига всех битов слова или группы слов влево на заданное число позиций. При выполнении сдвига заданное число битов в левой части выходной строки сдвигается влево. Когда биты верхнего конца строки (MSB) перемещены за конец строки, такое же число битов вдвигается в строку на ее нижнем конце (LSB).



Функция SHR (СДВИГ ВПРАВО) используется для сдвига всех битов слова или группы слов вправо на заданное число позиций. При выполнении сдвига, заданное число битов в правой части выходной строки сдвигаются вправо. Когда биты нижнего конца строки (LSB) перемещаются за конец строки, такое же число битов вдвигается в строку на ее верхнем конце (MSB).



Для обеих функций могут быть выбраны стоки длиной от 1 до 256 слов.

Если число сдвигаемых битов (N) больше числа битов в массиве (LEN)∗16, или если число сдвигаемых битов равно 0, то массив Q заполняется копиями входного бита (B1), и входной бит копируется в выходной поток энергии (B2). Если число сдвигаемых битов равно 0, то сдвиг не выполняется; входной массив копируется в выходной массив и входной бит (B1) копируется в поток энергии.

Биты, сдвигаемые в начало строки, задаются входным параметром B1. Если в качестве числа сдвигаемых битов задана длина, превышающая 1, то каждый из битов заполняется одним и тем же значением (0 или 1). Этими значениями могут быть:

- Логический выход другой программной функции
- Все единицы. Чтобы сделать так, используйте специальное мнемоническое имя ссылки ALW_ON в качестве допустимого для входа B1
- Все нули. Чтобы сделать так, используйте специальное мнемоническое имя ссылки

Функции SHL и SHR пропускают поток энергии вправо, кроме случая, когда число заданных для сдвига битов равно нулю.

Выход Q является сдвинутой копией входной строки. Если вы хотите сдвинуть входную строку, то выходной параметр Q должен использовать то же место в памяти, что и входной параметр IN. Сдвинутая строка записывается полностью на каждом цикле, когда энергия принимается. Выход B2 является последним битом, выдвинутым за конец строки. Например, если было сдвинуто 4 бита, то B2 будет содержать четвертый выдвинутый бит.

Параметры

Параметр	Описание
включить	Когда функция включена, то соответствующая ей операция исполняется.
IN	IN содержит первое сдвигаемое слово.
N	N содержит число позиций, на которое сдвигается массив.
B1	B1 содержит значение бита, сдвигаемого в массив.
B2	B2 содержит значение последнего бита, выдвинутого из массива.
LEN	Число слов в сдвигаемом массиве.
Q	Выход Q содержит первое слово выдвинутого массива.

Допустимые типы памяти

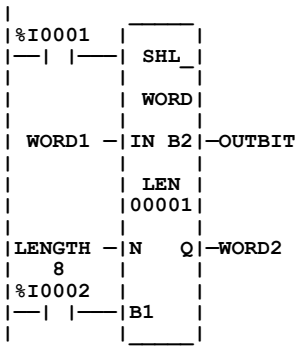
Параметр	поток	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	пост.	нет
включить	•											
IN		•	•	•	•	•	•	•	•	•		
N		•	•	•	•		•	•	•	•	•	
B1	•											
B2	•											•
Q		•	•	•	•	•†	•	•	•	•		

- Допустимая ссылка или место, где поток проходит через данную функцию.

† Только для %SA, %SB или %SC; %S использовать нельзя.

Пример

В следующем примере, каждый раз, когда устанавливается вход %I0001, выходная битовая строка, представленная мнемоническим именем WORD2, становится копией WORD1, сдвинутой влево на число бит, представленное именем LENGTH. Итоговые открывающие биты в начале выходной строки установлены к значению %I0002.



Функции ROL и ROR (WORD)

Функция ROL (Циклический сдвиг влево) используется для циклического сдвига всех битов строки влево на заданное число позиций. При выполнении циклического сдвига, заданное число битов в левой части входной строки сдвигаются влево и возвращаются в строку справа.

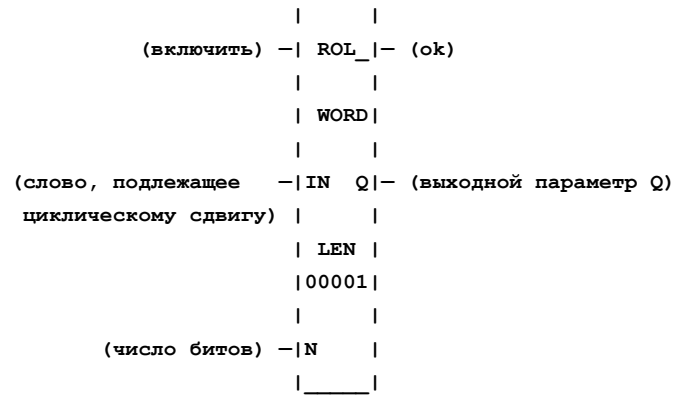
Функция ROR (Циклический сдвиг вправо) используется для циклического сдвига всех битов строки вправо на заданное число позиций. При выполнении циклического сдвига, заданное число битов в правой части входной строки сдвигаются вправо и возвращаются в строку слева.

Для обеих функций могут быть выбраны строки длиной от 1 до 256 слов.

Число позиций, на которое производится сдвиг, должно быть больше нуля и меньше числа битов в строке. В противном случае сдвиг не выполняется и поток не генерируется.

Функции ROL и ROR пропускают поток энергии вправо, кроме случая, когда число заданных для сдвига битов больше длины строки или меньше нуля.

Результат операции помещается в выходную строку Q. Если вы хотите сдвинуть входную строку, то выходной параметр Q должен использовать то же место в памяти, что и входной параметр IN. Сдвинутая строка записывается полностью на каждом цикле, когда принимается энергия.



Параметры

Параметр	Описание
включить	Когда функция включена, то соответствующая ей операция исполняется.
IN	IN содержит первое циклически сдвигаемое слово.
N	N содержит число позиций, на которое сдвигается массив.
OK	Выход OK возбуждается, когда возбуждается циклический сдвиг и когда длина сдвига не превышает размера массива.
Q	Выход Q содержит первое слово циклически сдвинутого массива.
LEN	Число слов в сдвигаемом массиве.

Допустимые типы памяти

Параметр	поток	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	пост.	нет
включить	•											
IN		•	•	•	•	•	•	•	•	•		
N		•	•	•	•		•	•	•	•	•	
OK	•											•
Q		•	•	•	•	•†	•	•	•	•		

- Допустимая ссылка или место, где поток проходит через данную функцию.

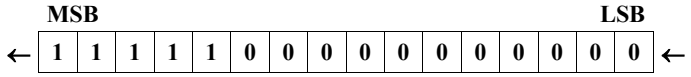
† Только для %SA, %SB или %SC; %S использовать нельзя.

Пример

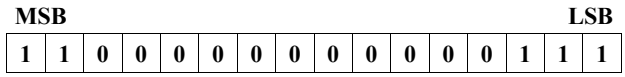
В следующем примере, каждый раз, когда устанавливается вход %I0001, входная битовая строка %R0001 циклически сдвигается на 3 бита и результат помещается в %R0002. По окончании операции входная битовая строка остается не измененной. Если для IN и Q используется одна и та же ссылка, будет происходить чередование.

```
|
|%I0001  |_____|
|—| |———| ROL |—
|         | WORD|
|         |     |
| %R0001—|IN  Q|—%R0002
|         |     |
|         | LEN |
|         |00001|
|         |     |
| CONST —|N    |
| +00003 |_____|
|
```

%R0001:



%R0001 (после того, как определена %I0001)

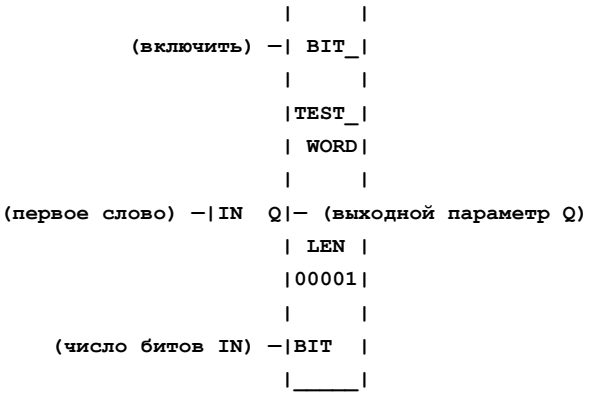


Функция BTST (WORD)

Функция BTST (ПРОВЕРКА БИТА) используется для проверки бита в битовой строке с целью определения текущего значения этого бита (1 или 0). Результат проверки помещается на выход Q.

При каждом цикле, когда принимается поток энергии, функция BTST устанавливает выход Q в то состояние, в котором находится проверяемый бит. Если не константа, а регистр используется для задания номера бита, то тот же функциональный блок может тестировать разные биты при последовательных циклах. Если величина BIT находится вне допустимого диапазона (1 < BIT < 16*LEN), то параметр Q устанавливается в состояние OFF (0).

Выбираемая длина строки может составлять от 1 до 256 слов.



Параметры

Параметр	Описание
включить	Когда функция включена, то соответствующая ей операция исполняется.
IN	IN содержит первое слово обрабатываемых данных.
BIT	Содержит номер проверяемого бита в IN. Допустимый диапазон 1< BIT <16*LEN.
OK	Выход OK возбуждается, когда возбуждается параметр Enable (Включить), и параметр BIT больше длины строки или равен нулю.
Q	Выход Q возбуждается, когда проверяемый бит равен 1.
LEN	Число слов в проверяемой строке.

Примечание

Когда используются функции Bit Test, Bit Set, Bit Clear или Bit Position, биты нумеруются с 1 до 16, ***а не с*** 0 до 15.

Допустимые типы памяти

Параметр	Поток	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	пост.	нет
включить	•											
IN		•	•	•	•	•	•	•	•	•		
N		•	•	•	•		•	•	•	•	•	
Q	•											•

- Допустимая ссылка или место, где поток проходит через данную функцию.

Пример

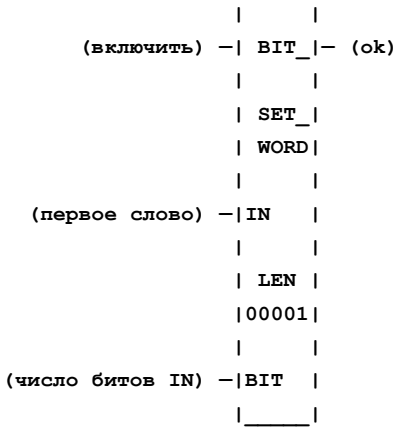
В следующем примере, каждый раз, когда устанавливается вход %I0001, проверяется бит в позиции, содержащийся в ссылке PICKBIT. Бит является частью строки PRD_CDE. Если бит равен 1, то выход Q пропускает поток энергии и обмотка %Q0001 запитывается.

Функции BSET и BCLR (WORD)

Функция BSET (УСТАНОВКА БИТА) используется для установки бита битовой строки в состояние 1.Функция BCLR (ОЧИСТКА БИТА) используется для очистки бита битовой строки путем установки его в состояние 0.

При каждом цикле, когда принимается поток энергии, функция устанавливает заданный бит в состояние 1 для функции BSET и в состояние 0 для функции BCLR. Если не константа, а переменная (регистр) используется для задания номера бита, то тот же функциональный блок может устанавливать разные биты при последовательных циклах. Функция пропускает поток энергии вправо, кроме случая, когда величина BIT находится вне допустимого диапазона (1 < BIT < 16∗LEN). При этом параметр OK устанавливается в состояние OFF.

Выбираемая длина строки может составлять от 1 до 256 слов.



Параметры

Параметр	Описание
включить	Когда функция включена, то соответствующая ей операция исполняется.
IN	IN содержит первое слово обрабатываемых данных.
BIT	Содержит номер устанавливаемого бита в IN. Допустимый диапазон 1< BIT <16∗LEN.
OK	Выход OK возбуждается, когда возбуждается параметр Enable (Включить).
LEN	Число слов в проверяемой строке.

Примечание

При использовании функций Bit Test, Bit Set, Bit Clear или Bit Position, биты нумеруются с 1 до 16, ***а не с*** 0 до 15.

Допустимые типы памяти

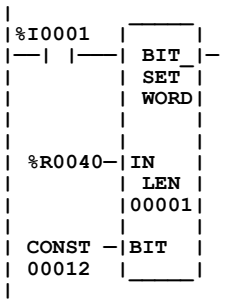
Параметр	поток	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	пост.	нет
включить	•											
IN		•	•	•	•	†	•	•	•	•		
BIT		•	•	•	•		•	•	•	•	•	
OK	•											•

- Допустимая ссылка или место, где поток проходит через данную функцию.

† Только для %SA, %SB или %SC; %S использовать нельзя.

Пример

В следующем примере, каждый раз, когда устанавливается вход %I0001, бит 12 строки, начинающейся со ссылки %R0040, устанавливается в состояние 1.



Функция BPOS (WORD)

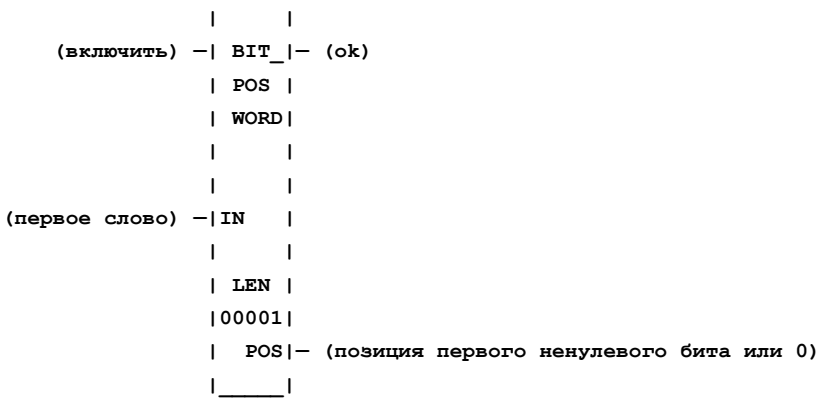
Функция BPOS (ПОЗИЦИЯ БИТА) используется для определения местоположения бита в состоянии 1 в битовой строке.

На каждом цикле, когда принимается поток энергии, функция сканирует строку, начиная с IN. Функция прекращает работу, когда обнаружен бит в состоянии 1, или просканирована вся строка.

Функция устанавливает позицию в битовой строке с первым ненулевым битом; если ненулевой бит не обнаружен, то функция устанавливается к нулю.

Функция пропускает поток энергии вправо, когда параметр Enable (Включить) находится в состоянии ON.

Выбираемая длина строки может составлять от 1 до 256 слов.



Параметры

Параметр	Описание
включить	Когда функция включена, то соответствующая ей операция исполняется.
IN	IN содержит первое слово обрабатываемых данных.
OK	Выход OK возбуждается, когда возбуждается параметр Разрешить.
POS	Позиция первого обнаруженного ненулевого бита или нуль, если ненулевой бит не обнаружен.
LEN	Число слов в проверяемой строке.

Примечание

Когда используются функции Bit Test, Bit Set, Bit Clear или Bit Position, биты нумеруются с 1 до 16, ***а не с*** 0 до 15.

Допустимые типы памяти

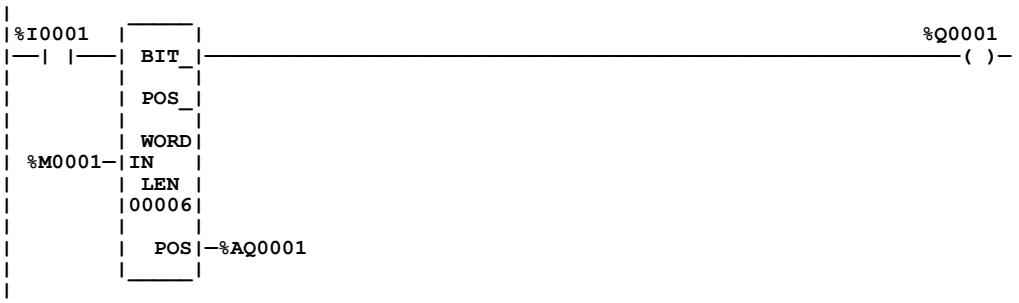
Параметр	поток	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	пост.	нет
включить	•											
IN		•	•	•	•	†	•	•	•	•		
POS		•	•	•	•		•	•	•	•	•	
OK	•											•

- Допустимая ссылка или место, где поток проходит через данную функцию.

† Только для %SA, %SB или %SC; %S использовать нельзя.

Пример

В следующем примере каждый раз, когда устанавливается вход %I0001, просматривается битовая строка, начинающаяся с %M0001, пока не будет обнаружен бит в состоянии 1, или пока не будет просмотрено 6 слов. Обмотка %Q0001 запитывается. Если единичный бит обнаружен, то его положение в битовой строке записывается в %AQ0001. Если %I0001 установлен, %M0001 равен 0 и бит %M0002 равен 1, то записанное в %AQ0001 значение равно 2.



Функция MSKCMP (WORD, DWORD)

Функция MSKCMP (МАСКИРУЕМОЕ СРАВНЕНИЕ) (доступная для ЦП выпусков 4.41 и более поздних) используется для сравнения содержимого двух различных битовых строк с возможностью маскирования выбранных битов. Длина сравниваемых битовых строк задается параметром LEN (значение LEN задает число 16-битовых слов для функции MSKCMPW и число 32-битовых слов для функции MSKCMPD).

Когда логика, управляющая входом включения функции, пропускает поток энергии к входу Enable (Включить), функция начинает сравнение битов первой строки с соответствующими битами второй строки. Сравнение продолжается до обнаружения несоответствия или до достижения конца строки.

Вход BIT используется для хранения номера бита, где должно начинаться следующее сравнение (0 указывает на первый бит строки). Выход BN используется для хранения номера бита, где было выполнено последнее сравнение (1 указывает на первый бит строки). Использование одной и той же ссылки для BIT и BN приводит к старту сравнения со следующей позиции по отношению к позиции, где обнаружено несоответствие; или, если все биты успешно прошли сравнение, то при следующем вызове функционального блока сравнение начнется сначала.

Если вы хотите запустить сравнение с некоторой другой позиции строки, то вы можете ввести разные ссылки для BIT и BN. Если значение BIT расположено за концом строки, то параметр BIT устанавливается в 0 перед запуском следующего сравнения.

Если все биты в I1 и I2 одинаковы

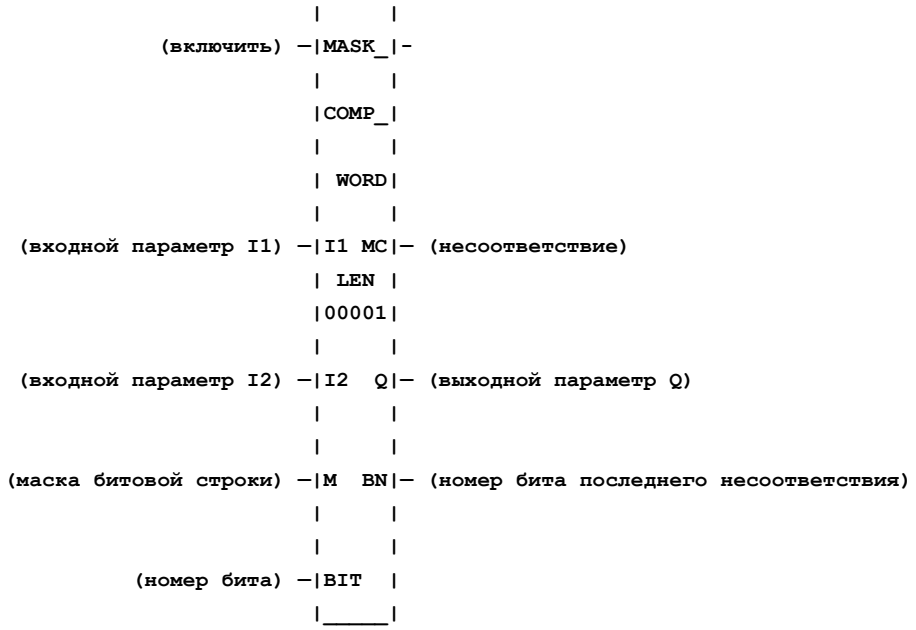
Если все соответствующие биты в строках совпадают, то функция устанавливает выход MC (несоответствие) в состояние 0, а параметр BN к наибольшему номеру бита входных строк. После этого сравнение останавливается. При следующем вызове MSKCMPW он будет установлен к 0.

Если обнаружено несоответствие

Когда два текущих бита не совпадают, функция проверяет соответственно нумерованный бит в строке M (маска). Если бит маски равен 1, то сравнение продолжается до достижения следующего несовпадения или до конца входных строк.

Если обнаружено несоответствие и соответствующий бит маски равен 0, то функция выполняет следующие действия:

1. Устанавливает соответствующий бит маски на 1.
2. Устанавливает выход MC на 1.
3. Обновляет выходную строку Q до соответствия с новым содержанием строки маски M.
4. Устанавливает номер выходного бита BN, равным номеру несоответствующего бита.
5. Останавливает сравнение.



Параметры

Параметр	Описание
включить	Когда функция включена, то соответствующая ей операция исполняется.
I1	Ссылка на первую сравниваемую строку.
I2	Ссылка на вторую сравниваемую строку.
M	Ссылка на битовую строку маски.
BIT	Ссылка на номер бита, с которого начнется следующее сравнение.
MC	Логика пользователя для определения наличия несоответствия.
Q	Выход копирует битовую строку маски.
BN	Номер последнего бита, прошедшего сравнение.
LEN	Число слов в проверяемой строке.

Допустимые типы памяти

Параметр	поток	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	пост.	нет
включить	•											
I1		o	o	o	o	o	o	•	•	•		
I2		o	o	o	o	o	o	•	•	•		
M		o	o	o	o	o†	o	•	•	•		
BIT		•	•	•	•	•	•	•	•	•	•	
LEN											•‡	
MC	•											•
Q		o	o	o	o	o†	o	•	•	•		
BN		•	•	•	•	•	•	•	•	•		

- • Допустимая ссылка или место, где поток проходит через данную функцию.
- o Допустимая ссылка только для данных типа WORD, не допускается для DWORD.
- † Только для %SA, %SB или %SC; %S использовать нельзя.
- ‡ Максимальное значение константы 4095 для WORD и 2047 для DWORD.

Пример

В следующем примере, после первого сканирования исполнен функциональный блок MSKCMP. %M0001 – %M0016 сравнивался с %M0017 – %M0032. %M0033 – %M0048 содержит значение маски. Значение %R0001 определяет, с какой битовой позиции начинается сравнение двух битовых строк. Содержание перечисленных выше ссылок перед исполнением функционального блока имеет вид:

(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

Содержание этих же ссылок после исполнения функционального блока имеет вид:

(I1) – %M0001 =

0	1	1	0	1	1	0	0	0	1	1	0	1	1	0	0

(I2) – %M0017 =

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

Представление многозвенной диаграммы

Обратите внимание, что в приведенном выше примере, мы использовали контакт FST_SCN для запуска одного и только одного исполнения; в противном случае маскированное сравнение будет повторено и не обязательно приведет к желаемым результатам.

Глава
9

Функции пересылки данных

Функции перемещения данных предоставляют базовые возможности по перемещению данных. В настоящем разделе рассмотрены следующие функции перемещения данных:

Сокращение	Функция	Описание	Стр.
MOVE	Перемещение	Копирует данные как отдельные биты. Максимальная допустимая длина составляет 256 слов, за исключением функции MOVE_BIT, для которой допустимо 256 бит. Данные могут быть перемещены в различные типы данных без предварительного преобразования.	9-2
BLKMOV	Перемещение блока	Копирует блок из семи констант в заданное место памяти. Константы вводятся как часть функции.	9-5
BLKCLR	Очистка блока	Замещает содержание блока данных нулями. Эта функция может использоваться для очистки памяти битов (%I,%Q,%M,%G,%T) или памяти слов (%R,%AI,%AQ). Максимальная длина 256 слов.	9-8
SHFR	Регистр сдвига	Сдвигает одно и более слов данных в таблицу. Максимальная допустимая длина 256 слов.	9-10
BITSEQ	Сортировщик битов	Выполняет сдвиг последовательности битов по массиву битов. Максимальная допустимая длина 256 слов.	9-13
COMMREQ	Запрос связи	Позволяет программе связываться с интеллектуальными модулями, например модулем программируемого сопроцессора или модулем связи Genius.	9-17

Функция MOVE (BIT, INT, WORD, REAL)

Используйте функцию MOVE (Переместить) для копирования данных (как индивидуальных битов) из одного места в другое. Так как данные копируются в битовом формате, новое место не обязательно должно соответствовать типу данных места, откуда производится копирование.

Функция MOVE имеет два входных и два выходных параметра. Когда функция принимает поток энергии, она копирует данные из входного параметра IN в выходной параметр Q, рассматривая их как биты. Если данные перемещены из одного места (ячейки) дискретной памяти в другое (например, из памяти %I в память %T), то информация передачи, связанная с элементами дискретной памяти, обновляется, чтобы индицировать, привела или нет операция MOVE к изменению состояния каких-либо элементов дискретной памяти. Данные входного параметра не изменяются, если нет перекрытия источника и места назначения данных.

Иной подход имеет место для данных типа BIT. Если массив BIT, заданный для параметра Q, не включает все биты байта, то биты перехода, связанные с байтом (которые массиву не принадлежат), будут очищены, когда функция MOVE_BIT принимает поток энергии.

Вход IN может быть либо ссылкой на перемещаемые данные, либо константой. Если задана константа, то ее значение помещается в ячейке, задаваемой выходной ссылкой. Например, если константа величиной 4 задана для входа IN, то 4 размещается в ячейке памяти, указанной Q. Если заданная длина превышает 1 и задается константа, то она размещается в ячейке памяти, определяемой Q, и всех последующих ячейках, в соответствии с заданной длиной. Например, если константа величиной 9 задана для IN и имеет длину 4, то 9 разместится в ячейке памяти, определенной Q, и в трех последующих ячейках.

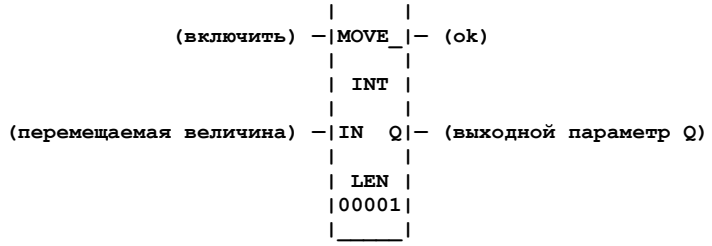
Операнд LEN задает число:

- перемещаемых слов для функций MOVE_INT и MOVE_WORD
- перемещаемых битов для функции MOVE_BIT
- перемещаемых действительных величин для функции MOVE_REAL

Примечание

Тип данных REAL доступен только для ЦП модели 352.

Функция пропускает поток энергии направо, когда принимается поток энергии.



Параметры

Параметр	Описание
включить	Когда функция включена, то соответствующая ей операция исполняется.
IN	IN содержит перемещаемое значение. Для функции MOVE_BIT можно использовать любую дискретную ссылку; она может не быть байт-выравненной. Тем не менее, 16 бит, начиная с заданного адреса ссылки, оперативно отображаются.
OK	OK выход возбуждается всегда, когда функция включена.
Q	Когда перемещение выполнено, то значение IN записывается в Q. Для функции MOVE_BIT можно использовать любую дискретную ссылку; она может не быть байт-выравненной. Тем не менее, 16 бит, начиная с заданной адресной ссылки, оперативно отображаются.
LEN	Параметр LEN задает число перемещаемых слов или бит. Для функций MOVE_WORD и MOVE_INT, значение LEN должно быть между 1 и 256 словами. Для функции MOVE_BIT, когда IN константа, LEN должно быть между 1 и 256 битами, иначе говоря LEN должно быть между 1 и 256.

Примечание

Для ЦП моделей 351 и 352 функции MOVE_INT и MOVE_WORD не работают должным образом, если вы допускаете наложение IN и Q параметров.

Также имейте в виду, что только ЦП 352 и старше поддерживают выполнение функции MOVE_REAL.

Допустимые типы памяти

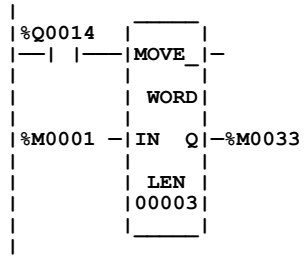
Параметр	поток	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	пост.	нет
включить	•											
IN		•	•	•	•	o	•	•	•	•	•	
OK	•											•
Q		•	•	•	•	o↑	•	•	•	•		

Замечание: Для данных типа REAL допустимыми типами являются %R, %AI и %AQ.

- • Допустимая ссылка для BIT, INT или WORD данных, или место, где поток проходит через данную функцию.
- o допустимая ссылка только для данных типа BIT или WORD; не допустима для данных типа INT.
- ↑ Только для %SA, %SB, и %SC; нельзя использовать %S.

Пример 1

Когда включенный вход %Q0014 находится в состоянии ON, 48 бит перемещаются из ячейки памяти %M0001 в ячейку памяти %M0033. Даже если место назначения перекрывает источник на протяжении 16 бит, перемещение выполняется корректно (кроме ЦП моделей 351 и 352, как отмечалось ранее).



До применения функции MOVE:

INPUT (%M0001 … %M0048)

1

%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 применения функции MOVE:

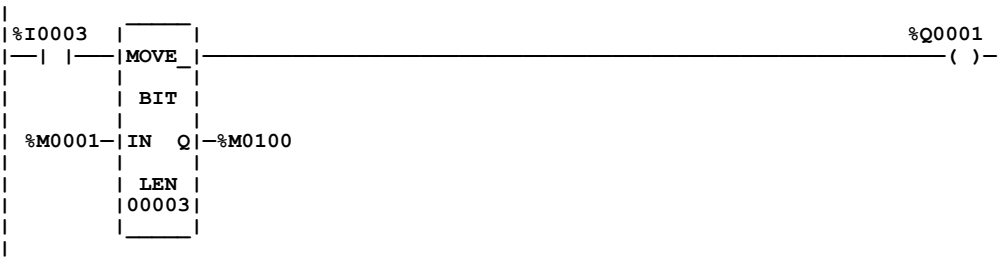
INPUT (%M0033 … %M0080)

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

Пример 2

В следующем примере, когда устанавливается %I0001, три бита %M0001, %M0002 и %M0003 перемещаются в %M0100, %M0101 и %M0102 соответственно. Обмотка %Q0001 запитывается.



Функция BLKMOV (INT, WORD, REAL)

Используйте функцию BLOCK MOVE (Переместить блок) для копирования блока из семи констант в заданное место.

Примечание

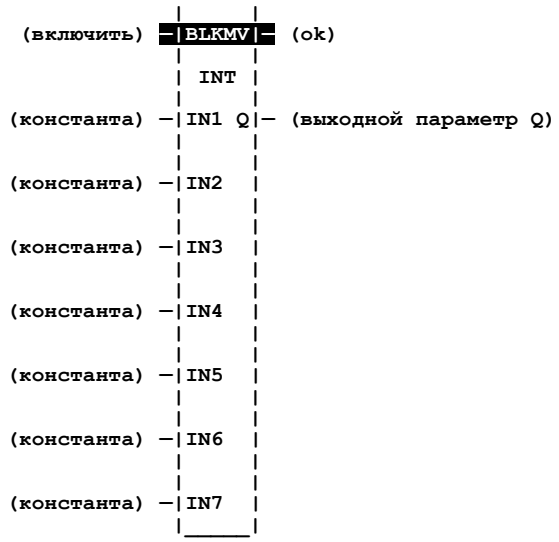
Тип данных REAL доступен только для ЦП модели 352.

Функция BLKMOV имеет восемь входных и два выходных параметра. Когда функция принимает поток энергии, она копирует значения констант в последовательные ячейки памяти, начиная с адреса назначения, заданного в выходе Q. Выход Q не может одновременно являться входом другой программной функции.

Примечание

Для функции BLKMOV_INT значения IN1-IN7 отображаются как десятичные числа со знаком. Для функции BLKMOV_WORD значения IN1-IN7 отображаются в шестнадцатеричном представлении. Для функции BLKMOV_REAL значения IN1-IN7 отображаются в формате действительных чисел.

Функция пропускает поток энергии направо каждый раз, когда принимается поток энергии.



Параметры

Параметр	Описание
включить	Когда функция включена, выполняется перемещение блока.
IN1-IN7	Все IN содержат значения констант.
OK	Выход OK возбуждается всегда, когда функция включена.

Допустимые типы памяти

Параметр	поток	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	пост.	нет
включить	•											
IN1-IN7											•	
OK	•											•
Q		•	•	•	•	o↑	•	•	•	•		

Замечание: Для данных типа REAL допустимыми типами являются только %R, %AI и %AQ.

- Допустимая ссылка или место, где поток проходит через данную функцию.

o Допустимая ссылка только для данных типа WORD; не допустима для данных типа INT или REAL.

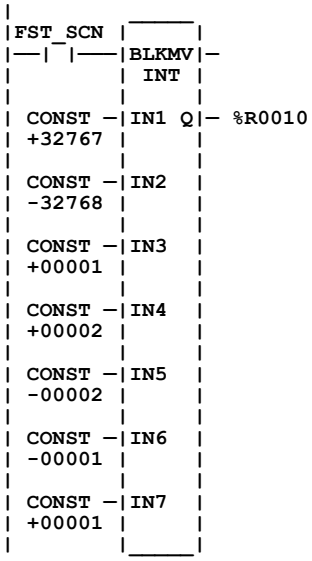
↑ Только для %SA, %SB, и %SC; нельзя использовать %S.

Примечание

ЦП модели 352 является в настоящее время единственным ЦП с плавающей запятой в Series 90-30, и только ему доступна функция BLCMOV_REAL.

Пример

В следующем примере, когда включенный вход, представленный мнемоническим именем FST_SCN, установлен в состояние ON, функция BLKMOV копирует семь входных констант в ячейки памяти с %R0010 по%R0016.

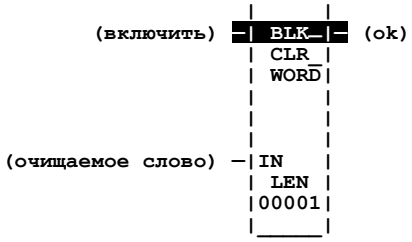


Функция BLKCLR (WORD)

Используйте функцию BLOCK CLEAR (Очистить блок) для заполнения заданного блока данных нулями.

Функция BLKCLR имеет два входных и один выходной параметр. Когда функция принимает поток энергии, она записывает нули в ячейки памяти, начиная со ссылки, заданной входом IN. Когда очищаемые данные размещены в дискретной памяти (%I, %Q, %M, %G, %T), информация перехода, связанная со ссылками, тоже очищается.

Функция пропускает поток энергии направо каждый раз, когда принимается поток энергии.



Параметры

Параметр	Описание
включить	Когда функция включена, массив очищается.
IN	IN содержит первое слово очищаемого массива.
OK	Выход OK возбуждается всегда, когда функция включена.
LEN	LEN должна быть в диапазоне от 1 до 256 слов.

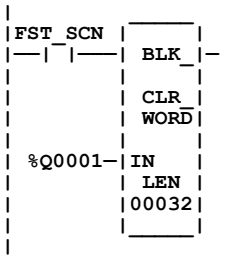
Допустимые типы памяти

Параметр	поток	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	пост.	нет
включить	•											
IN1-IN7		•	•	•	•	•↑	•	•	•	•		
OK	•											•

- • Допустимая ссылка или место, где поток проходит через данную функцию.
- ↑ Только для %SA, %SB, и %SC; нельзя использовать %S.

Пример

В следующем примере при включении питания 32 слова %Q памяти (512 элементов), начиная с %Q0001, заполняются нулями.



Функция SHFR (BIT, WORD)

Используйте функцию SHIFT REGISTER (Сдвиговый регистр) для сдвига одного и более слов или битов данных из ссылочной ячейки памяти в заданную область памяти. Например, одно слово может быть сдвинуто в область памяти длиной пять слов. В результате такого сдвига другое слово будет выдвинуто с конца области памяти.

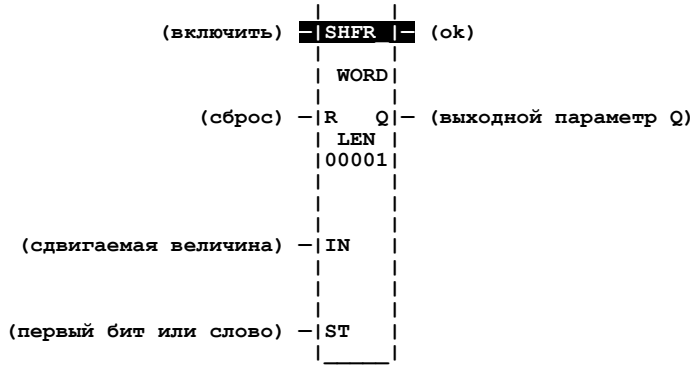
Примечание

При назначении ссылочных адресов наложение ссылочных адресов входа и выхода в многословных функциях может привести к непредсказуемым результатам

Функция SHFR имеет четыре входных и два выходных параметра. Вход сброса R имеет преимущество перед входом включения функции. Когда сброс активен, все ссылки от сдвигового регистра ST до длины, заданной параметром LEN, заполняются нулями.

Когда функция принимает поток энергии и параметр сброса активен, каждый бит слова сдвигового регистра перемещается в следующую по порядку ссылку. Последний элемент в сдвиговом регистре перемещается в Q. Наивысшая ссылка элемента сдвигового регистра параметра IN смещается в удаляемый элемент, начиная с ST. Содержание сдвигового регистра доступно всей программе, так как оно размещается в безусловных ячейках логически адресуемой памяти.

Функция пропускает поток энергии направо, когда принимается поток через включенную логику.



Параметры

Параметр	Описание
включить	Когда функция включена и сброс R не активен, выполняется сдвиг.
R	Когда R активен, сдвиговый регистр, расположенный в ST, заполняется нулями.
IN	IN содержит перемещаемое значение. Для функции SHFR_BIT можно использовать любую дискретную ссылку; она может не быть байт-выровненной. Тем не менее, 16 бит, начиная с заданного адреса ссылки, оперативно отображаются.
ST	ST содержит первый бит или слово сдвигового регистра. Для функции SHFR_BIT можно использовать любую дискретную ссылку; она может не быть байт-выровненной. Тем не менее, 16 бит, начиная с заданного адреса ссылки, оперативно отображаются.
OK	Выход OK возбуждается всегда, когда функция включена и параметр R не активен.
Q	Выход Q содержит бит или слово, выдвигаемое регистром. Для функции SHFR_BIT можно использовать любую дискретную ссылку; она может не быть байт-выровненной. Тем не менее, 16 бит, начиная с заданного адреса ссылки, оперативно отображаются.
LEN	Параметр LEN задает длину сдвигового регистра. Для функции SHFR_WORD, значение LEN должно быть между 1 и 256 словами. Для функции SHFR_BIT, LEN должно быть между 1 и 256 битами.

Допустимые типы памяти

Параметр	поток	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	пост.	нет
включить	•											
R	•											
IN		•	•	•	•	•	•	•	•	•	•	
ST		•	•	•	•	•↑	•	•	•	•		
OK	•											•
Q		•	•	•	•	•↑	•	•	•	•		

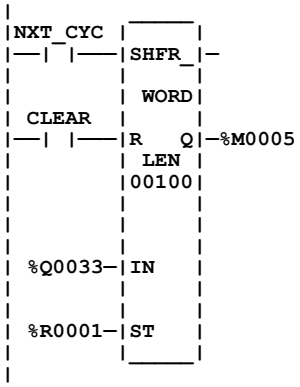
- Допустимая ссылка для BIT или WORD данных или место, где поток может проходить через данную функцию. Для функции SHFR_BIT дискретные ссылки пользователя %I, %Q, %M, и %T могут не быть байт-выровненными.

↑ Только для %SA, %SB; %S использовать нельзя.

Пример 1

В следующем примере сдвиговый регистр работает на ячейках памяти, расположенных от %R0001 до %R0100. Когда ссылка сброса CLEAR активна, слова сдвигового регистра устанавливаются в нуль.

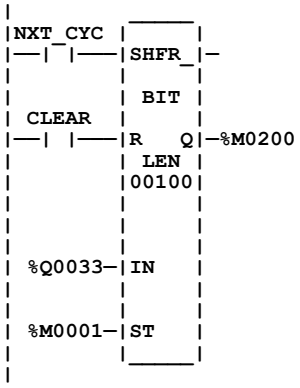
Когда активна ссылка NXT_CYC и ссылка CLEAR не активна, то слово таблицы статуса выхода из ячейки %Q0033 сдвигается в сдвиговый регистр со ссылкой %R0001. Слово, выдвигаемое из сдвигового регистра из %R0100, сохраняется в выходе %M0005.



Пример 2

В следующем примере сдвиговый регистр работает на ячейках памяти, расположенных от %M0001 до %M0100. Когда ссылка сброса CLEAR активна, функция заполняет эти ячейки памяти нулями.

Когда активна ссылка NXT_CYC и ссылка CLEAR не активна, функция SHFR сдвигает данные из ячейки %M0001 в %M0100 по одному биту. Бит в %Q0033 сдвигается в %M0001, а бит, выдвигаемый из %M0100, записывается в %M0200.



Функция BITSEQ (BIT)

Используйте функцию BIT SEQUENCER (Сортировщик битов) для сдвига последовательности битов через массив битов. Функция BITSEQ имеет пять входных параметров и один выходной параметр. Действие функции зависит от предыдущего значения параметра EN, как показано в следующей таблице.

R (текущего исполнения)	EN (предыдущего исполнения)	EN(текущего исполнения)	Действия сортировщика битов
OFF	OFF	OFF	Сортировщик не исполняется.
OFF	OFF	ON	Сортировщик увеличивается/уменьшается на 1
OFF	ON	OFF	Сортировщик не исполняется.
OFF	ON	ON	Сортировщик не исполняется.
ON	ON/OFF	ON/OFF	Сортировщик сбрасывается.

Вход сброса (R) превалирует над включением (EN) и всегда сбрасывает сортировщик. Когда параметр сброса активен, текущий номер шага устанавливается к значению, переданному через параметр номера шага. Если никакого номера шага не передано, то шаг устанавливается равным 1. Все биты сортировщика устанавливаются на нуль, кроме бита, указываемого текущим шагом, который устанавливается в состояние 1.

Когда параметр EN активен и параметр R неактивен, бит, указываемый текущим номером шага, очищается. Текущий номер шага может либо увеличиваться, либо уменьшаться, в зависимости от параметра направления. Затем бит, указываемый новым номером шага, устанавливается равным 1.

- Когда номер шага увеличивается и выходит за пределы диапазона (1<Номер шага<LEN), он снова устанавливается равным 1.
- Когда номер шага уменьшается и выходит за пределы диапазона (1<Номер шага<LEN), он снова устанавливается равным значению LEN.

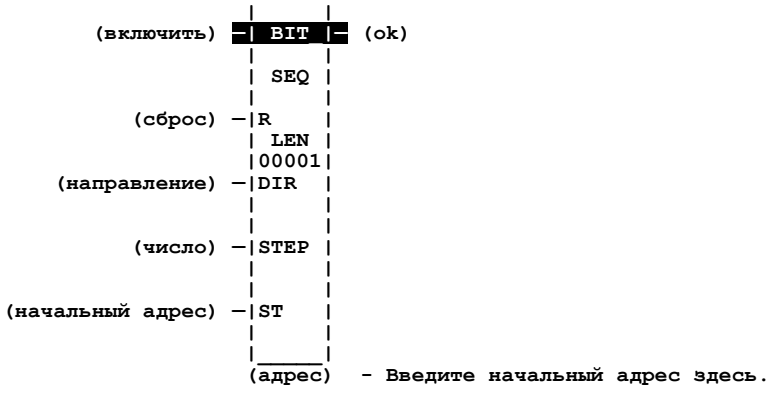
Параметр ST необязательный. Если он не используется, то функция работает, как описано выше, но биты не устанавливаются и не обнуляются. По существу, функция только циклически прогоняет текущий номер шага по разрешенному диапазону.

Память, необходимая сортировщику

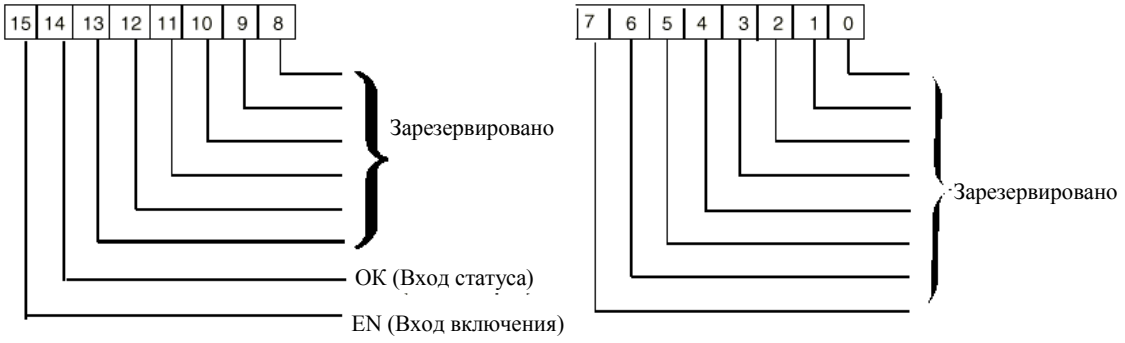
Каждый сортировщик битов использует три слова (регистра) памяти %R для хранения следующей информации:

Номер текущего шага	Слово 1
Длина последовательности (в битах)	Слово 2
Управляющее слово	Слово 3

При вводе последовательности битов вы должны ввести начальный адрес для каждого из этих трех слов (регистров) прямо под графическим представлением функции (см. следующий пример)



Управляющее слово хранит состояние логических входов и выходов, ассоциированных с ним логических блоков в соответствии со следующим форматом:



Примечание

Биты с нулевого до тринадцатого не используются.

Параметры

Параметр	Описание
адрес	Адрес – это местоположение текущего шага сортировщика битов, длины и последних состояний параметров OK и Enable (Включить).
включить	Когда функция включена, и если она не была включена на предыдущем цикле и если параметр сброса R не активен, то выполняется сдвиг последовательности битов.
R	Когда R активен, то номер шага сортировщика битов устанавливается равным значению параметра STEP (по умолчанию=1), и сортировщик битов заполняется нулями, за исключением бита текущего номера шага..
DIR	Когда параметр DIR активен, номер шага сортировщика битов последовательно увеличивается при сдвиге. В противном случае - уменьшается.
STEP	Когда параметр R активен, номер шага устанавливается равным данному значению.
ST	ST содержит первое слово сортировщика битов.
OK	Выход OK возбуждается всегда, когда функция включена.
LEN	Параметр LEN должен быть в диапазоне между 1 и 256 битами.

Примечание

Проверка обмотки для функции BITSEQ, осуществляет проверку всех 16 битов параметра ST, даже если параметр LEN меньше 16.

Допустимые типы памяти

Параметр	поток	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	пост.	нет
адрес								•				
включить	•											
R	•											
DIR	•											
STEP		•	•	•	•		•	•	•	•	•	•
ST		•	•	•	•	•↑	•	•	•	•		•
OK	•											•

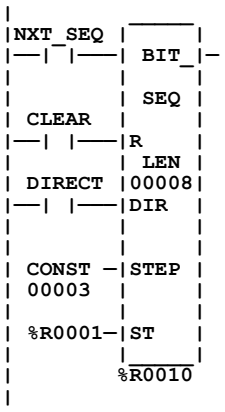
• Допустимая ссылка или место, где поток может проходить через данную функцию.

↑ Только для %SA, %SB; %S не может использоваться.

Пример

В следующем примере сортировщик работает на регистре памяти %R0001. Его статические данные хранятся в регистрах памяти %R0100, %R0011 и %R0012. Когда параметр сброса CLEAR активен, то сортировщик сбрасывается, и номер текущего шага устанавливается равным номеру шага 3. Первые 8 битов %R0001 устанавливаются равными нулю.

Когда активна ссылка NXT_SEQ и ссылка CLEAR не активна, бит шага номер 3 очищается и устанавливается бит шага номер 2 или номер 4, в зависимости от состояния параметра DIR.



Функция COMMREQ

Используйте функцию COMMUNICATION REQUEST (Запрос связи), если программе необходима связь с интеллектуальным модулем, например, модулем программируемого сопроцессора или модулем связи Genius.

Примечание

Информация, представленная ниже, поясняет формат функции COMMREQ. Вам понадобится дополнительная информация по программированию этой функции для каждого отдельного типа устройства. Требования по программированию каждого модуля, использующего функцию COMMREQ, описаны в документации на модули.

Функция COMMREQ имеет три входных параметра и один выходной параметр. Когда функция принимает поток энергии, интеллектуальному модулю посылается командный блок данных. Командный блок начинается со ссылки, заданной с использованием параметра IN. Стойка и номер слота интеллектуального модуля задаются параметром SYSID.

Функция COMMREQ может послать сообщение и ждать ответа, или может послать сообщение и продолжить работу, не ожидая ответа. Если командный блок определяет, что программа не будет ждать ответа, то содержание командного блока пересылается принимающему устройству, а исполнение программы немедленно возобновляется. (Величина тайм-аута игнорируется). Этот режим носит название NOWAIT (Без ожидания).

Если командный блок определяет, что программа должна ждать ответа, то содержание программного блока пересылается принимающему устройству и ЦП ждет ответа. Максимальное время, в течение которого ПЛК будет ожидать реакции устройства, задается командным блоком. Если устройство не ответило за этот интервал времени, то исполнение программы возобновляется. Этот режим носит название WAIT (Режим ожидания).

Выход FT (Function Faulted - Сбой функции) устанавливается в состояние ON, если:

1. Заданный целевой адрес не представлен (SYSID).
2. Заданная задача не разрешена для данного устройства (TASK).
3. Длина данных равна 0.
4. Адрес указателя статуса устройства (часть командного блока) не существует. Это может быть из-за некорректного выбора типа памяти или если адрес находится вне диапазона этого типа памяти.

Командный блок

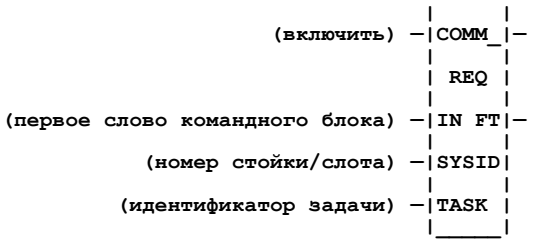
Командный блок несет информацию интеллектуальному модулю об исполняемой команде.

Адрес командного блока для функции COMMREQ задается параметром IN. Этот адрес может быть в любой области памяти типа WORD (%R, %AI, %AQ). Длина командного блока зависит от количества данных, пересылаемых устройству.

Командный блок имеет следующую структуру:

Длина (в словах)	адрес
Флаг режима Wait/No (Ожидание/Без)	адрес + 1
Память указателя статуса	адрес + 2
Смещение указателя статуса	адрес + 3
Значение тайм-аута ожидания	адрес + 4
Максимальное время связи	адрес + 5
Блок данных	адрес + 6 до адрес + 133

Информация командного блока может быть размещена в выделенной области памяти с использованием подходящей функции программирования.



Параметры

Параметр	Описание
включить	Когда функция включена, то выполняется запрос связи.
IN	IN содержит первое слово командного блока.
SYSID	SYSID содержит номер стойки (старший значащий байт) и номер слота (младший значащий байт) целевого устройства.
TASK	TASK содержит идентификатор задачи, предназначенной для выполнения на целевом устройстве.
FT	FT активизируется, если при обработке функции COMMREQ обнаруживается ошибка.

Допустимые типы памяти

Параметр	поток	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	пост.	нет
включить	•											
IN								•	•	•		
SYSID		•	•	•	•		•	•	•	•	•	
TASK								•	•	•	•	
FT	•											•

- Допустимая ссылка или место, где поток может проходить через данную функцию.

Пример

В следующем примере, когда разрешающий вход %M0020 находится в состоянии ON, командный блок, размещенный начиная с %R0016, посылается связной задаче номер 1 в устройстве, расположенном в стойке 1, в слоте 2 ПЛК. Если при обработке функции COMMREQ возникает ошибка, то устанавливается %Q0100.

Примечание

Для систем, не имеющих стоек расширения, старший байт SYSID должен быть равен 0 , указывая на главную стойку.

Глава 10

Табличные функции

Табличные функции используются для выполнения следующих функциональных задач:

Сокращение	Функция	Описание	Стр.
ARRAY_MOVE	Перемещение массива	Копирует заданное число элементов данных массива-источника в массив назначения.	10-2
SRCH_EQ	Поиск РАВНО	Ищет все значения массива, равные заданной величине.	10-6
SRCH_NE	Поиск НЕ РАВНО	Ищет все значения массива, не равные заданной величине.	10-6
SRCH_GT	Поиск БОЛЬШЕ ЧЕМ	Ищет все значения массива, большие заданной величины.	10-6
SRCH_GE	Поиск БОЛЬШЕ ИЛИ РАВНО	Ищет все значения массива, большие или равные заданной величине.	10-6
SRCH_LT	Поиск МЕНЬШЕ, ЧЕМ	Ищет все значения массива, меньшие заданной величины.	10-6
SRCH_LE	Поиск МЕНЬШЕ ИЛИ РАВНО	Ищет все значения массива, меньшие или равные заданной величине.	10-6

Максимально допустимая длина для данных функций равна 32.767 байт или слов, или 262.136 бит (биты доступны только функции ARRAY_MOVE).

Табличные функции оперируют со следующими типами данных:

Тип данных	Описание
INT	Целое со знаком.
DINT	Целое со знаком двойной разрядности.
BIT*	Тип данных - бит.
BYTE	Тип данных - байт.
WORD	Тип данных - слово.

* - только для функции ARRAY_MOVE.

Тип данных по умолчанию целое число со знаком. Тип данных можно изменить после выбора табличной функции. Для сравнения данных другого типа или для сравнения данных разных типов сначала используйте подходящую функцию преобразования данных (описанную в главе 11 «Функции преобразования») для приведения типа данных к одному из выше перечисленных.

Функция ARRAY_MOVE (INT, DINT, BIT, BYTE, WORD)

Используйте функцию ARRAY_MOVE (Переместить массив) для копирования заданного числа элементов данных из массива-источника в массив назначения.

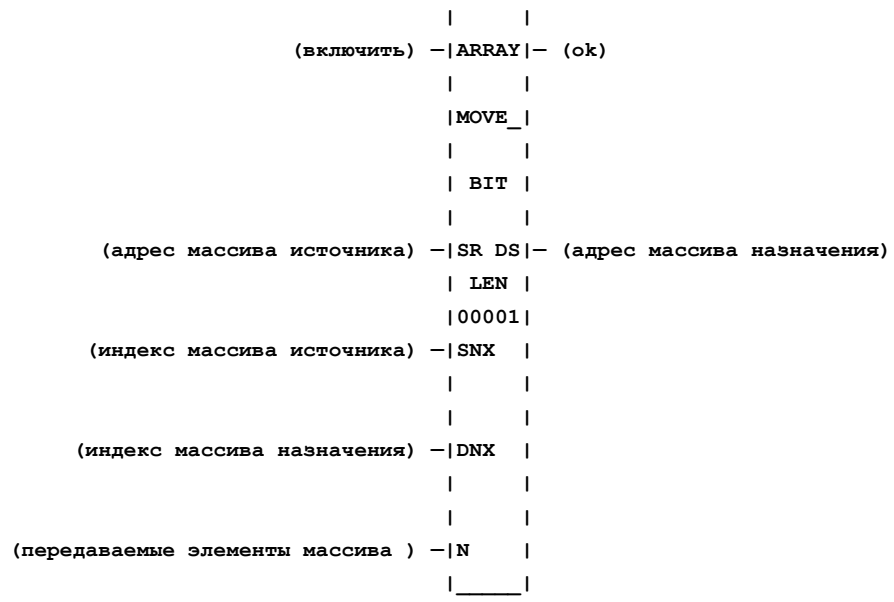
Функция ARRAY_MOVE имеет пять входных и два выходных параметра. Когда функция принимает поток энергии, то число элементов данных в индикаторе счета N выделяется из входного массива, начиная с индексированной ячейки (SR+SNX-1). Элементы данных записываются в выходной массив, начиная с индексированной ячейки (DS+DNX-1). Операнд LEN определяет число элементов, комплектуемых каждым массивом.

Для функции ARRAY_MOVE_BIT, если для параметров начальных адресов массива-источника и/или массива назначения выбирается память типа WORD, то первым битом массива является младший значащий бит заданного слова. Отображаемое значение содержит 16 бит независимо от длины массива.

Индексы команд ARRAY_MOVE являются единично-организованными (1-based). При использовании ARRAY_MOVE недопустимы ссылки на элементы вне массива источника или массива назначения (заданные начальным адресом и длиной).

Выход OK будет принимать поток энергии до тех пор, пока не произойдет одно из следующих событий:

- Параметр Enable (Включить) перейдет в состояние OFF.
- (NS+SNX-1) станет больше LEN.
- (DS+DNX-1) станет больше LEN.



Параметры

Параметр	Описание
включить	Когда функция включена, то соответствующая ей операция исполняется.
SR	SR содержит начальный адрес массива-источника. Для функции ARRAY_MOVE_BIT можно использовать любые ссылки; они могут не быть выровнены по байту. Тем не менее, оперативно отображаются 16 бит, начиная с заданного адреса ссылки.
SNX	SNX содержит индекс массива-источника.
DNX	DNX содержит индекс массива назначения.
N	N является индикатором счета.
OK	Выход OK активизируется всегда, когда активизируется включение.
DS	DS содержит начальный адрес массива назначения. Для функции ARRAY_MOVE_BIT можно использовать любые ссылки; они могут не быть выровнены по байту. Тем не менее, оперативно отображаются 16 бит, начиная с заданного адреса ссылки.
LEN	Параметр LEN задает число элементов, начиная с SR и DS, комплектующих каждый массив.

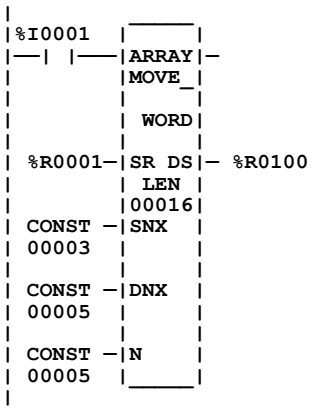
Допустимые типы памяти

Параметр	поток	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	пост.	нет
включить	•											
SR		o	o	o	o	Δ↑	o	•	•	•		
SNX		•	•	•	•		•	•	•	•	•	
DNX		•	•	•	•		•	•	•	•	•	
N		•	•	•	•		•	•	•	•	•	
OK	•											•
DS		o	o	o	o	↑	o	•	•	•		

- • Допустимая ссылка или место, где поток может проходить через данную функцию. Для функции ARRAY_MOVE_BIT, дискретные ссылки пользователя %I, %Q, %M, и %T могут не быть выровнены по байту.
- o Допустимая ссылка только для данных типа INT, BIT, BYTE или WORD; не допустима для данных типа DINT.
- Δ Допустимая ссылка только для данных типа BIT, BYTE или WORD; не допустима для данных типа INT или DINT.
- ↑ Только для %SA, %SB; %S использовать нельзя.

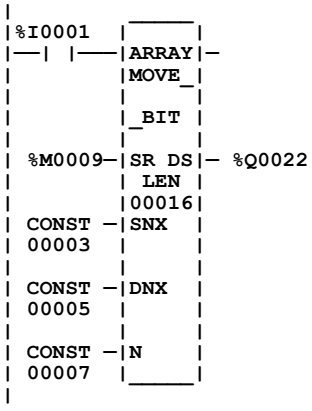
Пример 1

В этом примере %R0003—%R0007 массива %R0001—%R0016 считываются и затем пишутся в %R0104—%R0109 массива %R0100—%R0115.



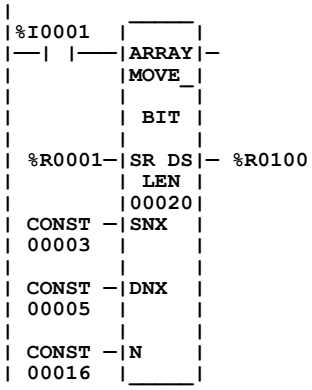
Пример 2

В этом примере, используя память битов для SR и DS, %M0011-%M0017 массива %M009—%M0024 считываются и затем пишутся в %Q0026—%Q0032% массива %Q0022—%Q0037.



Пример 3

В этом примере, используя память слов для SR и DS, третий младший значащий бит %R0001 через второй младший значащий бит %R0002 массива, содержащего все 16 бит %R0001, и четыре бита %R0002 считываются и затем пишутся в пятый младший значащий бит %R0100 через четвертый младший значащий бит %R0101 массива, содержащего все 16 битов %R0100 и четыре бита %R0101.



Функции поиска

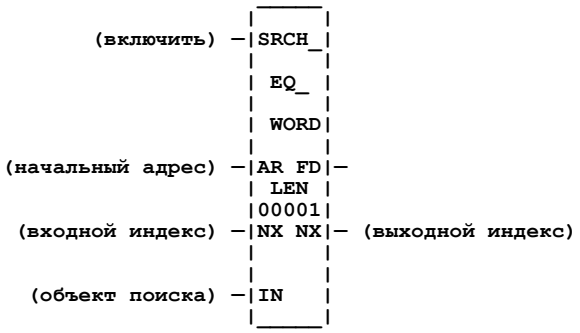
Используйте подходящую функцию поиска из числа перечисленных ниже для выполнения операций поиска:

Сокращение	Функция	Описание
SRCH_EQ	Поиск РАВНО	Ищет все значения массива, равные заданной величине.
SRCH_NE	Поиск НЕ РАВНО	Ищет все значения массива, не равные заданной величине.
SRCH_GT	Поиск БОЛЬШЕ ЧЕМ	Ищет все значения массива, большие заданной величины.
SRCH_GE	Поиск БОЛЬШЕ ИЛИ РАВНО	Ищет все значения массива, большие или равные заданной величине.
SRCH_LT	Поиск МЕНЬШЕ ЧЕМ	Ищет все значения массива, меньшие заданной величины.
SRCH_LE	Поиск МЕНЬШЕ ИЛИ РАВНО	Ищет все значения массива, меньшие или равные заданной величине.

Каждая функция имеет четыре входных параметра и два выходных параметра. Когда функция принимает энергетический поток, то выполняется поиск в массиве, начиная с (AR + входной NX). Это стартовый адрес массива (AR) плюс индекс для этого массива (входной NX).

Поиск продолжается до тех пор, пока не будет обнаружен элемент массива соответствующий объекту поиска (IN) или по достижении конца массива. Если элемент массива обнаружен, то выходной параметр (FD) устанавливается в состояние ON, и выходной параметр (выход NX) устанавливается в соответствующую позицию этого элемента в массиве. Если элемент массива не обнаружен ранее достижения конца массива, то выходной параметр (FD) устанавливается в состояние OFF и выходной параметр (выходной NX) устанавливается в 0.

Допустимым диапазоном значений для входного NX является диапазон от 0 до (LEN-1). NX должен быть установлен на 0 для начала поиска на первом элементе. Это значение прирастает на 1 во время исполнения. Таким образом, значение выходного NX равно от 1 до LEN. Если значение входного NX вне допустимого диапазона (<0 или >LEN), то его значение устанавливается в нуль по умолчанию.



Параметры

Параметр	Описание
включить	Когда функция включена, то соответствующая ей операция исполняется.
AR	AR содержит начальный адрес поискового массива.
Вход NX	Вход NX содержит индекс в массиве, с которого начинается поиск.
IN	IN содержит объект поиска.
Выход NX	NX содержит позицию массива, в которой производится поиск.
FD	FD указывает, что элемент массива найден и функция успешно выполнена.
LEN	Параметр LEN задает число элементов, начиная с AR, комплектующих массив поиска. Он может быть от 1 до 32767 байт или слов.

Допустимые типы памяти

Параметр	поток	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	пост.	нет
включить	•											
AR		o	o	o	o	Δ	o	•	•	•		
NX in		•	•	•	•		•	•	•	•	•	
IN		o	o	o	o	Δ	o	•	•	•	•	
NX out		•	•	•	•		•	•	•	•		
FD	•											•

• Допустимая ссылка или место, где поток может проходить через данную функцию.

o Допустимая ссылка только для данных типа INT, BIT, BYTE или WORD; не допустима для данных типа DINT.

Δ Допустимая ссылка только для данных типа BIT, BYTE или WORD; не допустима для данных типа INT или

Пример 1

Массив AR задан адресами памяти %R0001—%R0005. Когда EN в состоянии ON, часть массива между %R0004 и %R0005 просматривается для поиска элемента, величина которого равна IN. Если %R0001=7, %R0002=9, %R0003=6, %R0005=7, %R0100=7, то поиск начнется в %R0004 и закончится в %R0004, когда FD установлен в состояние ON и в %R0101 записано число 4.

Пример 2

Массив AR задан адресами памяти %AI0001—%AI0016. Значения элементов массива равны 100, 20, 0, 5, 90, 200, 0, 79, 102, 80, 24, 34, 987, 8, 0 и 500. Сначала %AQ0001 равно 5. Когда EN включено (ON), каждый цикл будет просматривать массив, определяя соответствие входному значению IN, равному 0. Первый цикл начинает поиск с %AI0006 и находит соответствие в %AI0007, так что FD включено (ON) и %AQ0001 равно 7. Второй цикл начнет поиск с %AI0008 и найдет соответствие в %AI0015, так что FD останется в состоянии ON и %AQ0001 будет равно 15. Следующий цикл начнется с %AI0016. Так как конец массива будет достигнут без обнаружения соответствия, то FD установится в OFF и %AQ0001 установится на 0. Следующий цикл начнет поиск с начала массива.

Глава
11

Функции преобразования

Используйте функции преобразования для преобразования элементов данных одного типа к другому типу. Многие команды программирования, например математические функции, должны использовать данные одного типа. В настоящем разделе описаны следующие функции преобразования:

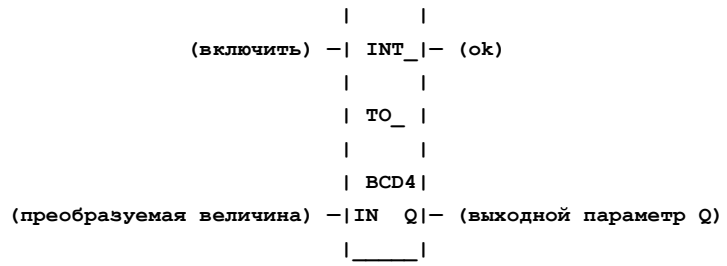
Сокращение	Функция	Описание	Стр.
BCD4	Преобразование в BCD4	Преобразует целое число со знаком в 4-значный формат BCD.	11-2
INT	Преобразование в целое со знаком	Преобразует BCD4 или REAL к целому числу со знаком.	11-4
DINT	Преобразование в целое со знаком двойной точности	Преобразует REAL в формат целого числа со знаком двойной точности.	11-6
REAL	Преобразование в действительное значение	Преобразует INT, DINT, BCD4 или WORD в REAL.	11-8
WORD	Преобразование в слово	Преобразует REAL в WORD.	11-10
TRUN	Усечение	Отсекает дробную часть действительного числа.	11-12

Функция преобразования в BCD-4 (INT)

Функция преобразования к BCD-4 используется для получения 4-значного BCD эквивалента целочисленных данных со знаком. Входные данные этой функцией не изменяются. Выходные данные можно непосредственно использовать в качестве входа другой программной функции.

Данные могут конвертироваться в BCD формат для управления BCD-кодированными светодиодными дисплеями или для представления внешним устройствам, таким как высокоскоростные счетчики.

Когда функция принимает поток энергии, она выполняет преобразование, при этом результат доступен на выходе Q. Функция пропускает поток энергии, когда энергия принимается до тех пор, пока заданное преобразование не приводит к значению, выходящему за пределы диапазона 0 – 9999.



Параметры

Параметр	Описание
включить	Когда функция включена, то соответствующая ей операция исполняется.
IN	IN содержит ссылку целочисленного значения, преобразуемого в BCD.
OK	Выход OK активизируется, когда функция выполняется без ошибок.
Q	Параметр Q содержит BCD-формат исходного значения параметра IN.

Допустимые типы памяти

Параметр	поток	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	пост.	нет
включить	•											
IN		•	•	•	•		•	•	•	•	•	
OK	•											•
Q		•	•	•	•		•	•	•	•		

- Допустимая ссылка или место, где поток может проходить через данную функцию.

Пример

В этом примере, при активизации входа %I0002 и отсутствии ошибок, целое число из входных ячеек с %I0017 по %I0032 преобразуется в четыре BCD цифры, и результат запоминается ячейках памяти с %Q0033 по %Q0048. Обмотка %Q1432 используется для проверки успешности преобразования.

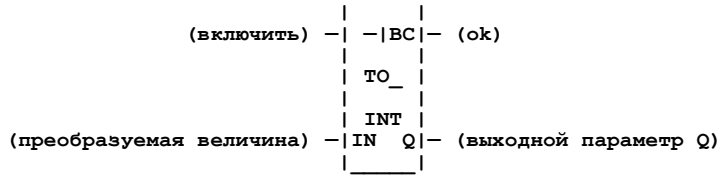
Функция преобразования в INT (BCD-4, REAL)

Функция преобразования к целочисленной величине со знаком используется для получения целочисленного эквивалента форматов BCD-4 и REAL. Входные данные этой функцией не изменяются. Выходные данные можно непосредственно использовать в качестве входа другой программной функции.

Примечание

Тип данных REAL доступен только в ЦП моделей 35x и 36x, версии 9 или старшей или во всех версиях ЦП модели 352.

Когда функция принимает поток энергии, она выполняет преобразование, при этом результат доступен на выходе Q. Функция пропускает поток энергии, когда энергия принимается до тех пор, пока заданное преобразование не приводит к значению, выходящему за пределы допустимого диапазона.



Параметры

Параметр	Описание
включить	Когда функция включена, то соответствующая ей операция исполняется.
IN	IN содержит ссылку на BCD-4, REAL или постоянное значение, преобразуемого к целочисленному.
OK	Выход OK активизируется, когда функция выполняется без ошибок.
Q	Параметр Q содержит целочисленный формат исходного значения параметра IN.

Допустимые типы памяти

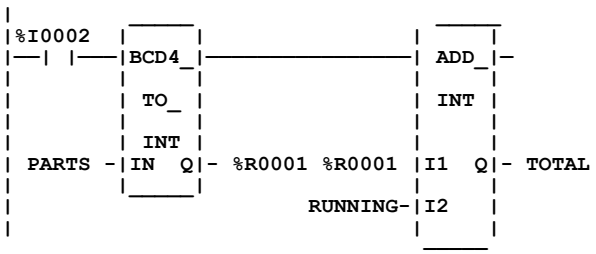
Параметр	поток	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	пост.	нет
включить	•											
IN		•	•	•	•		•	•	•	•	•	
OK	•											•
Q		•	•	•	•		•	•	•	•		

Замечание: Для данных типа REAL допустимыми типами являются только %R, %AI, %AQ.

- Допустимая ссылка или место, где поток может проходить через данную функцию.

Пример

В этом примере, при активизации входа %I0002 значение BCD-4 в PARTS преобразуется в целочисленную величину со знаком и направляется к ADD функции, где суммируется с целочисленным значением, представленным ссылкой RUNNING. Сумма появляется на выходе TOTAL.



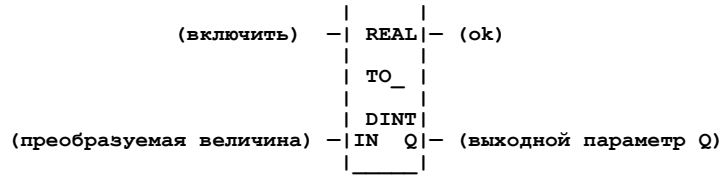
Функция преобразования в DINT (REAL)

Функция преобразования к целочисленной величине двойной точности со знаком используется для получения целочисленного эквивалента формата REAL. Входные данные этой функцией не изменяются. Выходные данные можно непосредственно использовать в качестве входа другой программной функции.

Примечание

Тип данных REAL доступен только в ЦП моделей 35x и 36x, версии 9 или старшей или во всех версиях ЦП модели 352.

Когда функция принимает поток энергии, то она выполняет преобразование, при этом результат доступен на выходе Q. Функция пропускает поток энергии, когда энергия принимается до тех пор, пока заданное преобразование не приводит к значению, выходящему за пределы допустимого диапазона.



Параметры

Параметр	Описание
включить	Когда функция включена, то соответствующая ей операция исполняется.
IN	IN содержит ссылку на значение, преобразуемое к целочисленному двойной точности.
OK	Выход OK активизируется, когда функция выполняется без ошибок.
Q	Параметр Q содержит целочисленный формат двойной точности исходного значения параметра IN.

Примечание

При конвертировании данных из REAL в DINT возможна некоторая потеря точности, так как данные типа REAL имеют 24 значащих бита.

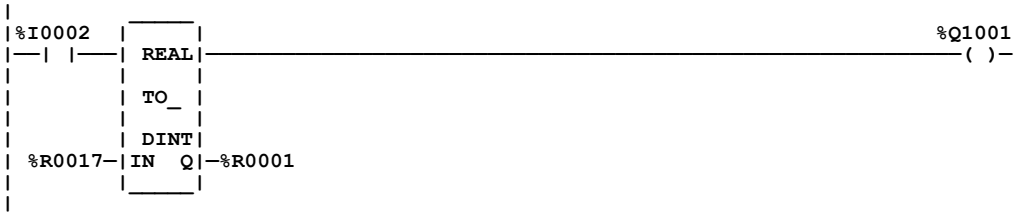
Допустимые типы памяти

Параметр	поток	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	пост.	нет
ВКЛЮЧИТЬ	•											
IN		o	o	o	o		o	•	•	•	•	
OK	•											•
Q							•	•	•	•		

- Допустимая ссылка или место, где поток может проходить через данную функцию.

Пример

В этом примере, при активизации входа %I0002 целочисленное значение из входной ячейки %I0017 преобразуется в целочисленную величину двойной точности со знаком, и результат помещается в ячейку %R0001. Выход %Q1001 активизируется, когда функция исполняется успешно.



Функция преобразования в REAL (INT, DINT, BCD-4, WORD)

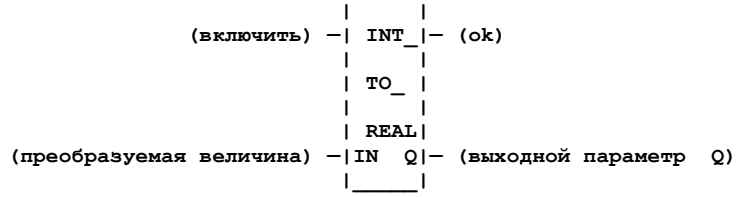
Функция преобразования к действительной величине используется для получения действительного эквивалента входных данных. Входные данные этой функцией не изменяются. Выходные данные можно непосредственно использовать в качестве входа другой программной функции.

Когда функция принимает поток энергии, то она выполняет преобразование; при этом результат доступен на выходе Q. Функция пропускает поток энергии, когда энергия принимается до тех пор, пока заданное преобразование не приводит к значению, выходящему за пределы допустимого диапазона.

Возможна некоторая потеря точности при преобразовании данных типа DINT в REAL, так как число значащих бит уменьшается до 24.

Примечание

Тип данных REAL доступен только в ЦП моделей 35x и 36x, версии 9 или старшей или во всех версиях ЦП модели 352.



Параметры

Параметр	Описание
включить	Когда функция включена, то соответствующая ей операция исполняется.
IN	IN содержит ссылку на целочисленное значение, преобразуемое к действительному.
OK	Выход OK активизируется, когда функция выполняется без ошибок.
Q	Параметр Q содержит действительный формат исходного значения параметра IN.

Допустимые типы памяти

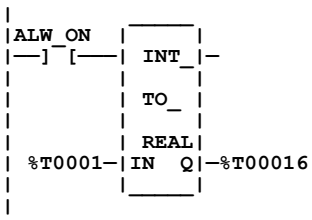
Параметр	поток	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	пост.	нет
ВКЛЮЧИТЬ	•											
IN		o	o	o	o		o	•	•	•	•	
OK	•											•
Q								•	•	•		

• Допустимая ссылка или место, где поток может проходить через данную функцию.

o Не допускается для функции DINT_TO_REAL.

Пример

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



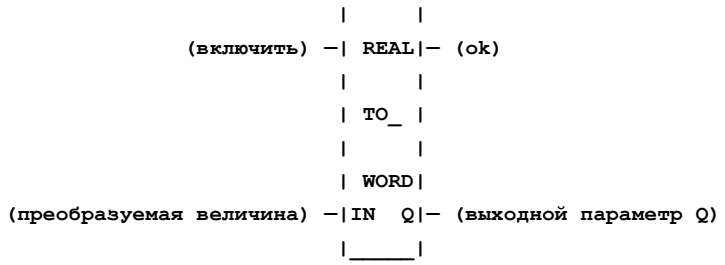
Функция преобразования в WORD (REAL)

Функция преобразования к формату WORD (Слово) используется для получения WORD эквивалента действительных входных данных. Входные данные этой функцией не изменяются.

Примечание

Эта функция доступна только для ЦП модели 352..

Когда функция принимает поток энергии, то она выполняет преобразование; при этом результат доступен на выходе Q. Функция пропускает поток энергии, когда энергия принимается до тех пор, пока заданное преобразование не приводит к значению, выходящему за пределы допустимого диапазона от 0 до FFFFh.



Параметры

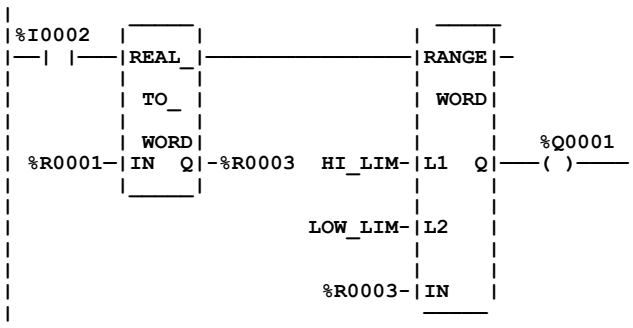
Параметр	Описание
включить	Когда функция включена, то соответствующая ей операция исполняется.
IN	IN содержит ссылку на целочисленное значение, преобразуемое к типу Слово.
OK	Выход OK активизируется, когда функция выполняется без ошибок.
Q	Параметр Q содержит целочисленное значение без знака от исходного значения параметра IN.

Допустимые типы памяти

Параметр	поток	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	пост.	нет
включить	•											
IN								•	•	•	•	
OK	•											•
Q		•	•	•	•		•	•	•	•		

• Допустимая ссылка или место, где поток может проходить через данную функцию.

Пример


%I0002
REAL_
TO_
WORD
IN
Q
%R0001
%R0003
HI_LIM
LOW_LIM
%R0003
RANGE
WORD
L1
Q
L2
IN
%Q0001

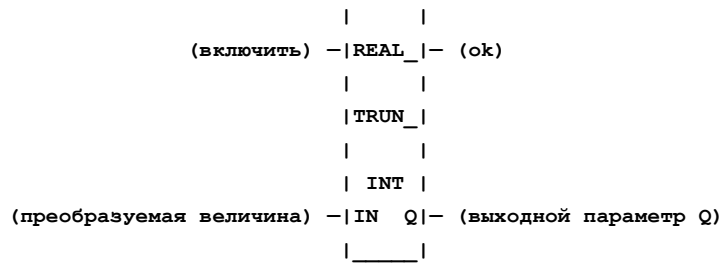
Функция TRUN (INT, DINT)

Функция усечения используется для отбрасывания дробной части действительного числа к 0. Входные данные этой функцией не изменяются. Выходные данные можно использовать в качестве входа другой программной функции.

Примечание

ЦП модели 352 является единственной моделью Series 90-30, выполняющей операции с плавающей запятой, поэтому функция усечения не применима для других ЦП.

Когда функция принимает поток энергии, то она выполняет преобразование; при этом результат доступен на выходе Q. Функция пропускает поток энергии, когда энергия принимается до тех пор, пока заданное преобразование не приводит к значению, выходящему за пределы допустимого диапазона, или пока вход не находится в состоянии NaN (Не число).



Параметры

Параметр	Описание
включить	Когда функция включена, то соответствующая ей операция исполняется.
IN	IN содержит ссылку округляемой действительной величины.
OK	Выход OK активизируется, когда функция выполняется без ошибок, пока значения не выходят за допустимый диапазон или вход не находится в состоянии NaN (Не число).
Q	Параметр Q содержит усеченное INT или DINT значение изначального значения параметра IN.

Примечание

Возможна потеря точности при преобразовании от типа REAL к типу DINT, так как тип REAL имеет 24 значащих бита.

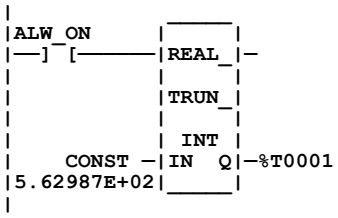
Допустимые типы памяти

Параметр	поток	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	пост.	нет
включить	•											
IN								•	•	•	•	
OK	•											•
Q		o	o	o	o		o	•	•	•		

- • Допустимая ссылка или место, где поток может проходить через данную функцию.
- o Только для функции REAL_TRUN_INT.

Пример

В следующем примере изображенная константа усекается и целочисленный результат 562 помещается в %T0001.



Глава
12

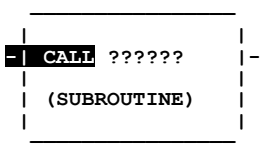
Функции управления

В настоящем разделе описаны функции управления, которые могут использоваться для ограничения исполнения программы и изменения пути исполнения программы. (Обратитесь к Главе 2, разделу 1 «Общие сведения о цикле ПЛК» для дополнительной информации о цикле ЦП)

Функция	Описание	Стр.
CALL	Вызывает переход исполнения программы к заданному блоку подпрограммы.	12-2
DOIO	Немедленно выполняет один цикл заданного диапазона входов и выходов. Все входы и выходы модуля обслуживаются, если ссылка на область памяти данного модуля включена в функцию DOIO. Частичные обновления модулей ввода/вывода не выполняются. Также копия просканированных входов/выходов может быть размещена во внутренней памяти быстрее, чем оригинал.	12-7
SER	Последовательный Регистратор Событий – производит запись выборок. Блок управления состоит из пользовательской конфигурации для блока управления, настройки значений и операционных параметров.	12-9
END	Производит временное прекращение программы. Программа выполняется, начиная с первого шага, и останавливается либо на последнем шаге, либо на команде END – смотря, что встретится раньше. Эта команда полезна для отладки, но не разрешена в SFC программировании (см. примечание на стр. 12-23).	12-23
MCR и MCRN	Функция MCR приводит к исполнению всех шагов между MCR и последующей функцией ENDMCR без инициализации. Программное обеспечение Logicmaster 90-30/20/Micro поддерживает две формы функции MCR - не вложенную форму (MCR) и вложенную форму (MCRN).	12-24
ENDMCR и ENDMCRN	Указывает, что следующая часть программы будет исполняться с инициализацией. Программное обеспечение Logicmaster 90-30/20/Micro поддерживает две формы функции ENDMCR: не вложенную форму (ENDMCR) и вложенную форму (ENDMCRN).	12-27
JUMP и JUMPN	Происходит переход выполнения программы к заданной в функции LABEL ячейке (см. ниже). Программное обеспечение Logicmaster 90-30/20/Micro поддерживает две формы функции JUMP, не вложенную форму (JUMP) и вложенную форму (JUMPN).	12-28
LABEL и LABELN	Задает целевую ячейку для функции JUMP. Программное обеспечение Logicmaster 90-30/20/Micro поддерживает две формы функции LABEL - не вложенную форму (LABEL) и вложенную форму (LABELN).	12-30
COMMENT	Помещает комментарий (пояснение к шагу) в программу. После программирования команды, текст может быть впечатан в команду путем “Раскрытия данной команды”.	12-31
SVCREQ	Производит запрос специальных услуг ПЛК. (см. список на странице 12-32)	12-32
PID	Предоставляет два алгоритма PID (пропорционального / интегрального / дифференциального управления замкнутым контуром): • Стандартный ISA алгоритм PID (PIDISA) • Алгоритм с независимыми составляющими (PIDIND).	12-74

Функция CALL

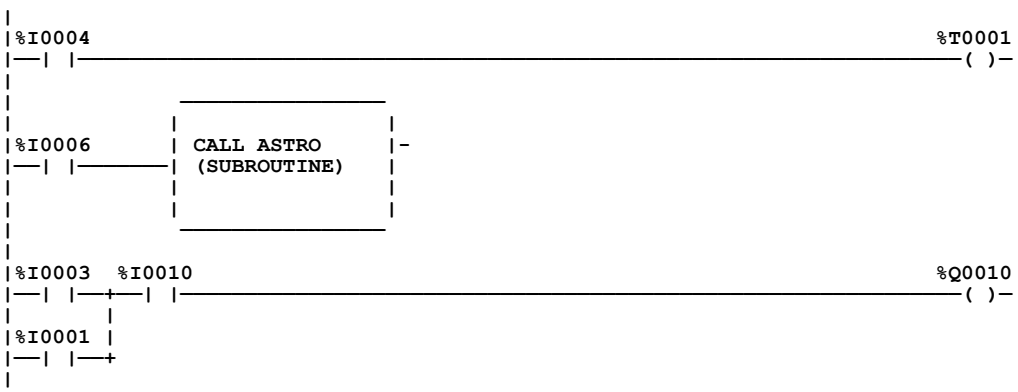
Используйте функцию CALL для перехода исполнения программы к заданному блоку подпрограммы.



Когда функция CALL инициализируется, то она вызывает немедленный переход к обозначенному блоку подпрограммы и исполняет его. По завершении исполнения подпрограммы управление возвращается в точку программы, расположенную сразу после команды CALL.

Пример:

Следующий пример демонстрирует подпрограмму команды CALL, появляющуюся в блоке вызова. Располагая курсор на команде, вы можете нажать клавишу F10, чтобы раскрыть подпрограмму.



Примечание

Micro ПЛК не приспособлены для работы с подпрограммами, поэтому функция CALL не используется совместно с ПЛК Series 90 Micro.

Функция DOIO

Функция DO I/O (DOIO) используется для обновления входов или выходов в каком либо одном цикле ЦП при работе программы. Функция DOIO может также использоваться для обновления выбранных В/В в процессе исполнения программы, в дополнение к обычному сканированию входов/выходов.

Если заданы входные ссылки, то эта функция позволяет сделать доступными программе самые свежие значения входных величин. Если заданы выходные ссылки, то функция DO I/O обновляет выходы, основываясь на последних текущих значениях, сохраненных в памяти В/В. В/В обрабатывается в соответствие с конфигурацией модулей В/В; при необходимости ПЛК настраивает ссылки при исполнении функции.

Функция DOIO имеет четыре входных параметра и один выходной. Когда функция инициализируется и заданы входные ссылки, она просматривает входные каналы с начальной ссылки ST и до конечной END. Если ссылка задана для параметра ALT, то копия новых входных значений размещается в памяти, начиная с данной ссылки, и реальные входные каналы не обновляются. Параметр ALT должен быть того же размера, что и сканируемый тип ссылки. Если для ST и END используется дискретная ссылка, то ALT должен быть тоже дискретным. Если для ALT ссылок не задано, то реальные входные каналы обновляются.

Когда функция DOIO инициализируется и выходные ссылки заданы, выходные каналы с начальной ссылки ST и конечной END записываются в выходные модули. Если выходы должны быть переписаны в выходные модули из внутренней памяти, то для ALT могут быть заданы начальные ссылки, отличные от %Q или %AQ. Диапазон выходов, записываемых в выходные модули, задается начальной ссылкой ST и конечной ссылкой END.

Исполнение функции продолжается до тех пор, пока либо не будут просмотрены все входы избранного диапазона, либо пока не будут обслужены все выходы плат вводов/выводов. После этого исполнение программы возвращается к функции, следующей за DO I/O.

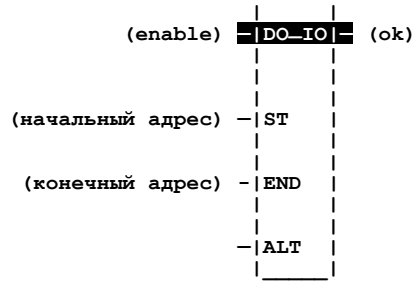
Если диапазон ссылок содержит дополнительные модули (HSC, APM), то все входные данные (%I и %AI) или все выходные данные (%Q и %AQ) этого модуля будут просканированы. При сканировании дополнительных модулей параметр ALT игнорируется. Кроме того, диапазон ссылок не должен включать модули Enhanced GCM.

Примечание

Для ЦП версии 9 или старше, функция DOIO **может** использоваться с модулем Enhanced GCM.

Выход функции активен всегда, кроме следующих исключений:

- Не все ссылки заданного типа присутствуют в выбранном диапазоне.
- ЦП не в состоянии правильно обслужить временный список вводов/выводов, созданный функцией.
- Заданный диапазон включает модули ввода/вывода, связанные со сбоем типа “Потеря модуля ввода/вывода”.



Параметры

Параметр	Описание
включить	Когда функция включена, то выполняется сканирование ограниченного числа входов или выходов.
ST	ST является начальным адресом или набором обслуживаемых входных или выходных каналов или слов.
END	END является конечным адресом или набором обслуживаемых входных или выходных каналов или слов.
ALT	При сканировании входов, ALT задает адрес для запоминания значений, просканированных входных каналов/слов. При сканировании выходов, ALT задает адрес получения значений выходных каналов /слов для пересылки модулям ввода/вывода. Для моделей ЦП 331 и старше, параметр ALT может влиять на скорость работы функционального блока DOIO (см. приведенное ниже примечание и раздел на странице 12-7, посвященный усовершенствованной функции DO I/O для ЦП моделей 331 и старше.)
OK	Выход OK активизируется, когда сканирование входов или выходов завершается успешно.

Примечание

Для моделей центрального процессора 331 и старше, параметр ALT функционального блока DOIO можно использовать для ввода номера слота модуля в главную стойку. Если это происходит, то блок DOIO выполняется за 80 мкс, вместо 236 мкс, необходимых при программировании блока без параметра ALT. Проверка на ошибки не выполняется во избежание наложения адресов ссылок или несоответствия типов модулей.

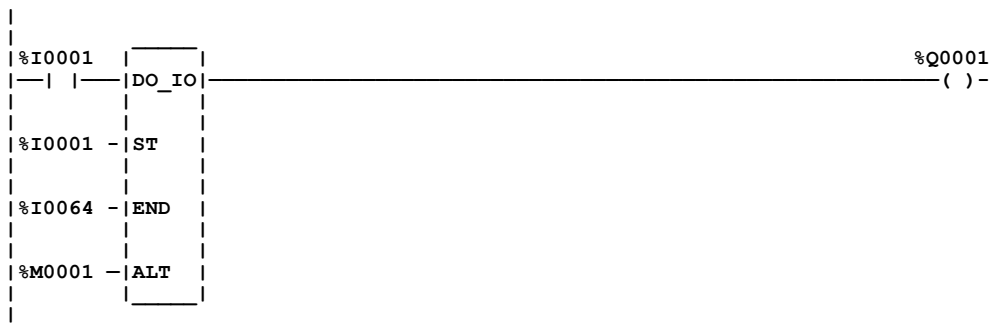
Допустимые типы памяти

Параметр	со-стоя ние	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	const	none
enable	•											
ST		•	•						•	•		
END		•	•						•	•		
ALT		•	•	•	•		•	•	•	•		•
ok	•											•

• Активное состояние входа (выхода).

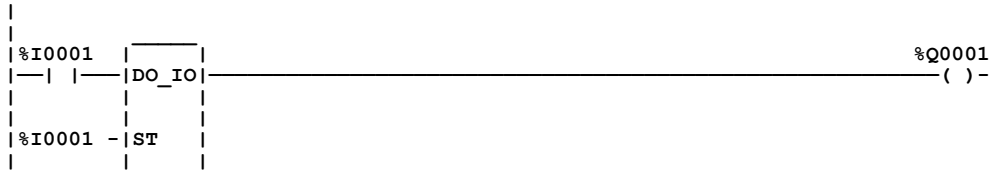
Пример 1 для входа:

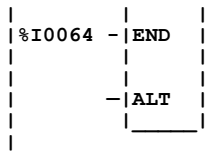
В следующем примере, когда контакт %I0001 устанавливается в 1, производится сканирование ссылок с %I0001 по %I0064 и обмотка %Q0001 включается. Копия просканированных входов помещается во внутренней памяти в ссылках от %M0001 до %M0064. Реальные каналы не обновляются. Эта форма функции может использоваться для сравнения текущих значений входных каналов со значениями входных каналов в начале сканирования.



Пример 2 для входа:

В следующем примере, когда контакт %I0001 устанавливается в 1, производится сканирование ссылок с %I0001 по %I0064 и обмотка %Q0001 включается. Просканированные входы помещаются в память статуса входов в ссылках от %I0001 до %I0064. Эта форма функции позволяет просканировать входные каналы один или несколько раз на протяжении исполнения программы в цикле ЦП.





Пример 1 для выхода:

В следующем примере, когда контакт %I0001 устанавливается в 1, значения ссылок с %R0001 по %R0004 записываются в выходные аналоговые каналы с %AQ001 по %AQ004 и обмотка %Q0001 включается. Значения с %AQ001 по %AQ004 не переписываются в модули аналоговых выходов.

Пример 2 для выхода:

В следующем примере, когда контакт %I0001 устанавливается в 1, значения ссылок с %AQ001 по %AQ004 записываются в каналы аналоговых выходов с %AQ001% по AQ004 и %Q0001 включается.

Улучшенная DO I/O функция для ЦП модели 331 и старше

Предупреждение

Если улучшенная DOIO функция используется в программе, то эта программа НЕ должна загружаться версией Logicmaster 90-30/20 более ранней, чем 4.01.

Улучшенная версия функции DOIO доступна для программного обеспечения версии 4.20 и старше. Эта версия функции DOIO может использоваться только для одиночного дискретного входа или дискретного выхода 8-канального, 16- канального или 32- канального модуля.

Параметр ALT идентифицирует слот главной стойки, в котором размещен модуль. Например, постоянное значение 2 этого параметра указывает ЦП, что исполняется модернизированная версия функционального блока DOIO для модуля, расположенного в слоте 2.

Примечание

Единственной выполняемой функциональным модулем DOIO проверкой является проверка состояния модуля в заданном слоте, чтобы убедиться, что он в порядке.

Модернизированная DOIO функция применима только к модулям, размещенным в главной стойке. Поэтому параметр ALT должен быть между 2 и 5 для 5-слотовой стойки или от 2 до 10 для 10-слотовой стойки.

Начальная и конечная ссылки должны быть либо %I, либо %Q. Эти ссылки задают первую и последнюю ссылки конфигурированного модуля. Например, если 16- канальный модуль конфигурирован со ссылками с %I0001 по %I0016 в слоте 10 для 10-слотовой главной стойки, то параметр ST должен быть %I0001, параметр END должен быть %I0016 и параметр ALT должен быть 10, как показано ниже:

В следующей таблице сравниваются времена исполнения обычного функционального блока DOIO для 8- канального, 16- канального и 32- канального модуля с дискретными входами/выходами с аналогичными показателями для модернизированного функционального блока DOIO:

Модуль	Время исполнения обычной функции	Время исполнения модернизированной функции
8- канальный модуль дискретного ввода 8- канальный модуль дискретного вывода	224 мкс 208 мкс	67 мкс 48 мкс
16- канальный модуль дискретного ввода 16- канальный модуль дискретного вывода	224 мкс 211 мкс	68 мкс 47 мкс
32- канальный модуль дискретного ввода 32- канальный модуль дискретного вывода	247 мкс 226 мкс	91 мкс 50 мкс

Функция SER

Возможности

- Функция SER (Последовательный Регистратор Событий) осуществляет протоколирование событий. Когда на вход Enable поступает 1, происходит запись до 32 битов данных для каждой записи.
- Каждый блок SER может содержать до 1024 записей.
- Если блок SER встроен в периодическую подпрограмму, скорость протоколирования определяется скоростью выполнения периодической подпрограммы.
- Метка времени присваивается только тем записям, которые были получены в режиме триггера. Эта метка может быть представлена в формате BCD (максимальное разрешение 1 сек.) или в формате POSIX (максимальное разрешение 10 мс.) Метка времени помещается в назначенную ей область памяти. Блок SER поддерживает не более одной метки времени на каждый процесс записи.
- Как триггер, блок SER может быть настроен на различные режимы работы: pre- trigger, mid- trigger и post- trigger (более подробно на странице 12-16).

Настройка блока SER осуществляется с помощью блока управления, который Вы можете создать, используя команду BLKMV (смотрите страницу 12-12).

Примечание

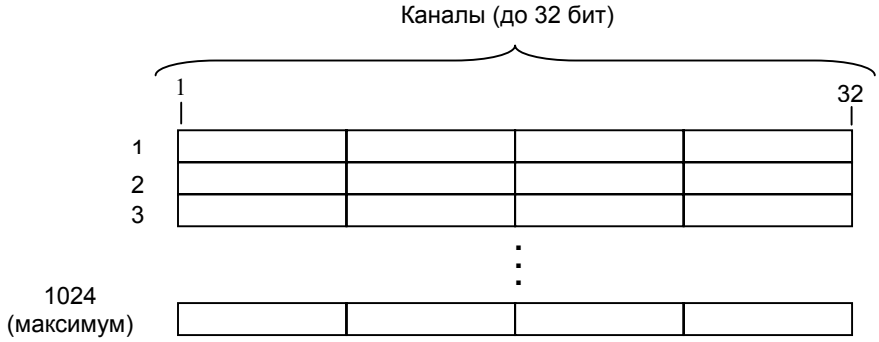
Синхронизация ПЛК – ПЛК не поддерживается.

Функциональный блок SER имеет один выход и три входа: enable, reset, и trigger.

Пример функционального блока SER

Примечание

Функция SER требует наличия микропрограммы ЦП Версии 9.0 или старше и доступна ***только*** в ЦП модели 350 и старше.


Каналы (до 32 бит)
1
32
1
2
3
1024
(максимум)

Параметры

Параметр	Описание
enable	Всякий раз, когда функция SER запускается, и вход reset установлен в 0, происходит запись событий со всех каналов.
R	Когда на вход reset поступает 1, функция SER обнуляется независимо от состояния входа Enable. Параметры Sample Buffer, Trigger Sample Offset, Trigger Time и Current Sample Offset также обнуляются. Блок SER остается в этом положении до тех пор, пока на вход reset поступает 1. Пока функция находится в состоянии сброса, выход OK установлен в 0. При подаче 0 на вход reset, протоколирование возобновляется.
T	Если выбран режим Trigger Input и функциональный блок запускается при поступлении сигнала на вход T, то блок SER переходит в режим триггера. Устанавливаются параметры Trigger Time, Trigger Sample Offset, и начинается протоколирование. Протоколирование в режиме триггера осуществляется независимо от ранее записанных событий. При переключении, регистратор событий будет осуществлять протоколирование до тех пор, пока число записей не достигнет количества, заданного в параметре «Number of Samples After Trigger». Если Trigger Mode настроен на работу в качестве Full Buffer, то сигнал, поступающий на вход T игнорируется. Для получения информации по настройке режима Full Buffer, смотрите пункт “Функциональный блок управления ” на странице 12-12.
Начальный адрес	Массив данных блока управления, состоящий из 78 слов, начинается с этого адреса. Блок управления определяет функционирование блока SER. Смотрите пункт “Функциональный блок управления ” на странице 12-12 для получения дополнительной информации.
ok	Выход ok устанавливается в 1, когда выполнено условие переключения (они определяются параметром Trigger Mode) и все записи получены. Если вход Reset установлен в 1, то выход Ok также устанавливается в 1 независимо от состояния входа Enable.

Применимые типы памяти

Параметр	состояние	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	постоянная	нет
enable	•											
Блок управления								•				
R	•											
T	•											
ok	•											•

• Активное состояние входа (выхода).

Функциональный блок управления

Функциональный блок управления – это массив из 78 слов, который содержит информацию о данных, предназначенных для протоколирования и о механизме протоколирования. В некоторых программах, одному блоку SER может соответствовать один блок команд и данных.

Для настройки параметров блока SER, выполните следующие шаги:

2. Настройте параметры массива, как описано в нижеприведенной таблице. Вы можете использовать команду BLKMV для инициализации регистров или инициализации данных, расположенных в этих регистрах и сохранении таблиц регистров до активизации функции SER.
3. Добавьте функцию SER к вашей программе.

Примечание

Если Вам необходимо использовать X каналов, где X не равно 8, 16, 24, но меньше 32, Вы должны выбрать такое число каналов, которое было бы больше, чем X и кратно 8, и установить для него нулевое описание канала. Данное описание имеет переключатель в ячейке 0xFFh, а параметр length равен числу неиспользуемых каналов со смещением равным 0.

Номер слова	Параметр	Описание
0 (начальный адрес)	Status	Переменная с атрибутом «только для чтения», которая указывает на текущее состояние блока SER. Дополнительная информация предоставляется параметром «Status Extra Data» (первое слово). **Примечание:** Если в блоке управления обнаружена ошибка, параметр Status устанавливается в значение 6, выход OK сбрасывается и никаких действий не происходит. Параметр «Status» может принимать следующие значения: 0 = Сброс 1 = Неактивно 2 = Активно 3 = Переключен 4 = Завершен 5 = Перегрузка 6 = Ошибка в параметре
1	Status Extra Data	Переменная с атрибутом «только для чтения», которая содержит дополнительную информацию о работе блока SER. Для настройки данного параметра обратитесь к разделу «Структура параметра Status Extra Data» на странице 12-14.
2	Trigger Mode	Данный параметр определяет условия, согласно которым блок SER переходит в состояние триггера. Доступные следующие настройки: 0 = Режим Trigger input 1 = Режим Full Buffer В режиме Trigger input, если функциональный блок запущен, генерируется метка времени в тот момент, когда на вход T поступает 1. Протоколирование продолжается до тех пор, пока число записей не достигнет количества, заданного в параметре «Number of Samples After Trigger». Когда это произойдет, выход OK установится в 1. В режиме Full Buffer сигнал, поступающий на вход T, игнорируется. При запущенном блоке, протоколирование продолжается до тех пор, пока буфер данных не будет заполнен. Когда это произойдет, на выход OK поступит 1. Размер буфера определяется параметром «Number of Samples».

Номер слова	Параметр	Описание
3	Trigger Time Format	Определяет, в каком формате будет представлен параметр Trigger Time. Для формата BCD установите этот параметр в 0. Для формата POSIX – в 1. (более подробно об этом рассказано на странице 12-22.)
4—7	Reserved	Слова с 4 по 7 являются зарезервированными и должны быть установлены в 0.
8	Number of Channels (bits per sample)	Определяет число битов, которые будут записаны и занесены в буфер при каждом цикле срабатывания блока SER. Варианты выбора: 8, 16, 24 или 32 бита. Шаг имеет размер 1 байт (8 бит) и, для неиспользуемых каналов необходимо настроить нулевое описание (см. слова с 14 по 77).
9	Number of Samples	Устанавливает размер буфера данных. Допустимый размер от 1 до 1024 (действительный размер буфера в битах равен числу записей, помноженному на число каналов.)
10	Number of Samples After Trigger	Устанавливает число записей, которые будут записаны после выполнения Триггерного условия. Этот параметр может принимать значения от 0 до N -1, где N – число записей. Он может быть использован только при включенном режиме Trigger input. (0).
11	Input Module Slot	Определяет месторасположение модуля ввода, с которого будет осуществляется запись (слот в главной стойке). Если значение равно 0, сканирование модуля не производится. Когда модуль сканируется, получаемые значения запоминаются во внутренней памяти, а начальный адрес не изменяется. Для запоминания значений, полученных при сканировании модуля ввода, в буфере данных необходимо назначить описания каналов. Если модуля нет в слоте или он оказался неисправен во время сканирования, то полученная информация будет нулевой. Если это произошло, сбой не будет занесен в таблицу сбоев. Сбой будет зафиксирован с сканере B/B.
12	Data Block Segment Selector	Определяет тип памяти для блока данных. Например, если Вы хотите использовать память %R и начать с ячейки %R0100, то Вам необходимо ввести число 08. Доступные значения для этого параметра следующие: %R (08h), %AI (0Ah), %AQ (0Ch). Более подробно блок данных описан на странице 12-15.
13	Data Block Offset	Определяет начальный адрес для блока данных. Отсчет для данного параметра начинается с нуля. Например, если Вы хотите иметь начало в ячейке %R0100 Вы должны ввести число 99 для этого параметра. Убедитесь, что Вы выделили достаточно памяти под блок данных.
14—77	Channel Descriptions	Определяет начальный адрес (Переключатель, длину и смещение) для конкретных каналов. Количество описаний (от 1 до 32) зависит от числа каналов, с которых будет производиться запись и объема данных. Возврат данных происходит по адресу, определенному здесь же.
	Channel Segment Selector/Length	Данное слово, вводимое как 16-тиричное число, определяет переключатель и длину данных (в битах). MSB = переключатель. LSB = Размер данных. Размер данных полезно знать, если выборки последовательные. Переключатель может иметь различных тип данных: %I (46h), %Q (48h), %M (4Ch), %T (4Ah), %G (56h), %S (54h), %SA (4Eh), %SB (50h), %SC (52h), нулевой переключатель (FFh), и переключатель модуля ввода (00h).
		Параметр length может изменяться в пределах от 1—32, но сумма всех длин не должна превышать значения параметра «Number of Channels». При длине больше чем 1, соседние каналы можно настроить на использование одного описания канала.
	Channel Offset	Данное слово, вводимое как шестнадцатеричное число, устанавливает битовое смещение для типа данных или для модуля ввода определенных в параметре «Channel Segment Selector». Отсчет для данного параметра начинается с нуля. Диапазон значений данного параметра изменяется и зависит от параметра Channel Segment Selector (типа данных и длина данных). Смещение указывает на ячейку в таблице данных или в модуле ввода, с которого производится запись.

Структура параметра «Status Extra Data»

Параметр «Status Extra Data» (первое слово в блоке управления) содержит дополнительную информацию о состояниях функции SER.

Значение	Состояние	Описание
0	Reset State	Если вход Reset установлен в 1, то параметры Sample Buffer, Trigger Sample Offset, Trigger Time, и Current Sample Offset обнуляются. Выход Ok установлен 0. Когда на вход Reset поступает 0, происходит переключение в ***состояние Inactive***. Параметр «Status Extra Data» не имеет большого значения и будет обнулен.
1	Inactive	Данное состояние является промежуточным между состоянием «Reset State» и состоянием «Active». Операции в этом состоянии не выполняются. Выход Ok блока SER установлен в 0. Переключение в активное состояние происходит, когда на вход Enable поступает 1.
2	Active	В этом состоянии вход Enable установлен в 1, но функциональный блок не сбрасывается при ошибке или переключении. Когда блок запущен, происходит запись одного события на каждом цикле срабатывания блока SER. Выход Ok блока установлен в 0. Триггерное условие (определяемое параметром «Trigger Mode») контролируется и при его выполнении блок переходит в состояние триггера. Если записей было произведено больше, чем значение параметра «Numbers of Sample», то параметр «Status Extra Data» устанавливается в 0x01, в обратном случае в 0x00.
3	Triggered	Блок переходит в данное состояние, в случае выполнения условия триггера. Дополнительное протоколирование зависит от режима Trigger Mode и от настроек параметров. Выход Ok блока установлен в 0. Переход в состояние «Complete» происходит после записи всех событий. Если записей было произведено больше, чем значение параметра «Numbers of Sample», то параметр «Status Extra Data» устанавливается в 0x01, в обратном случае в 0x00.
4	Complete	Протоколирование завершено. Выход Ok установлен в 1. Возможен переход только в режим Reset. Если записей было произведено больше, чем значение параметра «Numbers of Sample», то параметр «Status Extra Data» устанавливается в 0x01, в обратном случае в 0x00.
5	Overrun Error	Блок управления или блок данных превысил допустимый объем памяти Выход Ok установлен в 1. Возможен переход только в режим Reset. Параметр «Status Extra Data» не имеет большого значения и будет обнулен.
6	Parameter Error	Ошибка в блоке управления или в других параметрах. Выход Ok установлен в 1. Возможен переход только в режим Reset. Параметр «Status Extra Data» содержит информацию о блоке управления, в котором произошел сбой.
7	Status Error	Неверное значение в параметре «Status». Выход Ok установлен в 1. Возможен переход только в режим Reset. Значение неверного статуса запоминается.

Формат блока данных

Блок данных функции SER состоит из буфера данных, начального адреса и информации о триггере. Эта информация предоставляется Центральным процессором, и Вы можете ее только прочитать. Вы сами решаете, сколько необходимо выделить памяти для буфера данных. Блок данных имеет следующий формат:

Номер слова*	Описание параметра
0	В данном слове хранится значение параметра Current sample offset. Ссылка на ячейку, куда была помещена последняя запись. Отсчет данного параметра начинается с нуля. Допустимый диапазон от -1 до 1023. Register Location of Sample = (Num Bytes per Sample) * (Offset Parameter)/2 + (Sample Buffer Starting Register). **Примечание:** Данное значение неопределенно до тех пор, пока не будет получена первая запись. Это значение устанавливается в -1, когда происходит сброс блока SER.
1	Параметр Trigger sample offset number. Ссылка на область памяти, куда будут помещены записи, полученные в результате выполнения условия триггера. Отсчет данного параметра начинается с нуля. Допустимый диапазон от 1 до 1023. Register Location of Sample = (Num Bytes per Sample) * (Offset Parameter)/2 + (Sample Buffer Starting Register). **Примечание:** Данное значение неопределенно до тех пор, пока не будет получена первая запись. Это значение устанавливается в 0, когда происходит сброс блока SER.
С 2 до 5	Параметр Trigger Time: Показывает время, согласованное с внутренними часами ПЛК, когда было выполнено условие триггера. Время может быть представлено как в формате BCD (по умолчанию), так и в формате POSIX. Формат определяется параметром Trigger Time Format в блоке управления. Это значение устанавливается в 0, когда на вход R поступает 1.
6 и до конца	Параметр Sample Buffer. Область памяти, в которой сохраняются записи. Эта область обнуляется, когда вход Reset устанавливается в 1. Размер буфера данных изменяется и зависит от числа каналов и размера записи. Буфер данных – это циклический буфер, т.е. когда произведена последняя запись, следующая запишется на место первой. End of sample buffer = 5 + ({[(# of samples to be taken) * (# of channels to be sampled / 8)] +1} / 2

*Смещение от начального адреса определяется параметрами Data Block Segment Selector (слово 12) и Data Block Offset (слово 13) в блоке управления.

Операции блока SER

Если при выполнении программы блок SER запущен, он производит протоколирование каналов и записывает полученные данные в циклический список. Когда протоколирование завершено, выход Ok устанавливается в 1. Момент установки выхода в 1 может быть использован в качестве метки времени для регистрации времени, когда произошла последняя запись (смотри пункт Режимы записи выборок).

Блок SER должен быть сброшен (вход Reset установлен в 1) перед началом протоколирования. Сбрасывание назначается в блоке данных. Если блок SER не будет сброшен, он продолжит работу с текущими значениями в блоке данных, что послужит получению неверных результатов.

Блок управления сканируется все время, пока блок SER находится в состояниях Reset, Active или Triggered. Если Вы изменяете конфигурацию параметров блока управления во время

функционирования программы, то эффект будет достигнут на следующем цикле сканирования блока управления. При возникновении сбоя, операции останавливаются, и блок переходит в состояние сбоя. Вы должны исправить ошибку и перезапустить блок.

При выборе модуль ввода для сканирования, ПЛК не проверяет, является ли этот модуль дискретным модулем ввода и назначены ли ему описания каналов. Вы должны более точно настроить процесс протоколирования для модуля ввода. Хотя для модуля ввода может быть определено несколько описаний каналов, сканируется он один раз во время выполнения блока SER.

Блок SER можно разместить либо в программе, либо в циклической подпрограмме. Если Вы помещаете его в программе, то интервал между сканированиями будет равен времени цикла ПЛК, которое зависит от количества и типов действующих функций. Если Вы помещаете его в циклической подпрограмме, то интервал между операциями сканирования может быть установлен примерно 1 мс, но при этом протоколирование будет вестись с искажениями.

При настройке цикла работы блока на уровне 1 мс тратится до 50% вычислительных ресурсов ПЛК. Нельзя в одной программе использовать два блока со временем цикла работы на уровне 1 мс.

Режимы записи

Режимы записи определяются параметрами Trigger Mode (слово 2 в блоке управления) и Number of Samples After Trigger (слово 10). Вам необходимо разобрать содержимое буфера данных для понимания того, как эти параметры настраиваются.

Протоколирование под управлением триггера

Для настройки режимов pre-, mid-, and post-trigger используйте режим Trigger Mode (слово 2 = 0). Режим записи управляется параметром Number of Samples After Trigger (слово 10). Во всех случаях, старт записи происходит после установки входа Enable в 1. Когда вход T установлен в 1, протоколирование продолжается до тех пор, пока число записей не достигнет величины заданной параметром Number of Samples After Trigger. Выход блока SER устанавливается в 1 после окончания процесса записи.

Если число записей превысило величину, заданную параметром Number of Samples (слово 9) прежде чем был достигнут параметр «Number of Samples After Trigger», то новые записи будут размещаться в буфере на месте старых.

При первом переключении в режим триггера, генерируется метка времени, которая заносится в специальную ячейку памяти.

Pre Trigger

Производит непрерывное протоколирование, пока на вход T поступает 0

Для настройки этого режима установите для слова 10 значение 0, и когда на вход T поступит 1, запись прекратиться и сгенерируется метка времени (Все записи получены перед переключением).

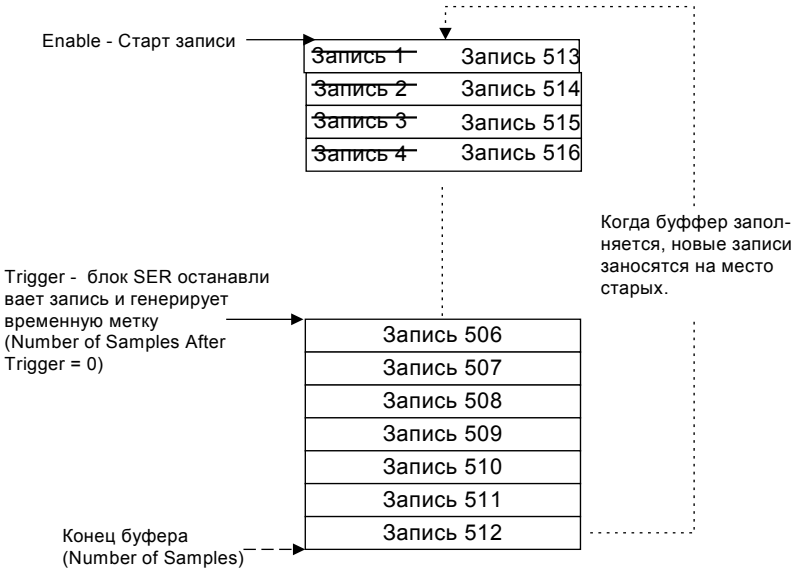


Рисунок 12-1. Сбор выборок на примере режима Pre-Trigger

Mid Trigger

Производит непрерывное протоколирование, пока число записей не достигнет величины, заданной параметром Number of Samples After Trigger.

Для настройки этого режима назначьте слову 10 значение 1 и установите параметр Number of Samples (слово 9). Когда на вход T поступит 1, протоколирование продолжится, пока не достигнет заданного числа. Например, параметр Number of Samples After Trigger установлен в 12. когда протоколирование прекратиться буфер будет содержать 500 записей, полученных в режиме Pre-Trigger и 12 в режиме post-trigger.

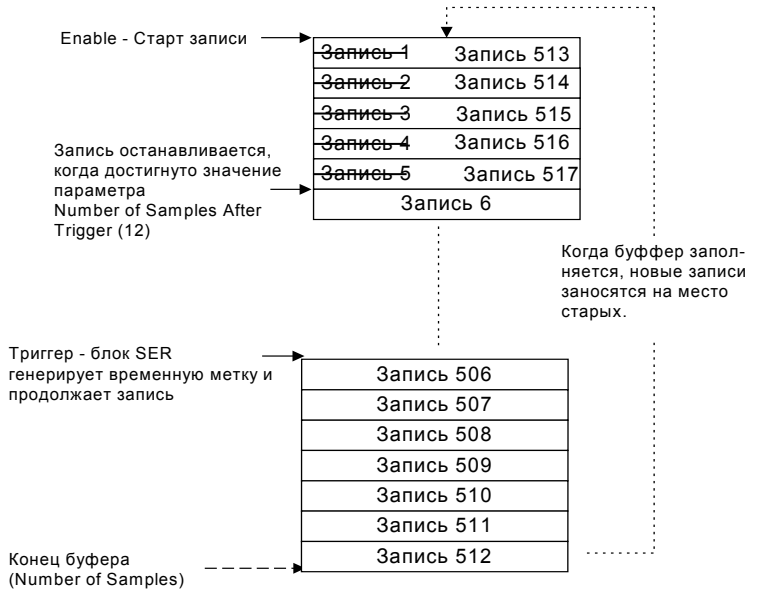


Рисунок 12-2. Сбор выборок на примере режима Mid-Trigger

Post Trigger

Производит непрерывное протоколирование, пока не будет достигнут параметр Number of Samples.

Для настройки этого режима установите для слова 10 значение равное параметру Number of Sample (слово 9). Когда на вход T поступит 1, протоколирование продолжится, пока не достигнет заданного числа.

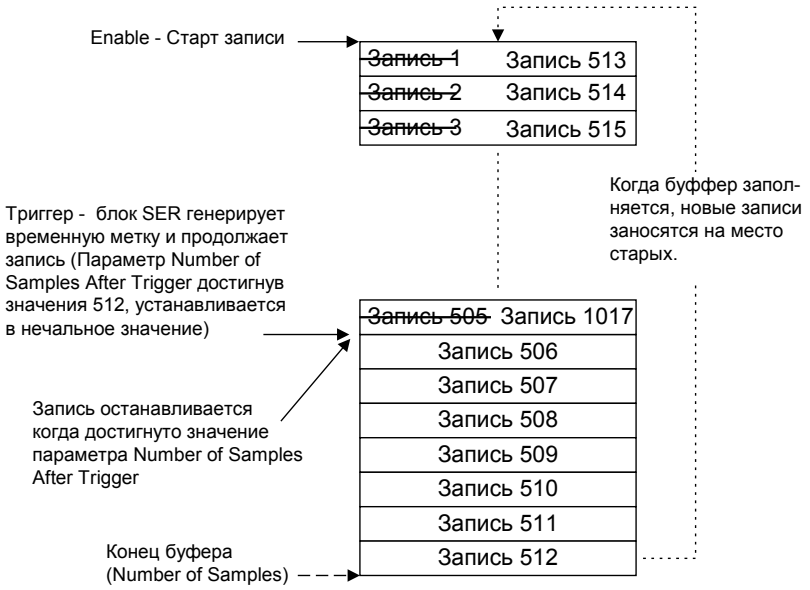


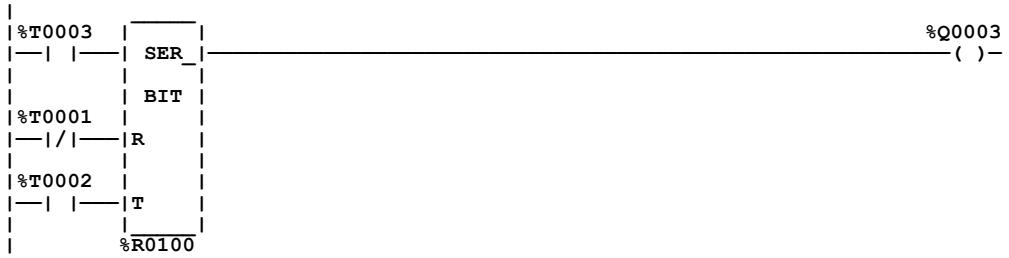
Рисунок 12-3. Сбор выборок на примере режима Post-Trigger

Режим Full Buffer

Если режим Trigger Mode установлен в 1, то параметр Number of Samples After trigger (слово 10) игнорируется и вход T не влияет на функционирование блока SER. Когда блок запущен, протоколирование продолжается до тех пор, пока число записей не превысит величины, установленной параметром Number of Samples (слово 8), происходит заполнение буфера. Когда буфер заполнится, запись остановится, сгенерируется метка времени и на выход OK блока SER поступит 1.

Пример блока SER

На этом примере показан блок, настроенный согласно таблице 12-1.



Пример блока управления

В данном примере система состоит из 16- канального дискретного модуля ввода, расположенного в главной стойке, слот 4, который может произвести 572 (512 + 60) записи. Вход Enable установлен в 1, а входы Reset и Trigger в 0.

Таблица 12-1. Пример блока управления для функции SER

Номер слова	Регистр	Параметр	Значение (десятичное)	Значение (шестнадцатеричное)	Описание
0	%R0100	Status	2	0002	Функциональный блок запущен. Это означает, что блок функционирует и производит протоколирование всякий раз, когда к нему обращается программа.
1	101	Status Extra Data	1	0001	Параметр «Status Extra Data» указывает, что будет взято более 512 записей и соответственно, буфер данных перезапуститься хотя бы один раз.
2	102	Trigger mode	0	0000	Регистратор событий настроен на режим триггера.
3	103	Trigger Time Format	0	0000	Время сохраняется в формате BCD
4	104	Reserved	0	0000	Данные параметры всегда установлены в 0.
5	105	Reserved	0	0000	
6	106	Reserved	0	0000	
7	107	Reserved	0	0000	
8	108	# of channels	24	0018	Размер данных для настройки выборки 24 бита.
9	109	# of samples to be taken	512	0200	В буфер помещается 512 записей. Заметьте, что размер буфера не 512 байт. Его размер 512 x (24/8) = 1536 байт или 768 слов. (Длина каждой записи 3 байта)
10	110	# of samples after trigger	12	000C	Параметр «Number of Samples After Trigger» равен 12.

Номер слова	Регистр	Параметр	Значение (десятичное)	Значение (шестнадцатеричное)	Описание
11	111	Input module slot	4	0004	Модуль ввода будет просканирован, следовательно, его текущие значения будут доступны для осуществления протоколирования.
12	112	Data Block Segment Selector	8	0008	Тип данных – 0x08 (%R).
13	113	Data Block Offset	200	00C8	Т.к. таблица регистров начинается с ячейки %R0001, данное смещение указывает, что начало блока данных находится в ячейке %R0001 + 200 = %R0201.
Характеристика каналов		В остальных словах данного блока располагаются описания каналов. В данном примере используется шесть описаний.			
14	114	Set. Sel. : Length	17921	4601	Первое описание канала: Данное описание канала выбирает сегмент памяти %I длинной 1 и смещение равное 0. Выбирается ячейка %I0001 для первого канала.
15	115	Offset	0	0000	
16	116	Seg. Sel. : Length	-253	FF03	Второе описание канала: Данное описание выбирает Нулевой переключатель с длиной 3 и смещением 0. Нулевой переключатель является причиной того, что каналы 2 - 4 будут пропущены. Эти каналы всегда содержат нулевую запись.
17	117	Offset	0	0000	
18	118	Seg. Sel. : Length	3	0003	Третье описание канала выбирает переключатель модуля ввода с длинной 3 и смещением 12. Данный переключатель позволяет производить запись из модуля, которая берется с каналов модуля 13, 14 и 15 для каналов 5-7.
19	119	Offset	12	0012	
20	120	Seg. Sel. : Length	18434	4802	Четвертое описание канала выбирает сегмент из памяти %Q длинной 2 и смешением 8. Ячейки %Q0009 и %Q0010 предназначены для каналов 8 и 9.
21	121	Offset	8	0008	
22	122	Seg. Sel. : Length	8	0008	Пятое описание – это другой переключатель модуля ввода. Он имеет длину 8 и смещение равное 0. Значения для каналов модуля с 1 по 8 размещаются в каналах 10 - 17.
23	123	Offset	0	0000	
24	124	Seg. Sel. : Length	-249	FF07	Шестое описание – это дополнительный нулевой переключатель. Он имеет длину 7 и смещение равное 0. Каналы с 18 по 24 заполняются нулями.
25	125	Offset	0	0000	

Содержимое записи

В таблице 12-2 приводятся все значения, содержащиеся в одиночной записи основанной на описании канала в блоке управления.

Таблица 12-2. Содержимое записи для функции SER

Номер канала	Содержимое канала
1	%I0001
2 - 4	Нули
5	13-ый канал модуля ввода
6	14-ый канал модуля ввода
7	15-ый канал модуля ввода
8	%Q0009
9	%Q0010
10 - 17	Каналы модуля ввода с 1-8
18 - 24	Нули

Пример блока данных для блока управления

Таблица 12-3 описывает блок данных, сконфигурированный в результате работы блока управления показанного на странице 12-19. Заметьте, что он начинается с регистра 201, как описано в параметре смещения сегментов (слова 12 и 13) блока управления.

Таблица 12-3. Пример блока данных для блока управления

Смещение	Регистр	Описание параметра	Значение (десятичное)	Значение (шестнадцатеричное)
0	%R0201	Current sample offset #	59	003B
1	202	Trigger sample offset #	0	0000
2 - 5	203 – 206	Trigger time (BCD)	0 0 0 0	0000 0000 0000 0000
6 - 768	207 – 975	Sample Buffer	sample data	sample data

Значение 59 параметра «Current Sample Offset» означает, что 59-я запись была последней записанной в буфер (не 59 регистров). Т.к каждая запись содержит 3 байта, то реальное значение смещения равно 59 * 3 = 177 байт или последний байт в 89 регистре. Таким образом, условие переключения триггера не выполнено и поэтому выход блока остался установленным в 0. Буфер данных содержит 512 записей, из которых 59 новых и 60 предыдущих.

Форматы временных меток блока управления

Пример времени срабатывания триггера: November 3, 1998 at 8:34:05:16 a.m.

BCD Format:

```
struct time_of_day_clk_rec {
        unsigned char  seconds;
        unsigned char  minutes;
        unsigned char  hours;
        unsigned char  day_of_month;
        unsigned char  month;
        unsigned char  year;
};
```

Регистр	Параметр	Значение (десятичное)	Значение (шестнадцатеричное)
%R0203	Минуты/Секунды	13317	3405
%R204	День месяца/часы	776	0308
%R205	Год/Месяц	-26607	9811
%R206	Не используется	0	0

POSIX Format:

```
struct timespec {
        long    tv_sec; /* Число секунд с 1 Января 1970 */
        long    tv_nsec;/* Число наносекунд в следующей секунде */
};
```

Регистр	Параметр	Значение (десятичное)	Значение (шестнадцатеричное)
%R0203	Секунды младшее слово	-7811	e17d
%R204	Секунды старшее слово	13845	3615
%R205	Наносекунды младшее слово	26624	6800
%R206	Наносекунды старшее слово	2441	0989

Функция END

Функция выполняет временную приостановку программы. Программа выполняется с первого до последнего шага или до функции END, смотря что, встретится раньше.

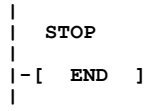
Функция END безоговорочно останавливает выполнения программы. Для функции END отводится целый шаг программы. Команды расположенные после функции END не выполняются, и ЦП возвращается к началу программы для запуска нового цикла.

Функция END полезна при отладке программы, потому что она предотвращает выполнение любых команд, которые следуют после выполнения функции END.

Программное обеспечение Logicmaster может использовать специальный маркер [END OF PROGRAM LOGIC] для отметки конца выполнения программы. Этот маркер может использоваться даже в случае отсутствия функции END в программе.

Пример

В следующем примере функция END запрограммирована для прекращения выполнения программы в конце цикла ЦП.



Примечание

Помещение функции END в программу, написанную на языке SFC или в программу вызываемую блоком SFC, вызовет сбой “END Function Executed from SFC Action” в ЦП версии 7 и старше. (В предыдущих версиях ЦП функция END работает некорректно, но сбой не регистрируется). Для получения полной информации по данному сбою (“System Configuration Mismatch”) обратитесь к Главе 3, разделу 2 настоящего руководства.

Функция MCRN/MCR

Функция Master Control Relay (MCR) должна использоваться с функцией End Master Control Relay (ENDMCR). Функции должны иметь одинаковые имена. Все шаги программы между включенной функцией MCR и соответствующей ей функцией ENDMCR выполняются без инициализации обмоток. Функция ENDMCR связанная с функцией MCR возвращает программу к нормальному функционированию. В отличие от команды JUMP, функция MCR работает только вперед. Функция ENDMCR должна появляться в программе после объявления функции MCR.

Функцией MCR накладываются следующие директивы:

- Таймеры не увеличивают и не уменьшают своего значения. Таймер типа TMR сбрасывается. Для функционального блока ONDTR аккумулятор удерживает его значение.
- Обычные обмотки переходят в режим выключено, инверсные обмотки переходят в режим включено.

Примечание

Когда функция MCR включена, подконтрольная ей часть программы сканируется и отображается текущий статус контактов, но обмотки остаются без питания. Если Вы не осведомлены о том, какая часть программы находится под контролем функции MCR, то возможно появление сбоя. Для визуального выделения подконтрольной части, в ПО используется вторая шина питания.

Программное обеспечение Logicmaster 90-30/20/Micro поддерживает две формы функции MCR, вложенную и не вложенную формы.

Совместимость с ЦП

Тип ЦП	Скрытое сравнение команд
ЦП серии 35x и 36x (Версия 2 и старше)	Используйте только вложенную форму (MCRN)
Версия 1 ЦП Series 90	Используйте только не вложенную форму (MCR)

Действие функции MCRN

Функция MCRN может быть использована в любом месте программы, так долго как это необходимо, если она не мешает другим функциям MCRN и не пересекается с областью вложенной функции MCR или не вложенной функции JUMP.

Если пара функций MCRN/ENDMCRN вложена в другую пару функций MCRN/ENDMCRN, она должна полностью находиться во второй паре. Позволяется использовать до восьми вложенных пар. Пример смотрите на странице 12-26.

Примечание

В ЦП серий 35x и 36x используйте только одну функцию MCR для каждой функции ENDMCR.

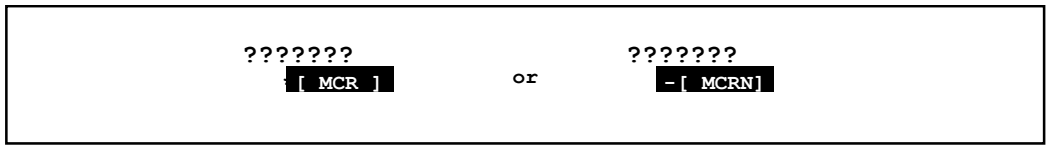
Совместно с одной функцией ENDMCR можно использовать несколько функций MCRN (кроме ЦП серий 35x и 36x, как отмечено выше). По аналогии с командой JUMP, когда Вы можете использовать несколько команд JUMP с одной и той же меткой. Для определения конкретных различий между функциями JUMP и MCR обратитесь к заголовку «Различия между функциями MCR и JUMP» на странице 12-25.

Действие функции MCR

Каждой функции ENDMCR должна соответствовать только одна функция MCR. Область вложенных функций MCR и ENDMCR не могут перекрывать или содержать область другой пары функций MCR/ENDMCR или JUMP/LABEL. Не вложенные функции MCR не могут находиться внутри пары функций JUMP/LABEL.

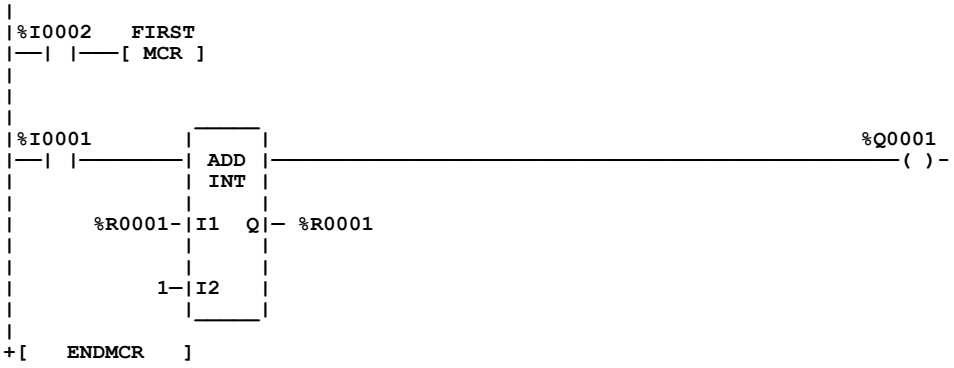
Параметры

Оба вида функций MCR имеют одинаковые параметры. Они имеют вход булевого типа EN и имя, которое позволяет их идентифицировать. Это имя вновь используется для функции ENDMCR. Ни функция MCR, ни MCRN не имеют выходов; для данной функции отводится целый шаг, т.е. другие функции на данном шаге не используются.



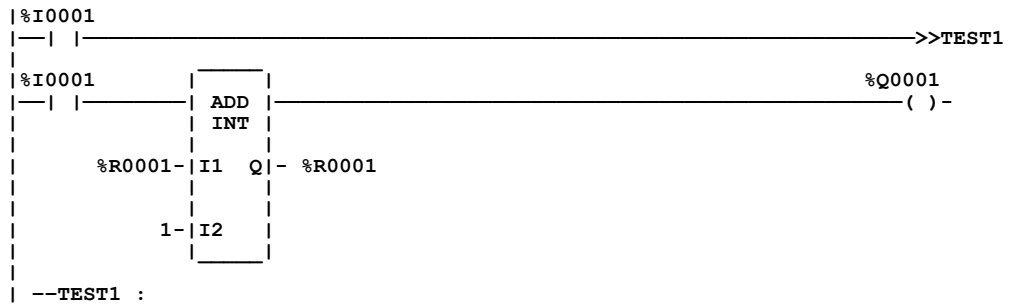
Различия между функциями MCR и JUMP

При использовании функции MCR, функциональные блоки в пределах ее границ, **выполняются без энергии** и обмотки **находятся в выключенном состоянии**. В следующем примере, если ссылка %I0002 установлена в 1, функция MCR включена. Когда функция MCR включена – даже если ссылка %I0002 установлена в 1 – функциональный блок ADD выполняется **без** инициализации (т.е. он не добавит 1 к значению, содержащемуся по адресу %R0001), ссылка %Q0001 будет содержать значение 0.



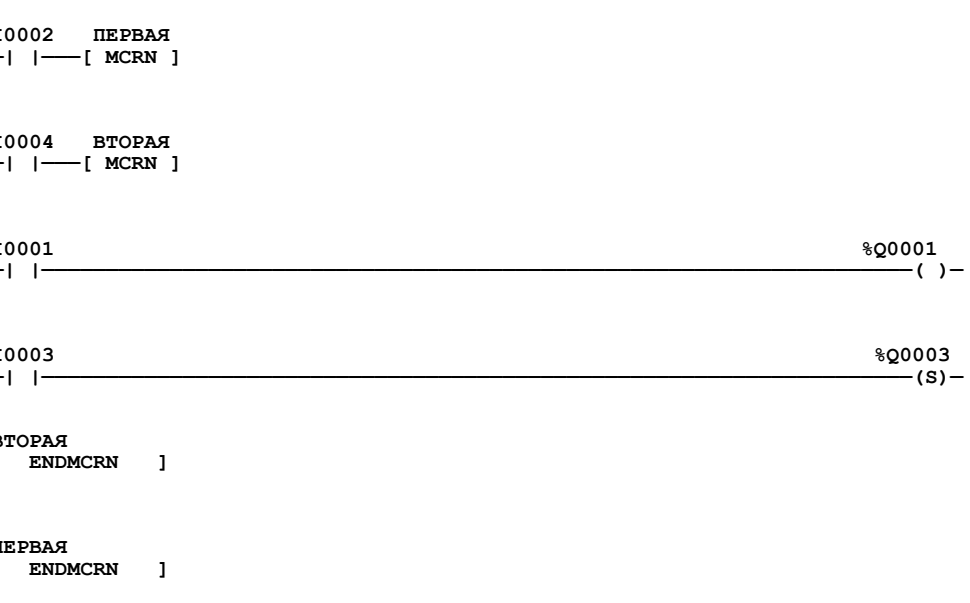
При использовании функции JUMP, любые функциональные блоки между командой JUMP и LABEL **не выполняются,** а обмотка **не используется**. В следующем примере, если ссылка

%I0002 установлена в 1, Используется функция JUMP. Часть программы между командой JUMP и LABEL пропускается, ссылка %Q0001 не затрагивается (т.е. если она имела значение 1, то она и сохранит 1; если она имела значение 0, то она и сохранит 0).



Пример

Следующий пример демонстрирует использование одной функции MCRN, назовем ее «Вторая», внутри другой функции MCRN, назовем ее «Первая». Всякий раз, когда функция MCRN инициализируется при помощи контакта %I0002, выполнение программы продолжается без активации обмотки, пока не встретится связанная с функцией MCR функция NDMCR. Если контакты %I0001 и %I0003 содержат значение 1, то обмотка %Q0001 выключается, а обмотка %Q0003 сохраняет свое значение.

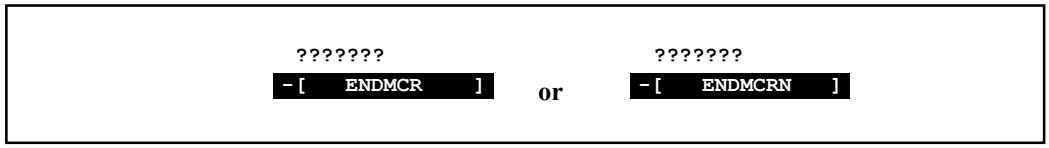


Функции ENDMCRN и EDNMCR

Для возврата программы к нормальному функционированию после использования функции MCR, используйте функцию End Master Control Relay (ENDMCR). Когда функция MCR связанная с ENDMCR задействована, функция ENDMCR восстанавливает нормальное функционирование программы. Когда функция MCR связанная с ENDMCR не задействована, функция ENDMCR не используется.

Программное обеспечение Logicmaster 90-30/20/Micro поддерживает две формы функции ENDMCR, вложенную и не вложенную формы. Не вложенная форма должна использоваться с не вложенной формой функции MCR, а вложенная форма функции ENDMCR должна использоваться с вложенной формой функции MCR

Функция ENDMCR имеет инверсный булевый вход EN. Команда на включение должна быть подключена к шине питания; выполнение данного условия не может быть условным. Функция ENDMCR также имеет имя, которое определяет данную функцию и связывает ее с соответствующими функциями MCR. Функция ENDMCR не имеет выхода и на данном шаге ничего кроме этой функции быть не должно.



Пример

Следующий пример показывает использование функции ENDMCR для прекращения действия функции MCR на участке “clear”.

Пример не вложенной функции ENDMCR.

```
|
|  CLEAR
|
|-[   ENDMCR    ]
|
```

Пример вложенной функции ENDMCR.

```
|
|  CLEAR
|
|-[   ENDMCRN   ]
|
```

Функция JUMP

Использование команды JUMP позволяет пропустить часть программы. Выполнение программы будет продолжено с команды помеченной специальной меткой (LABEL). Когда команда JUMP задействована, все обмотки, имеющие некоторую область действия, сохраняют свое предыдущее состояние. К таким обмоткам относятся обмотки, использующиеся совместно с таймерами, счетчиками, защелками и реле.

Программное обеспечение Logicmaster 90-30/20/Micro поддерживает две формы функции JUMP, вложенную форму и не вложенную. Не вложенная форма функции JUMP была доступна вместе с первой версией программного обеспечения и имела следующий вид

————————>>LABEL01, где LABEL01 это имя соответствующее не вложенной функции LABEL.

При использовании не вложенных функций JUMP, необходимо учитывать, что каждой функции LABEL должна соответствовать только одна функция JUMP. Функция JUMP может выполняться как в прямом, так и в обратном направлении.

Диапазон не вложенных функций JUMP и LABEL не должен перекрывать диапазон другой пары функций JUMP/LABEL или пары функций MCR/ENDMCR. Пара не вложенных функций JUMP/LABEL не может находиться в области другой пары функций JUMP/LABEL или MCR/ENDMCR. Также, пара функций MCR/ENDMCR или JUMP/LABEL не может находиться в области не вложенной пары функций JUMP/LABEL.

примечание

В ПЛК Версии 1 Series 90-30 можно использовать только не вложенную форму функции JUMP. Для более новых версий микропрограммы рекомендуется использовать вложенную форму функции JUMP.

Важно отметить, что ЦП серий 35x и 36x поддерживают только вложенную форму функции JUMP. Не вложенные формы этой функции в таких ЦП не поддерживаются.

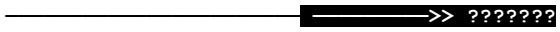
Вложенная форма функции JUMP имеет следующий вид, ————N——>>LABEL01, где LABEL01 это имя соответствующее не вложенной функции LABEL. Она доступна во второй и более поздних версиях программного обеспечения Logicmaster 90-30/20/Micro и микропрограмме ПЛК.

Вложенную форму функции JUMP можно использовать в любом месте программного кода до тех пор, пока она не затронет диапазон другой функции JUMP или MCR.

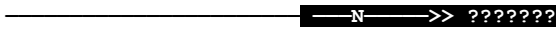
При использовании вложенных функций JUMP, одной функции LABEL может соответствовать несколько функций JUMP. Вложенная форма функции JUMP может выполняться как в прямом, так и в обратном направлении.

Обе формы функции JUMP всегда помещаются в 9 и 10 колонку текущего шага; после команды JUMP в данном шаге ничего не должно быть. Инициализируется шаг, отмеченный меткой (LABEL).

Не вложенная форма функции JUMP:



Вложенная форма функции JUMP:



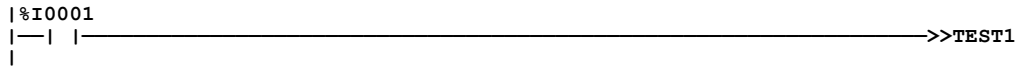
Предупреждение

Для предотвращения появления бесконечно повторяющегося цикла с прямыми и обратными функциями JUMP, обратная функция JUMP должна содержать условие для перехода.

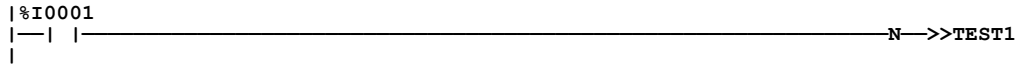
Примеры

На следующих примерах показано, что всякий раз, когда функция JUMP TEST1 включается, инициализируется шаг, отмеченный меткой LABEL TEST1.

Пример не вложенной формы функции JUMP:



Пример вложенной формы функции JUMP:



Функция LABEL

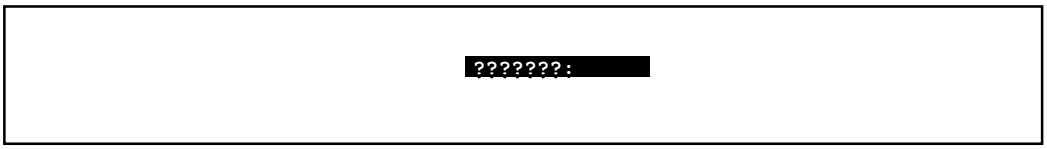
Команда LABEL действует, как «цель» для функции JUMP. Использование команды LABEL позволяет отклонить программу от обычного последовательного выполнения команд.

Каждая функция LABEL должна иметь свое оригинальное имя. При отсутствии согласованности между парами функций JUMP/LABEL, программа может быть создана и записана в ПЛК, но выполняться она не будет.

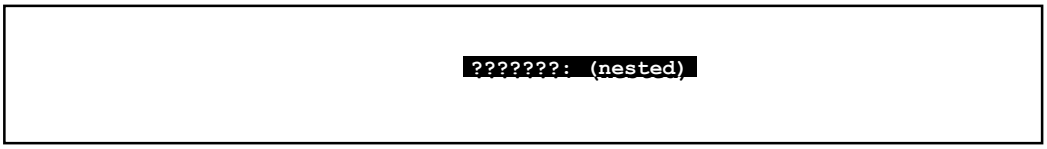
Программное обеспечение Logicmaster 90-30/20/Micro поддерживает две формы функции LABEL, вложенную форму и не вложенную. Не вложенная форма функции LABEL01 должна использоваться вместе с не вложенной формой функции JUMP————————>>LABEL01. вложенная форма, LABEL01:(nested) должна использоваться совместно с вложенной формой JUMP ———N——>>LABEL01.

Функция LABEL не имеет ни входов, ни выходов; на данном шаге никаких функций больше использовать нельзя.

Не вложенная форма функции LABEL:



Вложенная форма функции LABEL:



Пример

На следующем примере показано, как происходит инициализация функции LABEL TEST1 после выполнения функции JUMP TEST1.

Пример не вложенной формы функции LABEL:

```
|
| TEST1   :
|
```

Пример вложенной формы функции LABEL:

```
|
| TEST1   :(nested)
|
```

Функция COMMENT

Используйте функцию COMMENT для ввода комментария в программу. Комментарий может до 2048 символов. Вид функции COMMENT на языке лестничной логики выглядит следующим образом:

Текст комментария может быть прочитан или изменен простым подведением курсора к (* COMMENT *), после выделения шага и выбора инструментального средства Zoom (F10). Текст комментария также может быть выведен на печать.

Для создания более длинного комментария с возможностью вывода на печать, нужно использовать файл аннотации. Процесс создания показан ниже:

1. Создание комментария:

   A. Вводите необходимый текст, пока не встретите текст другого файла.

   B. Перейдите в начало новой строки и введите **I** или **i**, путь, куда должен сохраниться данный файл и имя для этого файла.

   ```
   \I d:\text\commnt1
   ```

   Если файл расположен в той же папке, что и программа, то путь для файла указывать не обязательно.

   C. Продолжение редактирования программы или выход в MS-DOS

2. После отключения программатора, создайте текстовый файл, используя любое MS-DOS совместимое программное обеспечение. Дайте файлу имя, приведенное в тексте комментария и разместите его в том месте, путь на которое указанно в комментарии.

Функция SVCREQ

Используйте функцию SVCREQ (Service Request) для формирования запроса для сервисов ПЛК представленных ниже:

Таблица12-4. Функции сервисных запросов

Функция	Описание
1	Изменение/чтение значения таймера постоянной цикла.
2	Чтение переменных.
3	Изменение режима и времени работы Окна Связи с программатором.
4	Изменение режима и времени работы Окна Связи с системой.
6	Изменение/чтение состояния контрольной суммы и количества слов использующихся для нее.
7	Изменение/чтение значения таймера астрономического времени.
8	Сброс сторожевого таймера.
9	Считать значение времени цикла от начала цикла.
10	Считать имя папки.
11	Считать ID ПЛК.
12	Считать состояние Run/Stop ПЛК.
13	Выключение ПЛК.
14	Очистить таблицы сбоев.
15	Считать последнюю запись в таблице сбоев.
16	Считать значение таймера времени работы.
18	Считать статус подстановок В/В.
23	Считать основную контрольную сумму.
26/30	Опрос системы В/В.
29	Считать значение таймера времени простоя.
45	Пропустить следующее сканирование В/В.
46	Получить доступ к статусу системной шины.

Обзор функции SVC REQ

Функция SVCREQ имеет три входных параметра и один выходной. Когда функция SVCREQ инициализируется, ПЛК производит запрос на выполнение, той функции, номер которой указан на входе FNC. Значения начальных параметров начинаются со ссылки указывающейся на входе PARM. Функция SVCREQ не выполняется, если введен неправильный номер функции, используются неправильные параметры или произошел выход из диапазона используемых ссылок. Дополнительные причины возникновения сбоев описаны на следующей странице.

Ссылки, использующиеся на входе PARM, могут располагаться в любом типе памяти (%R, %AI, or %AQ). Данные ссылки являются первыми в группе ссылок создающих «блок параметров» для функции. Последующие 16-ти бит предназначены для запоминания дополнительных параметров. Итоговое количество ссылок требующихся функции зависти от типа использующейся функции SVCREQ.

Блоки параметров могут быть использованы как на входе, так и на выходе, в качестве области памяти для запоминания данных после выполнения функции. Поэтому, данные, возвращаемые функцией доступны из области памяти выделенной для входа PARM.

Параметры

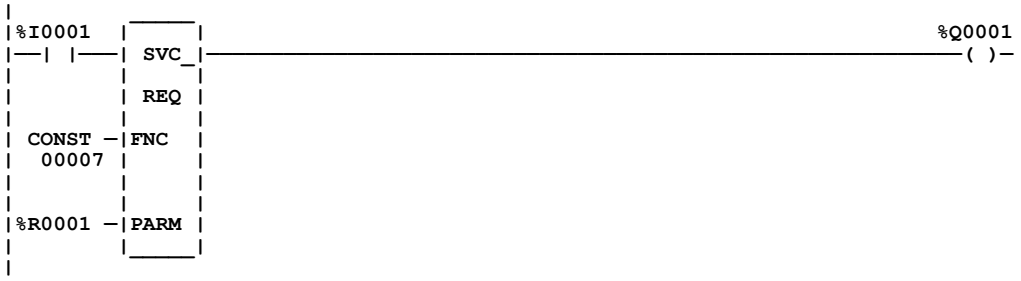
Параметр	Описание
Enable	Когда вход Enable активируется, выполняется запрос на вызов функции.
FNC	Вход FNC содержит постоянную или ссылку необходимую для вызываемого запроса.
PARM	Вход PARM содержит начальную ссылку для блока параметров необходимого для вызываемого запроса.
ok	Выход ok активируется при условии безошибочного выполнения функции.

Допустимые типы памяти

Параметр	состояние	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	постоянная	нет
enable	•											
FNC		•	•	•	•		•	•	•	•	•	
PARM		•	•	•	•		•	•	•	•		
ok	•											•

Пример

Когда вход Enable инициализируется, т.е. контакт %I0001 устанавливается в 1, вызывается функция SVCREQ под номером 7, с блоком параметров, начало которого расположено в ссылке %R0001. Выходная обмотка устанавливается в 1, если операция выполнена успешно.



SVCREQ #1: Изменение/чтение значения таймера постоянного цикла

Начиная с ЦП Series 90-30 Версии 8, Вы можете использовать этот сервисный запрос #1 для:

- Отключения режима **CONSTANT SWEEP**.
- Включения режима **CONSTANT SWEEP** с использованием предыдущего значения таймера.
- Включения режима **CONSTANT SWEEP** с использованием нового значения таймера.
- Установки нового значения для таймера.
- Считывания статуса режима **CONSTANT SWEEP** и значения таймера.

Примечание

Сервисный запрос #1 поддерживается только ЦП Series 90-30, начиная с версии 8.

Блок параметров имеет длину два слова.

Для отключения режима **CONSTANT SWEEP**, введите функцию системного запроса #1 со следующим блоком параметров:

0	address
Игнорируется	address + 1

Для включения режима **CONSTANT SWEEP**, введите функцию системного запроса #1 со следующим блоком параметров:

0	address
Игнорируется	address + 1

Примечание

Если таймер будет использовать новое значение, введите его на месте второго слова. Если значение таймера меняться не будет, введите на месте второго слова – 0 (нуль). Если таймер вообще не имеет никакого значения, ввод нуля приведет к установке на выходе значения 0 (нуль).

Для изменения значения таймера **без** изменения статуса работы цикла, используйте следующий блок параметров:

2	address
Новое значение	address + 1

Для считывания статуса и значения таймера, используйте следующий блок параметров:

2	address
Новое значение	address + 1

Примечание

После использования функции сервисного запроса #1 с блоками параметров, приведенными выше, ЦП Версии 8 и старше возвратит значение 0 для Обычного цикла, и значение 1 для Постоянного цикла. Не путайте это значение с входящими значениями представленными ниже:

Выполнение функции произойдет успешно, кроме следующих случаев:

1. Номер действия введен не верно:

0	Отключение режима **CONSTANT SWEEP**.
1	Включение режима **CONSTANT SWEEP**.
2	Установки нового значения для таймера.
3	Считывания статуса режима **CONSTANT SWEEP** и значения таймера.

2. Значение таймера превышает величину 2550 мс (2.55 секунды).
3. Постоянная времени цикла не была запрограммирована или предыдущее значение было нулевым.

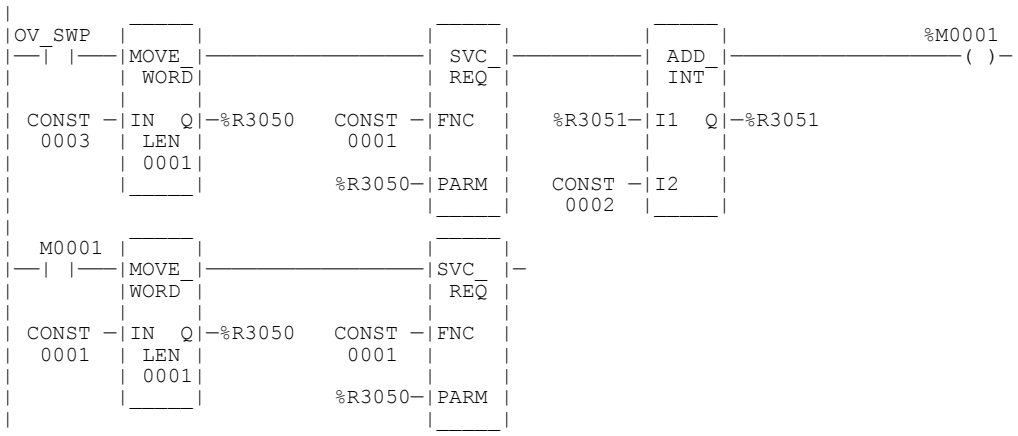
После выполнения функции, она возвращает статус таймера и переменную, сохраняя их в боке переменных:

0 = Отключен	
1 = Включен	address
Новое значение	address + 1

Если address + 1 содержит 16-тиричное значение FFFF, значение таймера не было запрограммировано.

Пример

Данный пример демонстрирует работу программного блока. Когда включающий контакт OV_SWP установлен в 1, считывается постоянная времени цикла, таймер увеличивает свое значение на две миллисекунды, и новое значение постоянной посылается обратно в ЦП. Блок параметров хранит свои значения, начиная с ссылки %R3050. Так как функция MOVE не поместилась на экране, в данном примере была использована дополнительная обмотка для промежуточного сохранения признака успешного выполнения первого шага. Для любых циклов, в которых контакт OV_SWP установлен в 0, обмотка %M0001 не используется.



SVCREQ #2: Считывание параметров окон

Для получения временных значений установленных для Окна связи с программатором, Окна связи с системой и Окна фоновых задач, используйте сервисный запрос #2.

Примечание

Сервисный запрос #2 поддерживается только ЦП Series 90-30, начиная с версии 8.0.

Существует три режима работы каждого окна:

Наименование режима	Значение	Описание
Ограниченный режим	0	Время работы окна ограничено значением, установленным по умолчанию или значением определенным сервисным запросом #3 для Окна связи с программатором или сервисным запросом #4 для Окна связи с системой. Работа окна будет прекращена, когда все задачи будут выполнены.
Постоянный режим	1	Каждое окно будет работать в режиме **РАБОТЫ ДО ЗАВЕРШЕНИЯ,** и время его работы будет складываться из суммы значений установленных для каждого окна. Если одно из трех окон переведено в **ПОСТОЯННЫЙ** режим, остальные два окна также будут переведены в **ПОСТОЯННЫЙ** режим. Если ПЛК работает в **ПОСТОЯННОМ** режиме и значение времени работы не определено функцией сервисного запроса, то по умолчанию используется значение, посчитанное для постоянного режима.
Режим «Выполнять до завершения»	2	Независимо от установленных значений времени работы окон в данном режиме, они будут выполняться до тех пор, пока не будут завершены все задачи.

Когда значение времени работы равно нулю, окно считается выключенным.

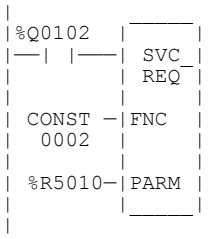
Блок параметров окна имеет длину три слова:

	Старший байт	Младший байт	
Окно связи с программатором	Режим	Величина в мс	address
Окно связи с системой	Режим	Величина в мс	address + 1
Окно фоновых задач			address + 2

Все параметры являются выходными параметрами. При объявлении данной функции в программе можно не вводить значения параметров в блок параметров. Входные значения для всех окон даются в миллисекундах.

Пример

На следующем примере показано, что когда разрешающий контакт %Q0102 замкнут (установлен в 1), то программа ПЛК помещает текущие значения параметров времени работы окон в блок параметров, начиная с отметки %R5010.



SVCREQ #3: Изменение режима и времени работы Окна Связи с программатором

Используйте сервисный запрос #3 для изменения режима и времени работы Окна связи с программатором. Изменения вступят в силу на следующем цикле ЦП.

Примечание

Сервисный запрос #3 поддерживается только ЦП Series 90-30, начиная с версии 8.0.

Сервисный запрос 3 будет инициализирован, пока не будет выбран один из трех режимов: 0 (Ограниченный), 1 (Постоянный), 2 (Выполнять до завершения).

Блок параметров имеет длину в одно слово.

Для выключения окна связи с программатором, используйте следующий блок параметров с сервисным запросом 3:

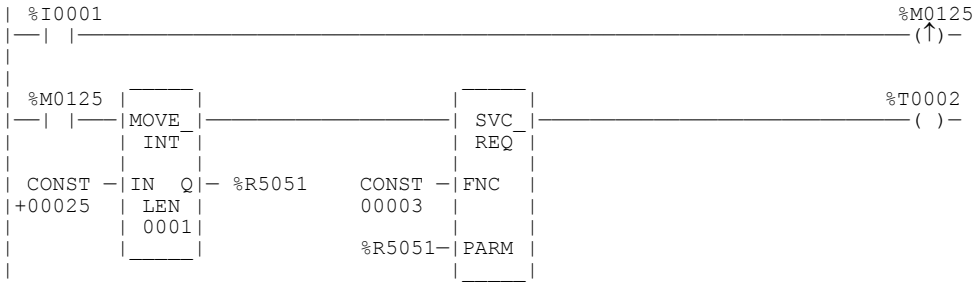
Старший байт	Младший байт	
0	0	address

Для включения окна связи с программатором, используйте следующий блок параметров:

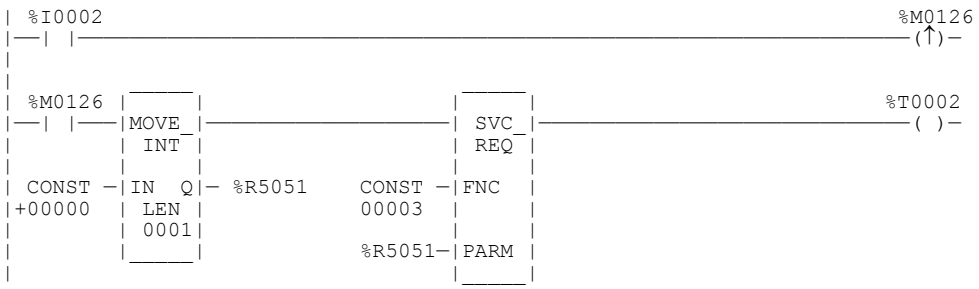
Старший байт	Младший байт	
Режим	Значение от 1 до 255 мс	address

Пример

Когда контакт %M0125 установлен в 1, окно связи с программатором включается, и значение временного параметра принимает значение 25 мс. Блок параметров располагается в памяти по адресу %R5051.



Для выключения окна связи с программатором используйте сервисный запрос 3 с нулевым значением временного параметра. В следующем примере показано, что при установленном в 1 контакте %M0126, окно связи с программатором включается, и значение временного параметра принимает значение 0 мс. Блок параметров располагается в памяти по адресу %R5051.



SVCREQ #4: Изменение режима и времени работы Окна Связи с системой

Используйте сервисный запрос #4 для изменения режима и времени работы Окна связи с системой. Изменения вступят в силу на следующем цикле ЦП.

Примечание

Сервисный запрос #4 поддерживается только ЦП Series 90-30, начиная с версии 8.0.

Сервисный запрос 3 будет инициализирован, пока не будет выбран один из трех режимов: 0 (Ограниченный), 1 (Постоянный), 2 (Выполнять до завершения).

Блок параметров имеет длину в одно слово.

Для выключения окна связи с системой, используйте следующий блок параметров с сервисным запросом 3:

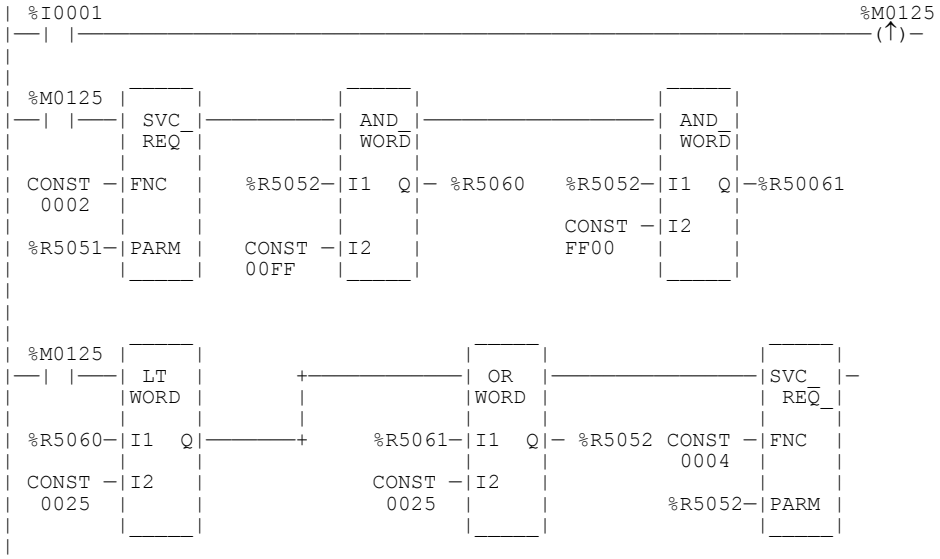
Старший байт	Младший байт	
0	0	address

Для включения окна связи с системой, используйте следующий блок параметров:

Старший байт	Младший байт	
Режим	Значение от 1 до 255 мс	address

Пример

На следующем примере, когда разрешающий контакт %M0125 установлен в 1, происходит считывание режима и значения времени работы окна связи с системой. Если значение переменной времени больше 25 мс, то она остается без изменений, в противном случае устанавливается в 25 мс. Так или иначе, после завершения данного шага окно связи с системой запускается. Блок параметров для всех трех окон располагается в памяти по адресу %R5051. Так как режим и временной интервал для окна связи с системой расположен во втором параметре блока параметров возвращаемый сервисным запросом #2, текущее значение временного интервала располагается в младшем байте ячейки памяти %R5052.



SVCREQ #6: Считывание/Изменение текущего количества слов контрольной суммы

Используйте сервисный запрос #6 для выполнения следующих задач:

- Считывания текущего количества слов.
- Установки нового количества слов.

Выполнение данной функции будет происходить успешно, до тех пор, пока не будет введено число отличное от 0 или 1 (см. ниже).

Для выполнения функций по работе с контрольной суммой, блок параметров имеет длину в два слова.

Считывания текущего количества слов

Для выполнения этого действия используйте функцию сервисного запроса 6 со следующим блоком параметров:

0	address
Игнорируется	address + 1

После выполнения, данная функция возвратит значение контрольной суммы, расположив его во втором слове блока параметров. Область памяти не резервируется для этой функции; возвращаемое значение – это и есть количество слов, которые использовались при последнем вычислении контрольной суммы.

0	address
Текущее количество слов	address + 1

Установки нового количества слов

Для выполнения этого действия используйте функцию сервисного запроса 6 со следующим блоком параметров:

1	address
Новое количество слов	address + 1

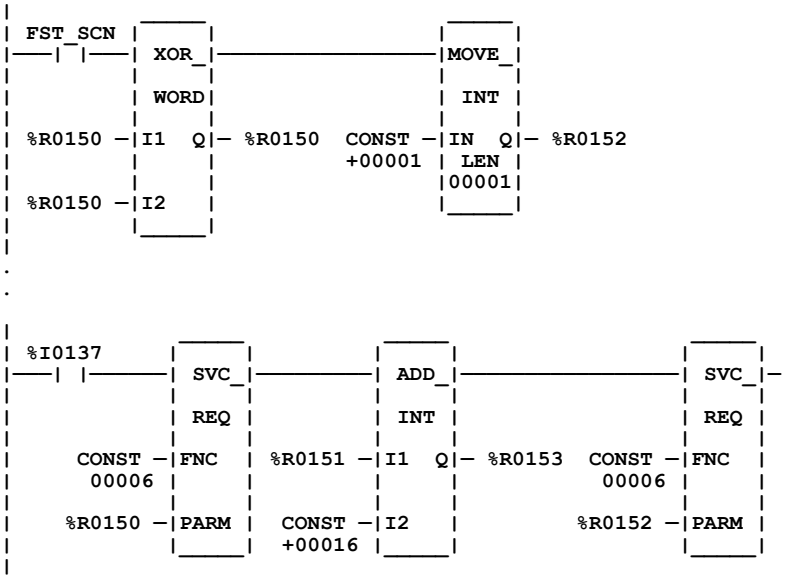
Введите 1 для установки нового количества слов, расположенных во втором слове блока параметров, использующихся при контрольном суммировании. Для ЦП моделей 331 и старше это значение может быть либо 0, либо 32. Для 221 модели это значение может быть или 0 или 4.

Примечание

Этот сервисный запрос не доступен в ПЛК Micro.

Пример

На следующем примере показано, что когда разрешающий контакт FST_SCN установлен в 1, происходит создание блока параметров для функции работы с контрольной суммой. Позднее в программе, когда входящий контакт %I0137 устанавливается в 1, считывается количество слов используемых при контрольном суммировании. Далее это число увеличивается на 16, и результат записывается в параметр «удержать новое значение». Далее, функция сервисного запроса #2 запрашивает ПЛК на установку нового значения.



Пример блоков параметров расположенных по адресу %R0150. Они могут иметь следующее содержание:

0 = читать текущее значение	%R0150
Удержать текущее значение	%R0151

1 = установить текущее значение	%R0152
Удержать новое значение	%R0153

SVCREQ #7: Считывание/Изменение значения таймера астрономического времени

Сервисный запрос #7 используется для считывания и установки значения астрономического таймера в ПЛК.

Примечание

Эта функция доступна только для ЦП 331 и старше Series 90-30 и для 28-ми канального ЦП Series 90 Micro (это модели, IC693UDR005, IC693UAA007, и IC693UDR010) и 23-х канального ЦП Series 90 Micro (IC693UAL006).

Данная функция не выполниться удачно, если:

1. В качестве запрашиваемого значения введено число отличное от 1 или 0 (см. ниже).
2. Указан неверный формат данных.
3. Итоговые данные имеют неправильный формат.

Для функций работающих со временем и датой длина блока параметров зависти от выбранного формата представления этих данных. Формату BCD необходимо 6 слов, сжатый формат ASCII требует 12 слов.

0 = считать время и дату 1 = установить время и дату	address
1 = формат BCD 3 = сжатый формат ASCII	address + 1
данные	address + 2 до конца

В первом слове определяется действие, которое необходимо произвести над данными:

```
0    =    считывание
1    =    изменение
```

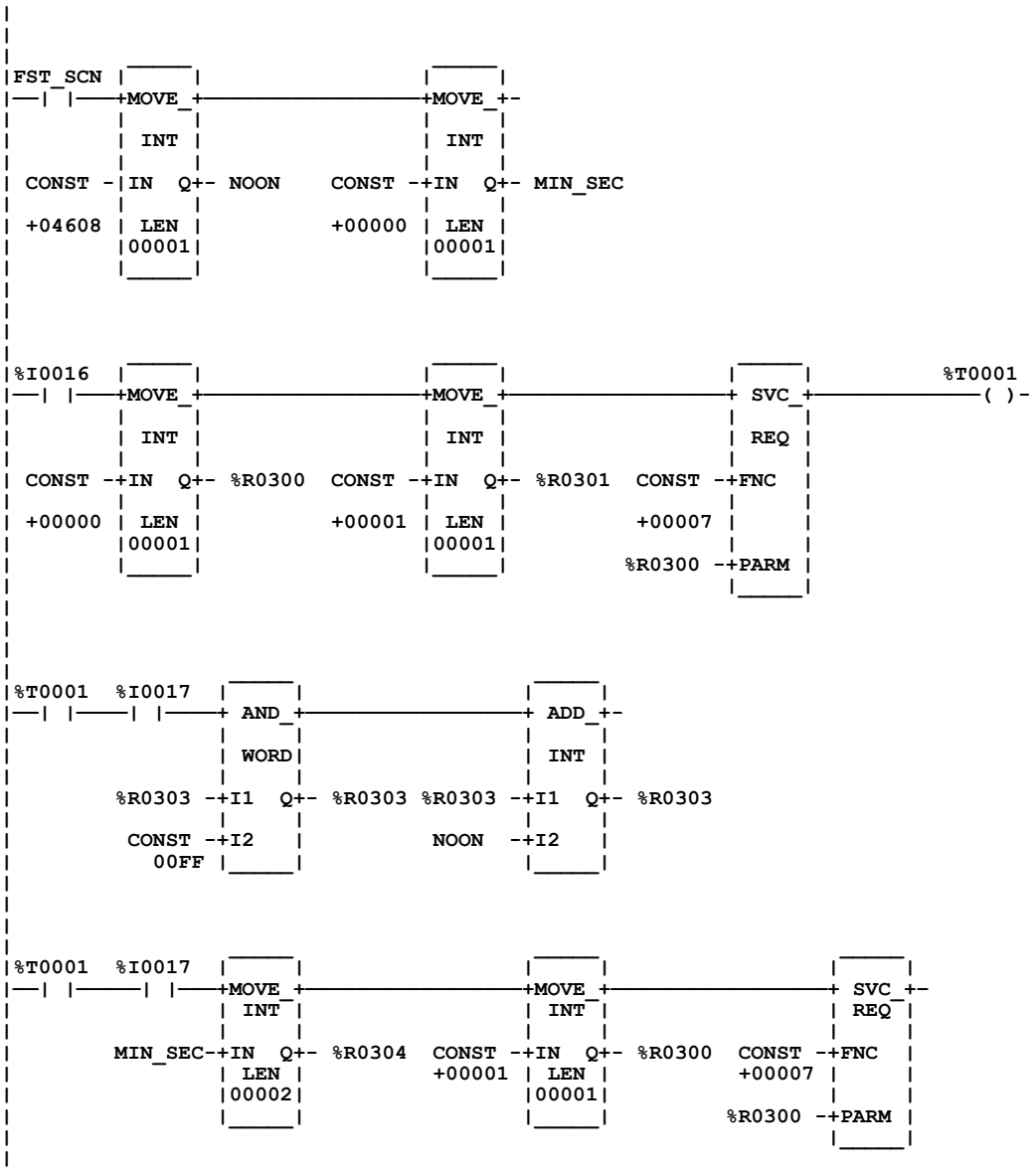
Во втором слове определяется формат данных:

```
1    =    формат BCD
3    =    сжатый формат ASCII с форматированием
```

От третьего слова и до конца блока параметров располагаются данные, возвращенные функцией, если происходило считывание или новые данные, если происходило изменение. В обоих случаях формат этих данных похож. При считывании даты и времени, слова с ячейки address+2 до ячейки address+8 игнорируются при подготовке запроса.

Пример

На следующем примере показано, что, будучи вызванным, из предыдущего шага программы, создается блок параметров для первого запроса текущей даты и времени и затем происходит установка времени на 12^{00} пополудни в формате BCD. Блок параметров располагается в глобальных данных по адресу %R0300. Массив NOON создается в программе, и он должен содержать значения 12, 0 и 0. (этот массив также должен содержать данные из ячейки %R0300). Для формата BCD отводится шесть последовательных ячеек памяти под блок параметров.



Содержание блока параметров

Содержание блока параметров для обоих форматов данных приведено ниже.

- Время в часах сохраняется в 24-часовом формате.
- День недели – это численное значение:

Значение	День недели
1	Воскресенье
2	Понедельник
3	Вторник
4	Среда
5	Четверг
6	Пятница
7	Суббота

Изменение/считывание даты и времени с использованием формата BCD:

В формате BCD для каждого элемента даты и времени отводиться один байт. Для этого формата требуется 6 слов памяти. Последний байт шестого слова не используется. При установке даты и времени этот байт игнорируется; при считывании функция возвращает нулевой символ (00).

Старший байт	младший байт		Пример блока параметров на выходе: считывание даты и времени в формате BCD (ВС., Июль 3, 1988, в 2:45:30 p.m.)	
1 = изменение или	0 = считывание	address	0	
1		address + 1	1	
месяц	Год	address + 2	07	88
Часы	День месяца	address + 3	14	03
Секунды	Минуты	address + 4	30	45
(нуль)	День недели	address + 5	00	01

.

Изменение/считывание даты и времени с использованием сжатого формата ASCII с форматированием:

В сжатом формате ASCII каждая цифра даты и времени представляется в виде отформатированного ASCII-байта. В дополнение, в данный формат введены пробелы и колонки для предоставления возможности корректного вывода на печатающее устройство или дисплей. Этот формат требует 12 слов памяти.

Старший байт	младший байт		Пример блока параметров на выходе: считывание даты и времени в сжатом формате ASCII (Пн, Окт. 2, 1989 в 23:13:00)	
1 = изменение или	0 = считывание	address	0	
3		address + 1	3	
Год	Год	address + 2	39	38
месяц	(пробел)	address + 3	31	20
(space)	Месяц	address + 4	20	30
День месяца	День месяца	address + 5	32	30
часы	(пробел)	address + 6	32	20
:	Часы	address + 7	3A	33
Минуты	минуты	address + 8	33	31
секунды	:	address + 9	30	3A
(пробел)	секунды	address + 10	20	30
День недели	День недели	address + 11	32	30

SVCREQ #8: Сброс сторожевого таймера

Используйте сервисный запрос #8 для сброса сторожевого таймера во время цикла ЦП.

Примечание

Сервисный запрос #8 поддерживается только ЦП Series 90-30, начиная с версии 8.0.

Когда время работы сторожевого таймера истекает, ПЛК выключается без предупреждения. Данная функция позволяет таймеру приостановить работу во время выполнения задач отнимающих много времени (например, во время ожидания ответа коммуникационного устройства).

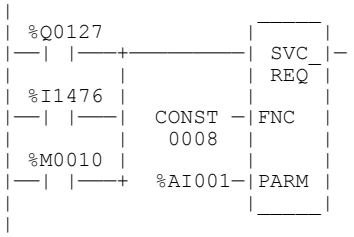
Внимание

Убедитесь, что сброс сторожевого таймера не повлияет на работу процесса управления.

Даная функция не имеет блока параметров, однако для ведения записи программным обеспечением необходимо назначить ячейку памяти для входа PARM. Можете назначить любую ячейку, использоваться она не будет.

Пример

Когда разрешающий выходной контакт %Q0127 или входной контакт %I1476, или внутренняя обмотка %M0010 установлена в 1, сторожевой таймер будет сброшен.



SVCREQ #9: Считать значение времени цикла от начала цикла

Используйте сервисный запрос #9 для получения значения времени в миллисекундах от начала цикла ЦП. Эти данные имеют формат 16-ти битных слов.

Примечание

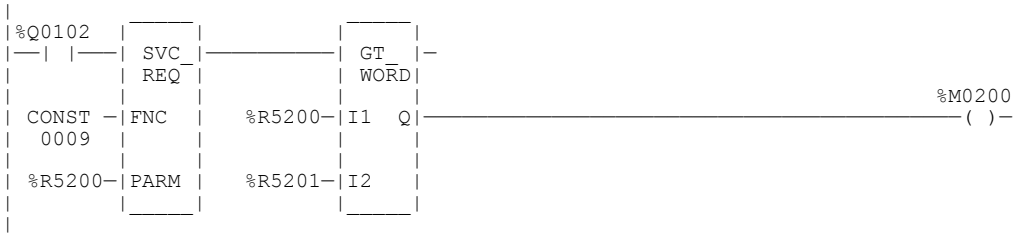
Сервисный запрос #9 поддерживается только ЦП Series 90-30, начиная с версии 8.0.

Блок параметров состоит только из выходного блока параметров; его длина одно слово.

Значение времени с начала цикла ЦП	address

Пример

Время, прошедшее с начала цикла всегда считываться в ячейку %R5200. если оно превышает значение, распложенное в ячейке %R5201, то внутренняя обмотка %M0200 устанавливается в 1.



SVCREQ #10: Считать имя папки

Используйте сервисный запрос #10 для получения имени текущей исполнительной папки.

Примечание

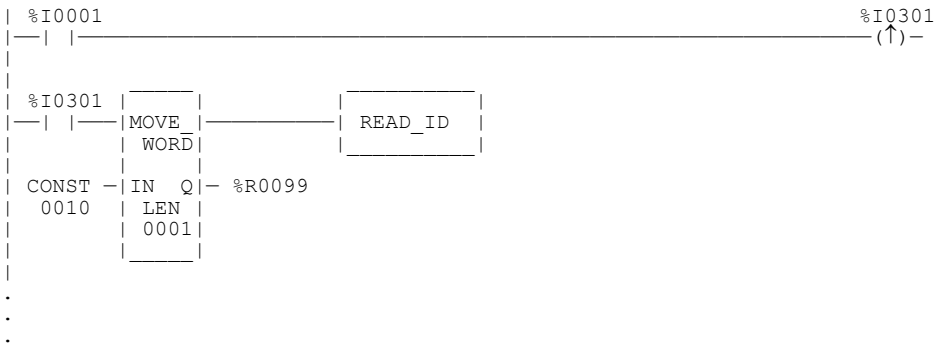
Сервисный запрос #10 поддерживается только ЦП Series 90-30, начиная с версии 8.0.

Выходной блок параметров имеет длину 4 слова. Он возвращает 8 ASCII символов. Последний символ – нулевой (00h). Если имя программы меньше восьми символов, то остающиеся символы заменяются нулевыми.

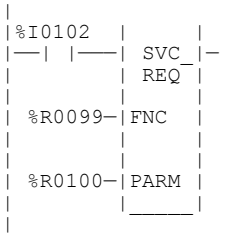
Младший байт	Старший байт	
Символ 1	Символ 2	address
Символ 3	Символ 4	address + 1
Символ 5	Символ 6	address + 2
Символ 7	00	address + 3

Пример

На следующем примере, когда входной разрешающий контакт %I0301 установлен в 0, регистр с адресом %R0099 загружается со значением 10, которое является функциональным кодом для функции считывания имени папки. В действительности, для получения имени папки вызывается программный блок READ_ID. Блок параметров данной функции расположен по адресу %R0100. Блок READ_ID используется также и в следующем примере.



Программный блок READ_ID



SVCREQ #11: Считать идентификатор ПЛК

Используйте сервисный запрос #11 для получения идентификатора ПЛК Series 90 выполняющего программу.

Примечание

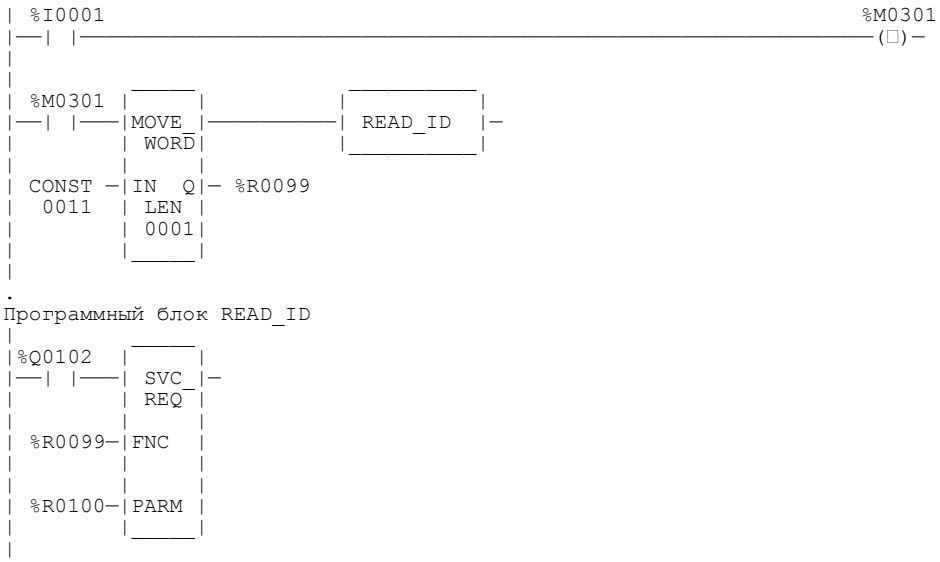
Сервисный запрос #10 поддерживается только ЦП Series 90-30, начиная с версии 8.0.

Выходной блок параметров имеет длину 4 слова. Он возвращает 8 ASCII символов. Последний символ – нулевой (00h). Если идентификатор меньше семи символов, то остающиеся символы заменяются нулевыми.

Младший байт	Старший байт	
Символ 1	Символ 2	address
Символ 3	Символ 4	address + 1
Символ 5	Символ 6	address + 2
Символ 7	00	address + 3

Пример

На следующем примере, когда входной разрешающий контакт %I0001 установлен в 0, регистр с адресом %R0099 загружается со значением 11, которое является функциональным кодом для функции считывания имени ПЛК. В действительности, для получения имени ПЛК вызывается программный блок READ_ID. Блок параметров данной функции расположен по адресу %R0100. Данный код полностью совпадает с предыдущим, за исключением использующихся адресов и номеров.



SVCREQ #12: Считывание статуса работы ПЛК

Используйте сервисный запрос #12 для получения текущего статуса работы ЦП ПЛК.

Примечание

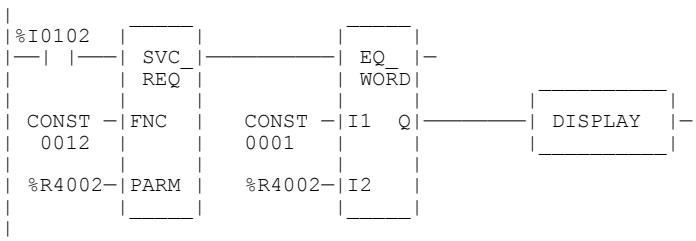
Сервисный запрос #11 поддерживается только ЦП Series 90-30, начиная с версии 8.0.

Блок параметров состоит только из выходного блока параметров; его длина одно слово.

1 = запущен/выключен	address
2 = запущен/включен	

Пример

Значение статуса работы ЦП всегда считывается в ячейку %R4002. Если текущим статусом является статус запущен/выключен, то функция CALL вызывает программный блок DISPLAY.



SVCREQ #13: Выключение ПЛК

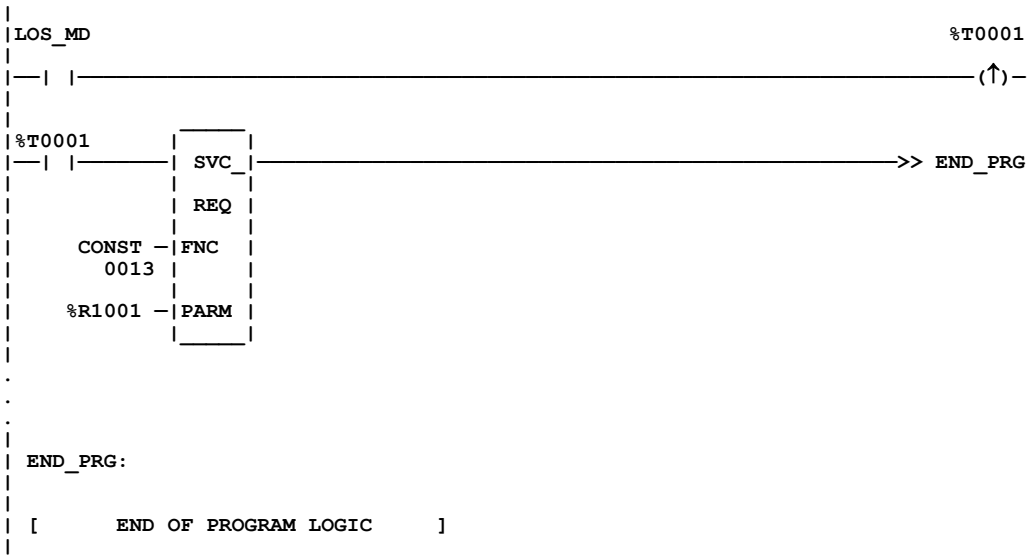
Используйте сервисный запрос 13 для остановки ПЛК ***в конце следующего цикла.*** Значения всех выходов будут переведены в значения, назначенные по умолчанию, к началу следующего цикла. Информационные сбои заносятся с таблицу сбоев ПЛК, заметьте, что был выполнен функциональный блок “SHUT DOWN PLC”.

Данная функция не имеет блока параметров.

Пример

На следующем примере показано, что если произошел сбой «потеря модуля B/B», то выполняется сервисный запрос #13. Так как блок параметров не используется, вход PARM не используется; однако, необходимо сделать запись для входа PARM в программном обеспечении.

Этот пример использует функцию JUMP для перемещения к концу программы, чтобы ускорить выключение ПЛК, в случае, если функция выключения ПЛК выполнилась успешно. Пара функций JUMP и LABEL необходимы, так как переключение в режим **STOP** происходит только после завершения цикла, даже если функция выключение уже выполнилась.



Примечание

Чтобы удостоверится, что контакт %S0002 LST_SCN сработал правильно, ПЛК выполняет один дополнительный цикл, после цикла, в котором был выполнен сервисный запрос 13.

SVCREQ #14: Очищение таблиц сбоев

Используйте сервисный запрос #14 для удаления записей либо из таблицы сбоев ПЛК либо из таблицы сбоев В/В. Выход функции будет находится в положении включено, пока вводятся числа 1 или 0.

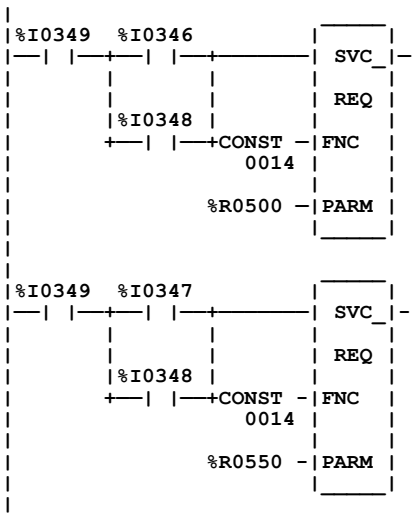
Для данной функции блок параметров имеет длину в одно слово. Данный блок используется только в качестве входного блока.

0 = очистить таблицу сбоев ПЛК. 1 = очистить таблицу сбоев В/В.	address

Пример

Когда контакт %I0346 установлен в 1, и контакт %I0349 установлен в 1, таблица сбоев ПЛК очищается. Когда контакт %I0347 установлен в 1 и контакт %I0349 установлен в 1, очищается таблица сбоев В/В. Когда контакт %I0348 установлен в 1 и контакт %I0349 установлен в 1, очищаются обе таблицы сбоев.

Блок параметров для таблицы сбоев ПЛК расположен по адресу %R0500, а для таблицы сбоев В/В – по адресу %R0550. Оба блока создаются в любом месте программы.



SVCREQ #15: Считать последнюю запись в таблице сбоев

Используйте сервисный запрос #15 для считывания последней записи либо из таблицы сбоев ПЛК либо из таблицы сбоев В/В. Выход функции будет находиться в положении включено, пока вводятся числа 1 или 0, или пока таблица сбоев не пуста (за дополнительной информацией относительно записей в таблицах сбоев обратитесь к части 3 данного руководства «Сбои и их исправление»).

Блок параметров для данной функции имеет длину 22 слова. Входной блок параметров имеет следующий формат:

0 = Read PLC fault table. 1 = Read I/O fault table.	address

Формат выходного блока параметров зависит от типа таблицы, из которой происходит считывание.

Выходной блок для таблицы сбоев ПЛК

Младший байт	Старший байт	
0		
Длинный/короткий		address + 1
резерв		address + 2
Адрес сбоя		address + 3
		address + 4
Группа и действие сбоя		address + 5
Код сбоя		address + 6
		address + 7
		address + 8
		address + 9
		address + 10
		address + 11
Специальные данные сбоя		address + 12
		address + 13
		address + 14
		address + 15
		address + 16
		address + 17
		address + 18
Временная отметка		address + 19
		address + 20
		address + 21

Выходной блок для таблицы сбоев В/В

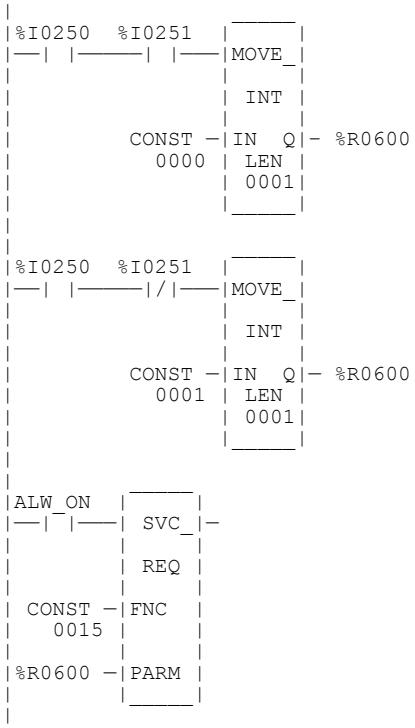
Младший байт	Старший байт
1	
Длинный/короткий	
Начальный адрес	
Адрес сбоя	
Группа и действие сбоя	
Категория сбоя	Тип сбоя
Описание сбоя	
Специальные данные сбоя	
Временная отметка	

В первом байте слова с адресом Address+1 индикатор «длинный/короткий» определяет объем специфических данных в записи о сбое. Он может быть следующим:

Таблица сбоев ПЛК: 00 = -8 bytes (короткий)
01 = 24 bytes (длинный)
Таблица сбоев B/B: 02 = —5 bytes (короткий)
03 = 21 bytes (длинный)

Пример 1

Когда контакт %I0251 установлен в 1 и контакт %I0250 установлен в 1, последняя запись в таблице сбоев ПЛК считывается в блок параметров. Когда контакт %I0251 установлен в 0 и контакт %I0250 установлен в 1, считывается последняя запись из таблицы сбоев B/B. Блок параметров располагается по адресу %R0600.



Пример 2

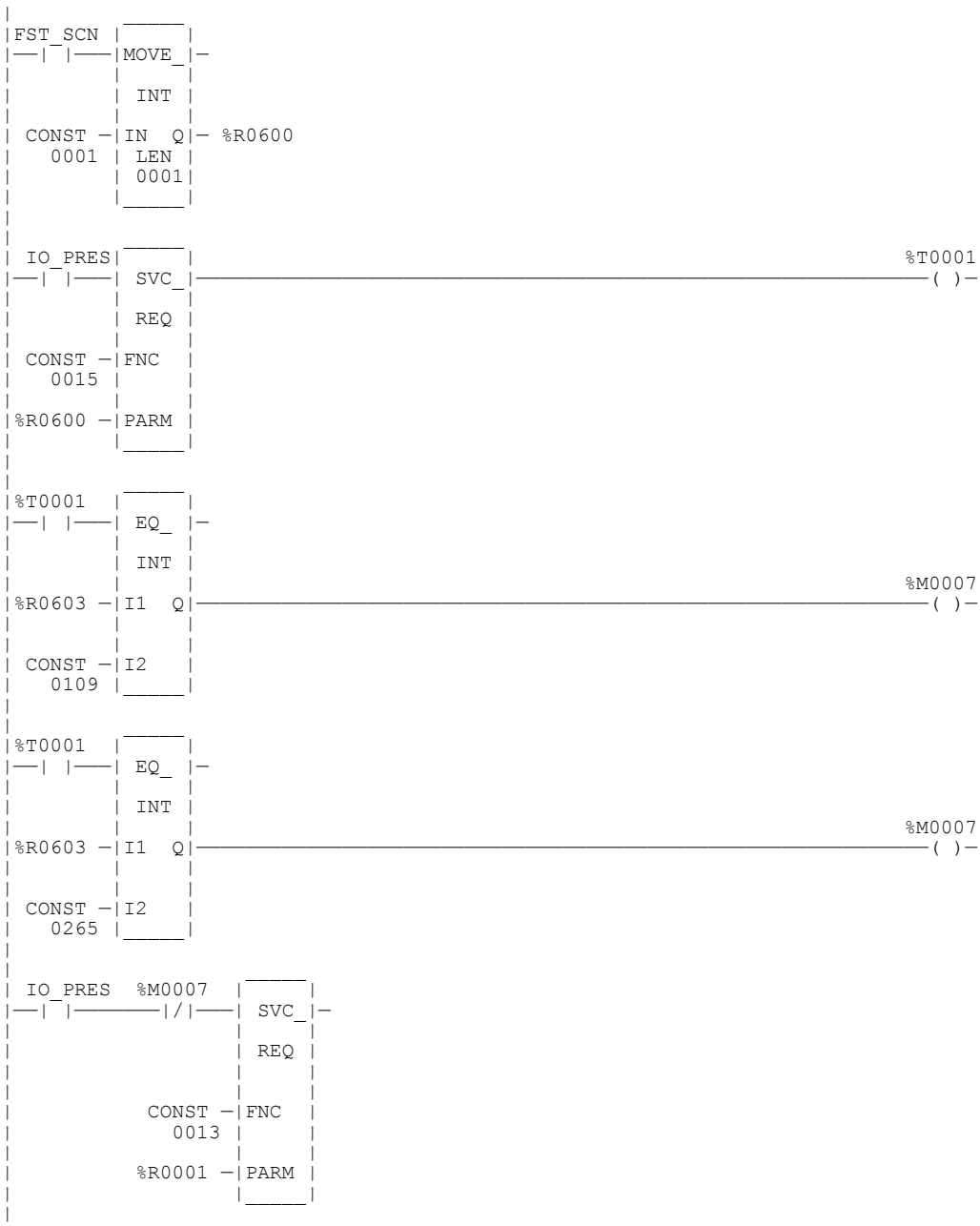
Как показано на следующем примере, ПЛК выключится, если произойдет сбой в модуле В/В, кроме случая, когда этот сбой происходит в модуле находящихся в 9-х слотах главной и первой стоек. Если сбой происходит в этих двух модулях, то ПЛК продолжает работать. Параметр для «типа таблицы» устанавливается во время первого цикла. Контакт IO_PRES, когда он установлен в 1, показывает, что таблица В/В содержит запись. ЦП ПЛК устанавливает, в цикле, нормально разомкнутый контакт в 1, после помещения записи в таблицу сбоев. Если запись сбоя помещается в таблицу 2 цикла подряд, то контакт устанавливается в 1 для двух последовательных циклов.

Пример использует блок параметров, расположенных по адресу %R0600. после выполнения сервисного запроса четвертое, пятое и шестое слова блока параметров содержат адрес модуля, который вызвал сбой:

1		%R0600
Длинный/короткий		%R0601
Начальный адрес		%R0602
Номер стойки	Номер слота	%R0603
Номер шины В/В	Адрес шины	%R0604
Адрес канала		%R0605

Данные сбоя

В программе, блок EQ_INT сравнивает адреса стойки и слота в таблице с 16-тиричной константой. Внутренняя обмотка %M0007 устанавливается в 1, когда стойка/слот, где произошел сбой при появлении определенных условий. Если обмотка %M0007 установлена в 1, то ее нормально замкнутый контакт выключается, препятствуя выключению ПЛК. И наоборот, если %M0007 установлен в 0, потому что сбой произошел в другом модуле, замкнутый контакт включается и происходит выключение ПЛК.


FST_SCN
MOVE_
INT
CONST
0001
IN
Q
%R0600
LEN
0001
IO_PRES
SVC_
%T0001
REQ
FNC
0015
PARM
%R0600
%T0001
EQ_
INT
%R0603
I1
Q
%M0007
CONST
0109
I2
%T0001
EQ_
INT
%R0603
I1
Q
%M0007
CONST
0265
I2
IO_PRES
%M0007
SVC_
REQ
FNC
CONST
0013
PARM
%R0001

SVCREQ #16: Считать значение таймера времени работы

Используйте сервисный запрос #16 для считывания значения времени работы ПЛК. Этот таймер отсчитывает в секундах время работы ПЛК с момента включения. Максимальное значение, которого достигает этот таймер – 100 лет.

Данная функция имеет только выходной блок параметров, имеющий длину 3 слова.

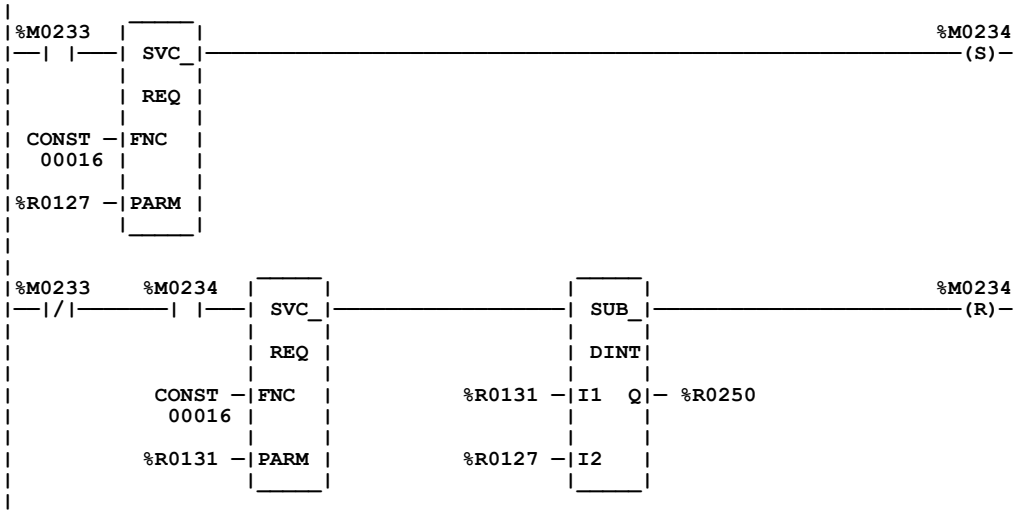
Секунд с момента включения (младший разряд)	address
Секунд с момента включения (старший разряд)	address + 1
100 микросекундные метки	address + 2

В первых двух словах содержится прошедшее время в секундах. Последнее слово – это количество 100 микросекундных меток содержащихся в текущей секунде.

Пример

Когда внутренняя обмотка %M0233 установлена в 1, считывается значение прошедшего времени и, обмотка %M0234 устанавливается в 1. Когда она устанавливается в 0, значение считывается снова. Затем, вычисляется разность между значениями, и результат записывается в регистр %R0250.

Блок параметров для первого считывания располагается в регистре %R0127, а для второго в %R0131. При вычислении не учитывается количество 100-микросекундных меток и то условие, что тип данных DINT в действительности знаковое число. Производящиеся вычисление считается корректным, пока время с момента включения не превысит величины в 50 лет.



SVCREQ #18: Считать статус подстановок B/B

Используйте сервисный запрос #18 для получения текущего статуса подстановок в ЦП.

Примечание

Данная возможность реализована только на ЦП модели 331 и старше.

Для данной функции блок параметров имеет длину в 1 слово и является только выходным блоком.

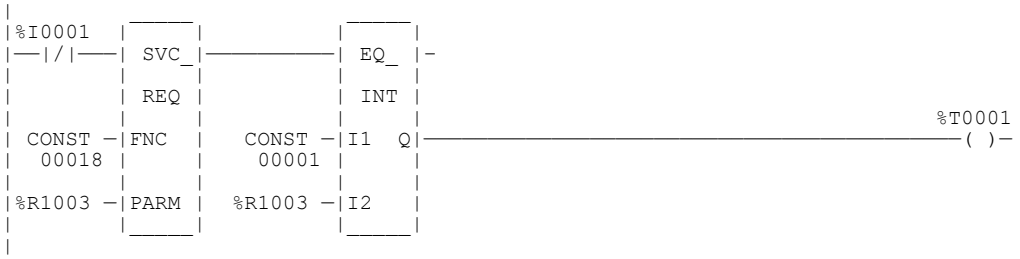
0 = подстановки не установлены. 1 = подстановки установлены.	address

Примечание

Сервисный запрос #18 сообщает только о подстановках в памяти %I и %Q.

Пример

На следующем примере показано, что статус подстановок всегда записывается в ячейку %R1003. Если подстановки существуют, то временная ячейка %T0001 устанавливается в 1.



SVCREQ #23: Считать основную контрольную сумму

Используйте сервисный запрос #23 для считывания основных контрольных сумм для программы и для конфигурации. Выход данной функции всегда устанавливается в 1, если она включена, и выходной блок данных (см. ниже) располагается в памяти, начиная с ячейки, адрес которой указан на входе PARM данной функции.

Когда **РЕЖИМ ЗАПИСИ** запущен, контрольные суммы программы могут быть не верными, пока не завершиться процесс записи. Поэтому, два байта-флага, расположенные в начале блока параметров, указывают на верную или не верную контрольную сумму.

Для данной функции выходной блок параметров имеет длину 12 слов и имеет следующий вид:

Основная контрольная сумма программы верна (0 = не верна, 1 = верна)	address
Основная контрольная сумма конфигурации верна (0 = не верна, 1 = верна)	address + 1
Число программных блоков (включая_MAIN)	address + 2
Размер программы в байтах (тип данных DWORD)	address + 3
Добавочная контрольная сумма программы	address + 5
CRC контрольная сумма программы (тип данных DWORD)	address + 6
Размер конфигурационных данных в байтах	address + 8
Добавочная контрольная сумма конфигурации	address + 9
CRC контрольная сумма гконфигурации (тип данных DWORD)	address + 10

Пример

На следующем примере, когда контакт %I0251 установлен в 1, информация о контрольной сумме помещается в блок параметров и выходная обмотка (%Q0001) устанавливается в 1. Блок параметров находится в регистре %R0050.

SVCREQ #26/30: Опрос системы B/B

Используйте сервисный запрос #26 (или #30 – они идентичны, т.е. вы можете использовать любой номер для получения одного и того же результата) для опроса присутствующих в стойке модулей и сравнения их с описанными в конфигурации, генерации сбоев добавления, потери и отсутствия модуля, как будто конфигурация была запомнена. Эта функция генерирует сбои в обоих таблицах (ПЛК и B/B) в зависимости от типа сбоя.

Эта функция не имеет блока параметров и ее выход всегда активен.

Примечание

Время выполнения данной функции зависит от количества существующих сбоев. Поэтому, чем больше модулей вызвало сбой, тем больше будет время выполнения данной функции

Пример

Когда контакт %I0251 установлен в 1, происходит опрос и сравнение с конфигурацией присутствующих модулей. Обмотка %Q0001 устанавливается в 1 после выполнения функции.

Примечание

Данный сервисный запрос не доступен на ПЛК Micro.

SVCREQ #29: Считать значение таймера времени простоя

Используйте сервисный запрос #29 для определения времени прошедшего с момента последнего выключения до момента нового включения. Выход данной функции всегда устанавливается в 1, если она включена, и выходной блок данных (см. ниже) располагается в памяти, начиная с ячейки, адрес которой указан на входе PARM данной функции.

Примечание

Данная возможность реализована только на ЦП модели 331 и старше.

Для данной функции блок параметров имеет длину в 3 слова и является только выходным блоком.

Секунд с момента выключения (младший разряд)	address
Секунд с момента выключения (старший разряд)	address + 1
100 микросекундные метки	address + 2

Первые два слова содержат прошедшее с момента выключения время в секундах. В последнем слове содержится тоже время только в 100 микросекундных метках (которое всегда равно 0). Всякий раз, когда ПЛК не может правильно подсчитать значение прошедшего времени, оно устанавливается в 0. Это случается, когда ПЛК запускается вместе с нажатой кнопкой CLR M/T на HHP. Также, это происходит, если значение сторожевого таймера истечет до выключения.

Пример

На следующем примере, когда контакт %I0251 установлен в 1, значение прошедшего времени помещается в блок параметров, и выходная обмотка (%Q0001) устанавливается в 1. Блок параметров находится в регистре %R0050.

SVCREQ #45: Пропустить следующее сканирование B/B

(Приостановить B/B). Используйте сервисный запрос #45 для пропуска следующего сканирования B/B. Любые изменения в таблице выходных ссылок в течении цикла, в котором выполняется данный запрос, не будут влиять на физическое состояние каналов выхода. Любые изменения в канале входа не будут влиять на содержание таблицы входящих ссылок, в течение цикла следующего за тем, в котором выполнялся данный запрос.

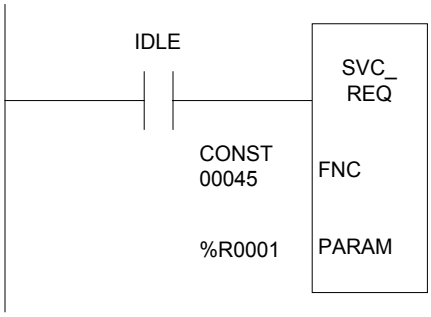
Данный запрос не имеет блока параметров.

Примечание

Сервисный запрос #45 не затрагивает функциональный блок DOIO. Он продолжает обновлять B/B, когда используется в той же программе, что и сервисный запрос.

Пример

В этом примере, когда контакт IDLE инициализирован, последующее сканирование B/B пропускается.



SVCREQ #46: Получить доступ к статусу системной шины

Используйте сервисный запрос #46 для:

1. Считывания слова данных особого статуса с одного из нескольких интеллектуальных модулей.
2. Записи слова данных особого статуса в один из нескольких интеллектуальных модулей.
3. Считывания/записи: Считывания слова данных особого статуса с одного из нескольких интеллектуальных модулей и записи значений в диапазоне от 0 до 15 в модуль.

Примечание

Данный сервисный запрос доступен только для тех модулей, которые его поддерживают. Сегодня существует только один модуль, который поддерживает эту функцию – это DSM (Digital Servo Module) версии 312, который не был доступен во время написания данного руководства. Однако, вскоре он будет доступен.

Функциональный блок COMM_REQ или DOIO не будет работать с интеллектуальным модулем во время цикла ЦП, во время которого выполняется также запись данных, так как записываемые данные могут быть потеряны.

Две функции, которые осуществляют запись данных в модуль (функция Запись и Считывание/Запись) не будут выполняться одновременно, так как данные могут быть потеряны.

Как показано ниже, данный сервисный запрос имеет переменную длину. Первое слово блока параметров определяет, какая функция будет использоваться и имеет следующий вид:

1 = считать особые данные 2 = записать особые данные 3 = считать/записать особые данные	address

Считать статус особых данных (функция #1)

Данная функция считывает статус особых данных с каждого модуля, определяемого списком блока параметров, и помещает значение статуса в блок параметров. Блоку параметров необходимо (N+4) ячеек памяти, где N число модулей с которых будет происходить считывание.

Используйте приведенную ниже таблицу для определения выходных значений:

Таблица 12-5. Выходные значения для функции считывания статуса особых данных

Расположение	Поле	Значение
Address	Тип функции	1 = считать статус особых данных
Address + 1	Код ошибки	Здесь располагается код ошибки, в случае если модули отсутствуют, непригодны или не работают. За деталями обращайтесь к станице 12-71 “Коды ошибок”.
Address + 2	Ошибки в стойке и слоте	Номер стойки и слота, в котором произошла ошибка
Address + 3	Первая стойка и слот	Номер стойки и слота (шестнадцатеричная форма записи; RRSS, где RR номер стойки, а SS номер слота) первого модуля, с которого будут считываться данные.
Address + 4	Считать данные с первого модуля	Данные, считанные с первого модуля, помещаются здесь.
Address + 5	Вторая стойка и слот	Номер стойки и слота (шестнадцатеричная форма записи; RRSS, где RR номер стойки, а SS номер слота) второго модуля, с которого будут считываться данные.
Address + 6	Считать данные со второго модуля	Данные, считанные со второго модуля, помещаются здесь.
Address + (i * 2) + 1	i-ая стойка и слот	Номер стойки и слота (шестнадцатеричная форма записи; RRSS, где RR номер стойки, а SS номер слота) i-ого модуля, с которого будут считываться данные.
Address + (I * 2) + 2	Считать данные с i-ого модуля	Данные, считанные с i-ого модуля, помещаются здесь.
Address + (N * 2) + 1	Последняя стойка и слот	Номер стойки и слота (шестнадцатеричная форма записи; RRSS, где RR номер стойки, а SS номер слота) последнего модуля, с которого будут считываться данные.
Address + (N * 2) + 2	Считать данные с последнего модуля	Данные, считанные с последнего модуля, помещаются здесь.
Address + (N * 2) + 3	Индикатор конца списка	Нуль в данном слове указывает на конец списка модулей

Запись данных (функция #2)

Данная функция производит запись значения, в пределах от 0 до 15, из блока параметров в один или несколько модулей, определяемых списком, находящимся в блоке параметров. Блоку параметров необходимо (N+4) ячеек памяти, где N число модулей с которых будет происходить считывание.

Таблица 12-6. Выходные значения для функции записи данных

Расположение	Поле	Значение
Address	Тип функции	2 = запись данных
Address + 1	Код ошибки	Здесь располагается код ошибки, в случае если модули отсутствуют, непригодны или не работают. Код ошибки не устанавливается, если функция была выполнена, а какой-либо модуль не получил данные. За деталями обращайтесь к станице 12-71 “Коды ошибок”.
Address + 2	Ошибки в стойке и слоте	Номер стойки и слота, в котором произошла ошибка
Address + 3	Первая стойка и слот	Номер стойки и слота (шестнадцатеричная форма записи; RRSS, где RR номер стойки, а SS номер слота) первого модуля, в который будет произведена запись данных.
Address + 4	Запись данных в первый модуль	Эти данные будут записаны в первый модуль
Address + 5	Вторая стойка и слот	Номер стойки и слота (шестнадцатеричная форма записи; RRSS, где RR номер стойки, а SS номер слота) второго модуля, в который будет произведена запись данных.
Address + 6	Считать данные со второго модуля	Эти данные будут записаны во второй модуль
Address + (i * 2) + 1	i-ая стойка и слот	Номер стойки и слота (шестнадцатеричная форма записи; RRSS, где RR номер стойки, а SS номер слота) i-ого модуля, в который будет произведена запись данных.
Address + (i * 2) + 2	Считать данные с i-ого модуля	Эти данные будут записаны в i-ый модуль
Address + (N * 2) + 1	Последняя стойка и слот	Номер стойки и слота (шестнадцатеричная форма записи; RRSS, где RR номер стойки, а SS номер слота) последнего модуля, в который будет произведена запись данных.
Address + (N * 2) + 2	Считать данные с последнего модуля	Эти данные будут записаны в последний модуль
Address + (N * 2) + 3	Индикатор конца списка	Нуль в данном слове указывает на конец списка модулей

Считывание/Запись данных (Функция #3)

Функция считывания/записи считывает статус особых данных с модуля определенного в блоке параметров, затем производит запись значения в диапазоне от 0 до 15 из блока параметров в этот модуль. Этот процесс повторяется для каждого модуля представленного в списке блока параметров. Блоку параметров необходимо (N*3) + 3 ячеек памяти, где N число модулей.

Таблица 12-7. Выходные значения для функции считывания/записи

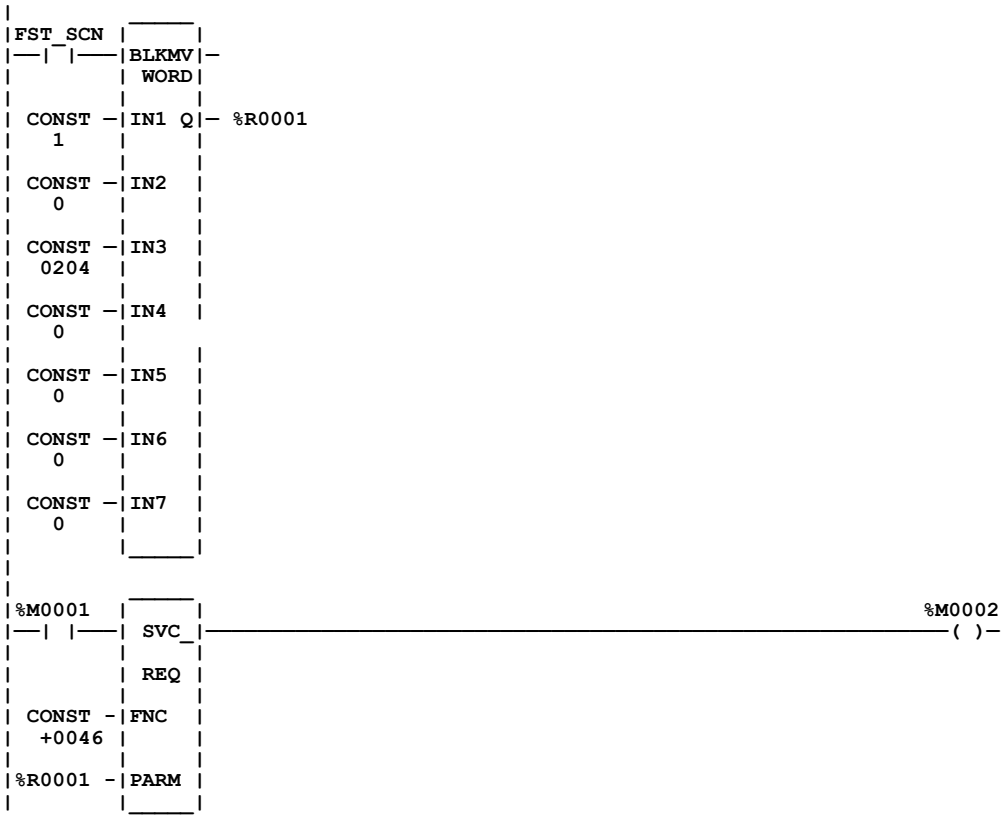
Расположение	Поле	Значение
Address	Тип функции	3 = считывание/запись
Address + 1	Код ошибки	Здесь располагается код ошибки, в случае если модули отсутствуют, непригодны или не работают. Код ошибки не устанавливается, если функция была выполнена, а какой-либо модуль не получил данные. За деталями обращайтесь к станице 12-71 “Коды ошибок”.
Address + 2	Ошибки в стойке и слоте	Номер стойки и слота, в котором произошла ошибка
Address + 3	Первая стойка и слот	Номер стойки и слота (16-тиричная форма записи; RRSS, где RR номер стойки, а SS номер слота) первого модуля, в котором будут изменены данные.
Address + 4	Считать данные с первого модуля	Данные, считанные с первого модуля, помещаются здесь.
Address + 5	Запись данных в первый модуль	Эти данные будут записаны в первый модуль
Address + 6	Вторая стойка и слот	Номер стойки и слота (шестнадцатеричная форма записи; RRSS, где RR номер стойки, а SS номер слота) второго модуля, в котором будут изменены данные.
Address + 7	Считать данные со второго модуля	Данные, считанные со второго модуля, помещаются здесь.
Address + 8	Запись данных во второй модуль	Эти данные будут записаны во второй модуль
Address + ((i-1) * 3) + 3	i-ая стойка и слот	Номер стойки и слота (шестнадцатеричная форма записи; RRSS, где RR номер стойки, а SS номер слота) i-ого модуля, в котором будут изменены данные.
Address + ((i-1) * 3) + 4	Считать данные с i-ого модуля	Данные, считанные с i-ого модуля, помещаются здесь.
Address + ((i-1) * 3) + 5	Запись данных в i-ый модуль	Эти данные будут записаны в i-ый модуль
Address + ((N-1) * 3) + 3	Последняя стойка и слот	Номер стойки и слота (шестнадцатеричная форма записи; RRSS, где RR номер стойки, а SS номер слота) последнего модуля, в котором будут изменены данные.
Address + ((N-1) * 3) + 4	Считать данные с последнего модуля	Данные, считанные с последнего модуля, помещаются здесь.
Address + ((N-1) * 3) + 5	Запись данных в последний модуль	Эти данные будут записаны в последний модуль
Address + (N * 3) + 3	Индикатор конца списка	Нуль в данном слове указывает на конец списка модулей

Коды ошибок

Значение	Описание
1	Успех – функция была выполнена успешно.
-1	Модуль не находится в соответствующем слоте
-2	Неправильный модуль – модуль, расположенный в слоте не является интеллектуальным или не поддерживает данную функцию.
-3	Модуль неисправен – модуль, расположенный в слоте не взаимодействует корректно с ЦП.
-4	Ошибки четности при считывании данных – ошибки четности происходят при считывании данных со стоек расширения или с удаленных стоек.
-5	В главном блоке описана неверная функция.

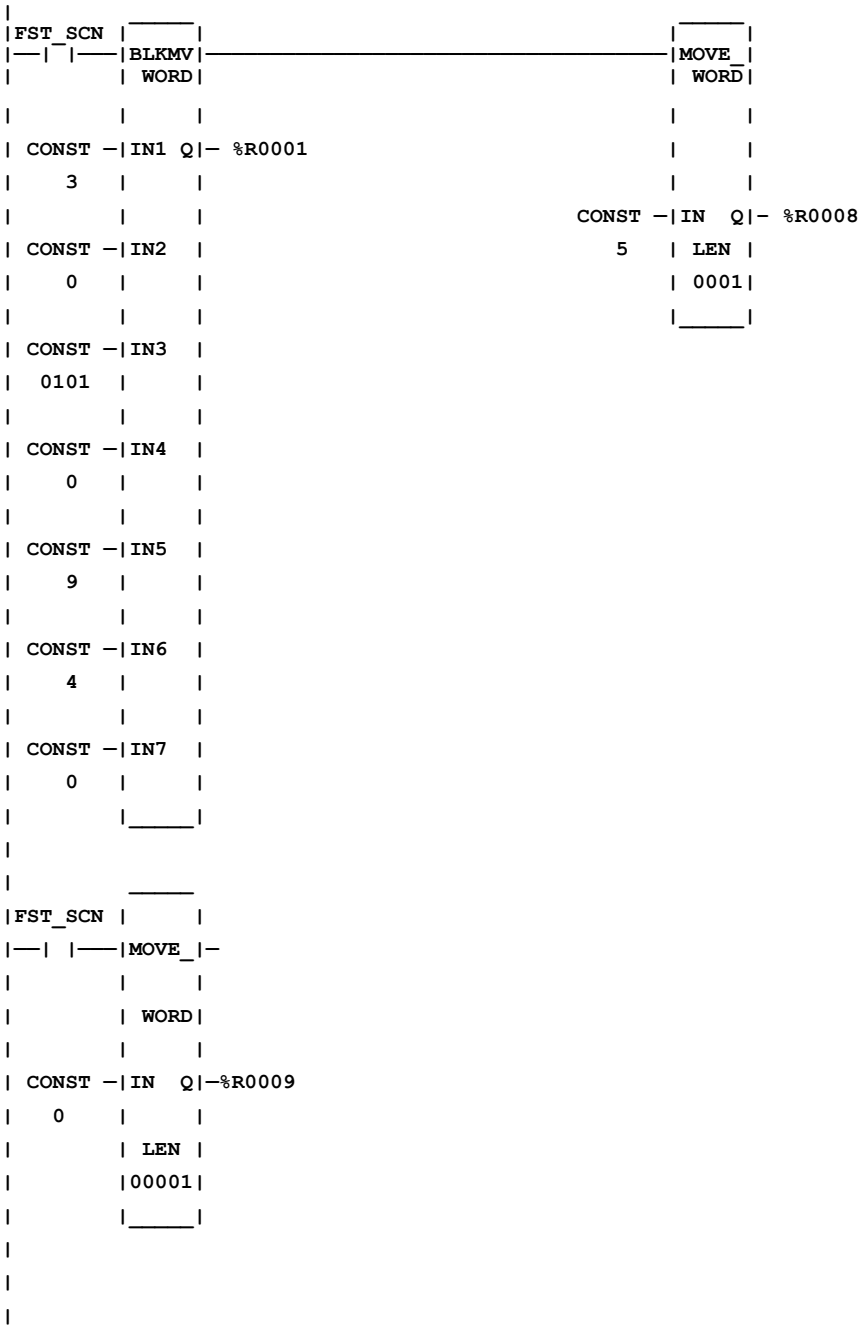
Пример 1

Следующий пример демонстрирует считывание с одного модуля, расположенного во второй стойке и в 4-ом слоте. Входы IN4 и IN5 должны быть установлены в 0. Входы IN6 и IN7 не используются в данном примере. Если функция будет выполнена удачно, данные поступят в ячейку %R0004.



Пример 2

Этот пример демонстрирует процесс считывания статуса особых данных из модуля расположенного в главной стойке, в слоте 4 и из модуля, расположенного в стойке 1, слот 1. Производится запись числа 5 в первый модуль и числа 9 во второй. Заметьте, что номер слота в списке указывать не обязательно. Данные из модуля расположенного в главной стойке, в слоте 4 будут помещены в ячейку %R0007, а из модуля, расположенного в стойке 1, слот 1 в ячейку %R0004.

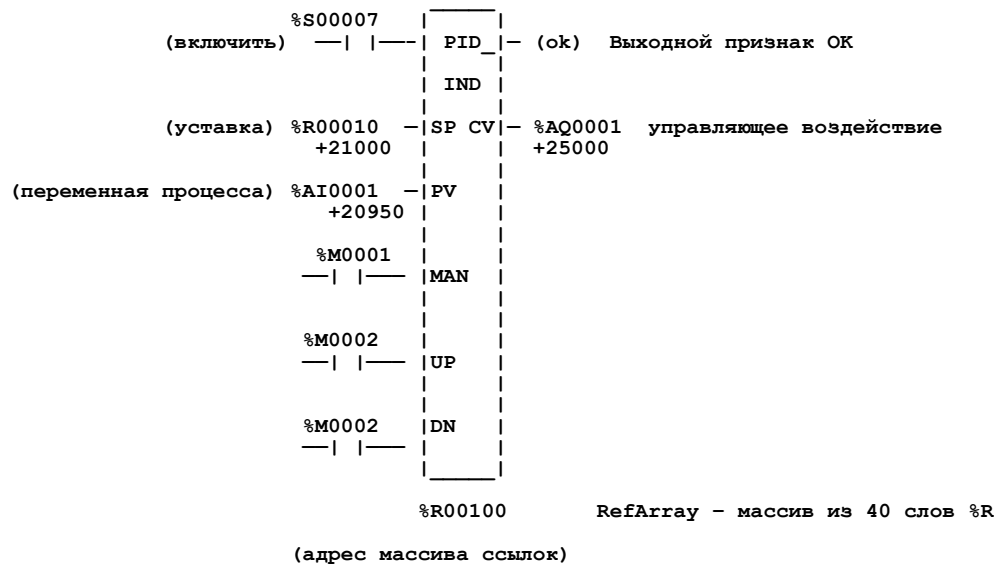



%M0001
SVC_
REQ
CONST
+0046
FNC
%R0001
PARM
%M0001

Функция PID

Функция управления по пропорциональному, интегральному и дифференциальному отклонению (PID) хорошо известна, как алгоритм общего применения для управления замкнутыми следящими системами. Блок функции PID Series 90 сравнивает значение выходной переменной процесса (Process Variable) с желаемым значением уставки (Set Point) и в соответствии с величиной ошибки обновляет переменную управления (Control Variable).

Блок использует коэффициенты усиления контура PID и другие параметры, записанные в массиве из 40 16-разрядных слов (см. стр 12-76) для исполнения PID-алгоритма на заданным интервале времени. Все параметры являются 16-битовыми целочисленными словами для совместимости с 16-битовыми аналоговыми переменными процесса. Это позволяет использовать память %AI для переменных процесса и память %AQ для выходных управляющих переменных. Приведенный ниже пример демонстрирует типичные входы.



Так как входные уставки (Set Point), а также составляющие переменной процесса и переменной управления часто используются, им присвоены аббревиатуры SP, PV и CV. Как масштабируемые 16-разрядные значения, многие параметры должны быть заданы в PV-отсчетах или единицах, либо в CV-отсчетах или единицах. Например, вход SP должен быть масштабирован в том же диапазоне, что и вход PV, так как блок PID вычисляет ошибку путем вычитания этих двух входов. PV и CV-отсчетами могут быть от -32000 до 0, или от 0 до 32000, что соответствует аналоговому масштабу от 0 до 10000 при отображении переменных от 0.00% до 100.00%. PV и CV-отсчеты могут не иметь одинакового масштаба, в этом случае в коэффициент усиления PID включаются масштабные коэффициенты.

Примечание

Функция PID не исполняется чаще одного раза в 10 миллисекунд. Этот факт может изменить желаемый результат, если вы установили исполнение ее на каждом цикле и цикл менее 10 мсек. В этом случае функция PID не будет работать до тех пор, пока через достаточное число циклов не накопится время до 10 мсек, т.е. если время цикла 9 мсек, то функция PID будет исполняться через раз с промежутком 18 мсек.

Параметры

Параметр	Описание
Включить	Когда срабатывает контакт включения, функция PID исполняется.
SP	SP является уставкой контура управления или процесса. Будучи установленным от PV-отсчетов, PID регулирует выходные CV так, что PV соответствует SP (нулевая ошибка).
PV	Вход переменной регулируемого процесса, что является %AI входом.
MAN	При установке в 1 (через контакт), блок PID ставится в MANUAL (Ручной) режим. В противном случае режим автоматический.
UP	При возбуждении с MAN, CV подстраивается на 1 при каждом обращении.*
DN	При возбуждении с MAN, CV подстраивается на 1 при каждом обращении.*
RefArray Address	Адрес является местоположением информации блока управления PID (внутренние параметры и параметры пользователя). Использует 40 %R слов, которые не могут использоваться совместно.
OK	Выход OK активизируется, когда функция исполняется без ошибок. При наличии ошибки находится в состоянии OFF.
CV	CV является выходной управляющей переменной процесса, чаще всего %AQ аналоговым выходом.

* Прирост (UP параметр) или уменьшение (DN параметр) на 1 за каждый доступ к функции PID.

Допустимые типы памяти

Параметр	По-ток	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	пост.	нет
включить	•											
SP		•	•	•	•		•	•	•	•	•	
PV		•	•	•	•		•	•	•	•		
MAN	•											
UP	•											
DN	•											
адрес								•				
OK	•											•
CV		•	•	•	•		•	•	•	•		

- Допустимая ссылка или место, где поток может проходить через данную функцию.

Блок параметров PID

Кроме 2 входных слов и 3 контактов ручного управления, блок PID использует 13 параметров в массиве RefArray. Эти параметры должны быть установлены до вызова блока. Другие параметры используются ПЛК и являются не конфигурируемыми. Приведенный в таблице адрес %Ref является тем же адресом RefArray, что и внизу PID блока. Число после знака плюс (+) есть смещение в массиве. Например, если RefArray начинается в %R100, то %R113 будет содержать команду Manual (Ручной), которая используется для установки переменной управления и интегратора в ручной режим.

Таблица 12-8. Сводка параметров PID

Регистр	Параметр	Единицы младшего бита	Диапазон значений
%Ref+0000	Номер контура	Целое число	От 0 до 255 (только для отображения у пользователя)
%Ref+0001	Алгоритм	Отсутствует; устанавливается и поддерживается ПЛК	Не конфигурируется
%Ref+0002	Период выборки	10 мсек	От 0 (на каждом цикле) до 65535 (10,9 мин). Используйте для ПЛК Series 90-30 значение не менее 10 (см. стр 4-141)
%Ref+0003	Зона нечувствительности +	PV-отсчеты	От 0 до 32000 (неотрицательное)
%Ref+0004	Зона нечувствительности -	PV-отсчет	От -32000 до 0 (неположительное)
%Ref+0005	Пропорциональное усиление	0.01CV% PV%	От 0 до 327.67%%
%Ref+0006	Усиление по производной	0.01 сек	От 0 до 327.67 сек
%Ref+0007	Темп интегрирования	повторов/1000 сек	От 0 до 32.767 повт/сек
%Ref+0008	Смещение CV/выхода	CV-отсчеты	От –32000 до-+32000 (добавляется к выходу интегратора)
%Ref+0009	Фиксатор верхнего уровня	CV-отсчеты	От –32000 до-+32000 (>%Ref+10) ограничение выхода
%Ref+0010	Фиксатор нижнего уровня	CV-отсчеты	От –32000 до-+32000 (<%Ref+9) ограничение выхода
%Ref+0011	Минимальное время нарастания	сек/полный размах	От 0 до -32000 сек для 32000 CV
%Ref+0012	Конфигурационное слово	Использует 5 младших битов	Биты 0-2 для ошибки +/-, полярности на выходе, производной.
%Ref+0013	Команда Manual (Ручной)	CV-отсчеты	Отслеживает CV в автоматическом режиме или устанавливает CV в ручном режиме
%Ref+0014	Управляющее слово	Устанавливается ПЛК, если бит 1 не установлен	Обслуживается ПЛК до тех пор, пока не задано обратное: младший бит устанавливает переопределение, если равен 1 (см. таблицу стр. 4-144)
%Ref+0015	Внутр. SP	Отсутствует; устанавливается и поддерживается ПЛК	Не конфигурируется
%Ref+0016	Внутр. CV	Отсутствует; устанавливается и поддерживается ПЛК	Не конфигурируется

Регистр	Параметр	Единицы младшего бита	Диапазон значений
%Ref+0017	Внутр. PV	Отсутствует; устанавливается и поддерживается ПЛК	Не конфигурируется
%Ref+0018	Выход	Отсутствует; устанавливается и поддерживается ПЛК	Не конфигурируется
%Ref+0019	Хранит составляющую Diff	Отсутствует; устанавливается и поддерживается ПЛК	Не конфигурируется
%Ref+0020 и %Ref+0021	Хранит составляющую Int	Отсутствует; устанавливается и поддерживается ПЛК	Не конфигурируется
%Ref+0022	Хранит составляющую Slew	Отсутствует; устанавливается и поддерживается ПЛК	Не конфигурируется
%Ref+0023	Часы	Отсутствует; устанавливается и поддерживается ПЛК	Не конфигурируется
%Ref+0024			
%Ref+0025	(время последнего исполнения)		
%Ref+0026	Хранит остаток Y	Отсутствует; устанавливается и поддерживается ПЛК	Не конфигурируется
%Ref+0027	Нижний диапазон для SP, PV	PV-отсчеты	От –32000 до +32000 (>%Ref+28) для отображения
%Ref+0028	Верхний диапазон для SP, PV	PV-отсчеты	От –32000 до-+32000 (<%Ref+9) для отображения
%Ref+0029 … %Ref+0034	Зарезервировано для внутреннего использования	Нет	Не конфигурируется
%Ref+0035 … %Ref+0039	Зарезервировано для внутреннего использования	Нет	Не конфигурируется

Массив RefArray должен быть регистрами %R в ПЛК Series 90-30. Обратите внимание, что каждый вызов блока PID должен использовать разные массивы из 40 слов, даже если все 13 параметров пользователя одинаковы, поскольку другие слова массива используются для хранения внутренних данных PID. Убедитесь, что массив не выходит за пределы памяти.

Для конфигурирования пользовательских параметров, выберите функцию PID и нажмите F10 для раскрытия экрана, содержащего параметры пользователя; затем используйте клавиши со стрелками для выбора полей и введите желаемые значения. Вы можете использовать 0 для большинства значений по умолчанию, кроме верхнего фиксатора CV, который должен быть больше нижнего фиксатора CV для нормальной работы блока PID. Имейте в виду, что блок PID не пропускает энергию, если имеется ошибка в параметрах пользователя, поэтому осуществляйте контроль с помощью временной обмотки при вводе данных.

После выбора подходящих значений PID, их нужно задать, как константы функции BLKMOV, чтобы их можно было использовать для перезагрузки установленных по умолчанию параметров PID пользователя при необходимости.

Работа команды PID

Нормальной работой в автоматическом режиме является вызов блока PID на каждом цикле, с потоком энергии к входному контакту Enable (Включить), и при отсутствии потока энергии к входному контакту Manual (Ручной). Блок сравнивает текущее время часов использованного времени ПЛК с последним временем решения PID, хранящемся во внутреннем массиве RefArray. Если разница по времени больше периода выборки, заданного в третьем слове массива RefArray (%Ref+2), то PID-алгоритм исполняется с использованием разницы во времени, после чего время последнего исполнения и выход управляющей переменной обновляются. В автоматическом режиме выходная переменная управления размещается в параметре команды Manual (Ручной) %Ref+13.

Если поток энергии направлен сразу к обоим входным контактам Enable (Включить) и Manual (Ручной), блок PID переводится в ручной режим и выходная переменная управления удаляется из параметра команды Ручной %Ref+13. Если поток энергии поступает на какой-либо из входов UP или DN, то слово команды Manual (Ручной) наращивается или уменьшается по одному отсчету CV при каждом исполнении PID. Для ускорения ручного изменения выходной переменной управления можно добавить или вычесть любое значение отсчета CV прямо в слово или из слова команды Manual (Ручной).

Блок PID использует параметры Upper (Верхний) ограничитель (фиксатор) CV и Lower (Нижний) ограничитель CV для ограничения выходного значения переменной CV. Если задан положительный параметр минимального времени нарастания, то эти параметры также используются для ограничения скорости изменения выхода CV. Если либо амплитуда, либо скорость выхода CV оказываются превышены, то сохраненное интегратором значение регулируется так, чтобы оно равнялось пределу. Эта противосбросовая концевая функция (см. стр 12-82), означает, что даже если ошибка пытается заставить CV выйти за пороговое значение на длительный период времени, то выход CV будет уходить с ограничения, как только составляющая ошибки изменит знак.

Такая организация работы, с установкой значения CV в ручном режиме и с сопровождением CV в автоматическом режиме с использованием команды Manual (Ручной), обеспечивает плавный переход от ручного режима в автоматический и наоборот. Значения параметров верхнего и нижнего уровней ограничения и минимального времени нарастания остаются действующими по отношению к CV и в ручном режиме, а внутреннее значение, хранимое интегратором, обновляется. Это означает, что если вы сделали приращения содержимого команды Manual (Ручной) в ручном режиме, то выход CV не будет изменяться быстрее, чем это определено параметром минимального времени нарастания (обращенным), и не выйдет за пределы верхнего и нижнего уровней ограничения.

Примечание

Каждая функция PID не должна вызываться чаще одного раза за цикл.

Следующая таблица содержит более подробные сведения о параметрах, кратко рассмотренных в таблице 12-8. Числа в кавычках после имени каждого параметра означают его смещение в массиве RefArray.

Таблица 12-9. Описание параметров PID

Элемент данных	Описание
Номер контура (00)	Необязательный параметр, служащий для идентификации блока PID. Он является целочисленной величиной без знака, обеспечивающей общую идентификацию в ПЛК посредством номера контура, задаваемого устройством интерфейса оператора. Номер контура отображается под адресом блока, когда логика просматривается программным обеспечением Logicmaster 90-30/20/Micro.
Алгоритм (01)	Целочисленная величина без знака, которая устанавливается ПЛК для идентификации, какой алгоритм будет использоваться функциональным блоком. Алгоритм ISA задается как алгоритм 1, независимый алгоритм идентифицируется как алгоритм 2.
Период выборки (02)	Наименьшее время, измеряемое с шагом 10 мсек между исполнениями PID-алгоритма. Например, для периода выборки 100 мсек, нужно использовать значение 10. Величина UINT может принимать значения до 65535 при периоде выборки 10,9 мин. Если эта величина равна 0, то алгоритм исполняется каждый раз при вызове блока (см. ниже о планировании блока PID). PID алгоритм исполняется только, если текущее время часов использованного времени ПЛК превышает время последнего исполнения PID по крайней мере на период выборки. Помните, что серия 90-30 не использует времена решения, меньшие чем 10 мс (см. примечание на стр 12-74); поэтому циклы с меньшим временем будут пропущены. Данная функция округляет время, прошедшее с последнего исполнения с точностью до100 мкс. Если значение период выборки установлено равным 0, то функция исполняется каждый раз, когда включена, с учетом минимально допустимого интервала 10 мс, рассмотренного ранее.
Зона нечувствительности (+/-) (03/04)	Значение INT, задающее верхний (+) и нижний (-) пределы зоны нечувствительности в CV-отсчетах. Если зона нечувствительности не требуется, это значение должно быть установлено равным 0. Если ошибка PID (SP минус PV) или (PV минус SP) выше отрицательного значения или ниже положительного значения, то в вычислениях PID подставляется нулевое значение ошибки. При ненулевых параметрах значение (+) должно быть больше 0, а значение (-) меньше нуля, иначе блок PID не будет исполняться. Вы должны оставлять эти параметры нулевыми до тех пор, пока усиление контура PID устанавливается или настраивается. После установки вы можете добавить значение зоны нечувствительности, чтобы исключить изменения выхода CV при малых вариациях ошибки, например, для уменьшения механических колебаний.
Пропорциональный коэффициент усиления Кр (05)	Это число INT, называемое в версии ISA усилением контроллера Кс, задает изменение отсчета CV на 100 отсчетов PV в составляющей ошибки. Они отображается как 0.00%% с двумя полагаемыми знаками после децимальной точки. Например, введенное значение Кр 450 будет отображено как 4.50, и будет означать, что выход PID равен Кр∗ошибка/100 или 450∗ошибку/100. Кр обычно устанавливается первым при настройке контура PID.
Усиление по производной – Kd (06)	Это число INT задает изменение отсчета CV при изменении ошибки или PV на один PV-отсчет за каждые 10 мсек. Вводимое как время с младшим значащим битом в 10 мсек, оно отображается в виде 0.00 сек с двумя знаками после запятой. Например, введенное значение 120 будет отображено, как 1.20, и будет означать, что выход PID будет равен Kd∗(приращение ошибки)/(приращение времени) или 120∗4/3 при изменении ошибки на 4 отсчета PV каждые 30 мсек. Kd может использоваться для ускорения медленного отклика контура, однако при этом заметно увеличивается чувствительность к входным шумам PV.

Элемент данных	Описание
Усиление по скорости Ki (07)	Это число INT задает изменение отсчета CV за каждые 10 мсек при постоянном значении ошибки, равном 1 отсчету PV. Оно отображается в виде 0.000 повторов/сек с тремя знаками после запятой. Например, введенное значение 1400 будет отображено как 1.400 повторов/сек и будет означать, что выход PID будет равен Ki* (приращение времени) или 1400*20*50/1000 при ошибке 20 отсчетов PV и времени цикла 50 мсек (период выборки равен 0). Ki обычно устанавливается вторым после Kp.
Смещение CV / Сдвиг выхода (08)	Это число INT, выраженное в отсчетах CV, добавляется к выходу PID перед ограничением по скорости и амплитуде. Оно может использоваться для установки ненулевого CV значения, если используется только пропорциональное усиление, или для прямого управления выходом контура PID от другого контура управления.
Нижний и верхний ограничения CV (09/10)	Это INT значение выраженное в отсчетах CV, которое задает наибольшее и наименьшее значение для CV. Эти значения необходимы, и наибольшее значение должно быть более положительным, чем наименьшее значение, иначе PID блок не будет работать. Они обычно используются для задания пределов, основываясь на физических ограничениях выхода CV. Они также используются для масштабирования гистограммы CV для дисплеев LM90 и ADS. Блок имеет противосбросовую концевую функцию для изменения значения интегратора при достижении уровня фиксации CV.
Минимальное время нарастания (11)	Положительное значение UINT, служащее для задания минимального количества секунд для изменения CV выхода от 0 до 100 % или до 32000 отсчетов CV. Оно обратно пропорционально пределу скорости, с которой может изменяться выход CV. Если оно положительно, то CV не может изменяться быстрее, чем 32000, умноженное на приращение времени и деленное на минимальное время нарастания. Например, если период выборки равен 2.5 сек и минимальное время равно 500 сек, то CV не может изменяться быстрее, чем 32000*2.5/500 или 160 отсчетов CV на PID интервал решения. Как и для фиксаторов CV, имеется противосбросовая концевая функция, которая настраивает интегратор при превышении предела скорости. Если минимальное время равно 0, то порога скорости не имеется. Проверьте, что вы установили нулевое значение минимального времени нарастания при настройке усиления контура PID.
Конфигурационное слово	Младшие 5 бит этого слова используются для изменения трех стандартных установок PID. Другие биты должны быть установлены на 0. Установите младший бит на 1, чтобы изменить стандартную составляющую ошибки PID с нормальной (SP минус PV) на инверсную (PV минус SP), изменив тем самым знак обратной связи. Это используется для реверса управления, когда CV должен уменьшаться при увеличении PV. Установите второй бит на 1, чтобы инвертировать полярность выхода так, чтобы CV стал отрицательным вместо обычной положительной величины. Установите четвертый бит на 1 для изменения действия по производной, перейдя от использования обычного изменения составляющей ошибки к изменению в составляющей в обратной связи PV. Младшие 5 битов конфигурационного слова задаются и детально определяются следующим образом: Бит 0 = Составляющая ошибки. Когда этот бит установлен на 0, составляющая ошибки равна (SP-PV). Когда этот бит установлен на 1, составляющая ошибки равна (PV-SP). Бит 1 = Полярность выхода. Когда этот бит установлен на 0, CV выход представляет результат PID вычисления. Когда этот бит установлен на 1, CV выход равен результату вычисления с обратным знаком. Бит 2 = Дифференцирование PV. Когда этот бит установлен на 0, дифференцирование применяется к составляющей ошибки. Когда этот бит установлен на 1 дифференцирование применяется к PV. Все остальные биты должны быть 0. Бит 3 = Действие зоны нечувствительности. Когда этот бит установлен на 0, зона

<table>
<tr><th>Элемент данных</th><th>Описание</th></tr>
<tr><td></td><td>нечувствительности не выбирается. Если ошибка находится в пределах зоны, она приравнивается к нулю. Если нет, то наличие зоны не учитывается. Когда этот бит установлен на 1, зона нечувствительности выбирается. Если ошибка находится в пределах зоны, то она приравнивается к нулю. Если нет, то значение ошибки уменьшается на величину зоны нечувствительности, точнее на величину ее предела (ошибка = ошибка – предел зоны нечувствительности).

Бит 4 = Противосбросовая концевая функция. Когда этот бит установлен на 0, это действие использует сброс при вычислении. При фиксации выхода, значение накопленного остатка Y (см. стр. 12-82) заменяется значением, необходимым для точного воспроизведения выходного предела. Когда этот бит установлен на 1, это действие замещает значение накопленной составляющей Y значением составляющей Y в начале вычислений. Таким образом, предварительно ограниченная величина Y удерживается в течении всего времени фиксации выхода.

ПРИМЕЧАНИЕ: Бит противосбросовой концевой функции доступен только для ЦП Series 90-30 выпусков 6.50 и более поздних.

Помните, что биты устанавливаются по степеням 2. Например, чтобы установить конфигурационное слово на нулевое значение конфигурации PID по умолчанию, нужно добавить 1, чтобы изменить составляющую ошибки с (SP-PV) на (PV-SP), или добавить 2, чтобы изменить полярность с CV=PID Out на CV=-PID Out, или добавить 4, чтобы изменить Дифференцирующее действие со скорости изменения ошибки на скорость изменения PV.</td></tr>
<tr><td>Команда Manual (Ручной)

(13)</td><td>Это значение INT, которое устанавливается к текущему значению выхода CV, когда блок PID находится в автоматическом режиме. Когда блок переключается в ручной режим, это значение используется для установки выхода CV и внутреннего значения интегратора в пределах, ограничиваемых верхним и нижним уровнями и минимальным временем нарастания.</td></tr>
<tr><td>Управляющее слово

(14)</td><td>Это внутренний параметр, обычно оставляемый нулевым.

Если младший бит переопределения установлен на 1, то это слово и другие внутренние параметры (SP, PV,CV) должны использоваться для удаленной работы блока PID (см. ниже). Это позволяет удаленным устройствам интерфейса оператора, например, компьютеру, перехватить управление у программы ПЛК. Предупреждение: если вы не хотите этого, то используйте управляющее слово, установленное на 0. Если младший бит равен 0, то следующие 4 бита можно прочитать для отслеживания статуса входных контактов PID, пока контакт включения PID получает энергию. Структура дискретных данных с положением первых пяти битов имеют следующий формат:
<table>
<tr><th>Бит:</th><th>Значение слова:</th><th>Функция</th><th>Статус или внешнее действие, если бит переопределения равен 1:</th></tr>
<tr><td>0</td><td>1</td><td>Переопределение</td><td>Если 0, контролируйте контакты блока (ниже). Если 1, установите их извне.</td></tr>
<tr><td>1</td><td>2</td><td>Ручной/Авто</td><td>Если 1, то блок в режиме Manual; другие числа соответствуют режиму Auto.</td></tr>
<tr><td>2</td><td>4</td><td>Enable (Включить)</td><td>Нормально должен быть равен 1; в противном случае блок никогда не вызывается.</td></tr>
<tr><td>3</td><td>8</td><td>UP/Выше</td><td>Если 1 и режим Manual (бит 1) равен 1, то CV получает приращение при каждом исполнении</td></tr>
<tr><td>4</td><td>16</td><td>DN/Ниже</td><td>Если 1 и режим Manual (бит 1) равен 1, то CV получает приращение при каждом исполнении</td></tr>
</table></td></tr>
<tr><td>SP (15)</td><td>(Не конфигурируется - устанавливается и обслуживается ПЛК). Отслеживает вход SP; должен устанавливаться внешне, если бит переопределения = 1.</td></tr>
<tr><td>CV (16)</td><td>(Не конфигурируется - устанавливается и обслуживается ПЛК).

Отслеживает выход CV.</td></tr>
</table>

Элемент данных	Описание
PV (17)	(Не конфигурируется - устанавливается и обслуживается ПЛК). Отслеживает вход PV; должен устанавливаться внешне, если бит переопределения = 1.
Выход (18)	(Не конфигурируется - устанавливается и обслуживается ПЛК). Это знаковая величина, представляющая выход функционального блока перед применением необязательного инвертирования. Если инвертирование не сконфигурировано и бит выходной полярности в управляющем слове установлен на 0, то это значение равно выходу CV. Если инвертирование выбрано и бит выходной полярности установлен на 1, это значение равно выходу CV со знаком минус.
Память для составляющей Diff (19)	Используется внутренне для хранения промежуточного значения. Не пытайтесь делать запись в эту ячейку.
Память для составляющей Int (20/21)	Используется внутренне для хранения промежуточного значения. Не пытайтесь делать запись в эту ячейку.
Память для составляющей Slew (19)	Используется внутренне для хранения промежуточного значения. Не пытайтесь делать запись в эту ячейку.
Часы (23-25)	Используется внутренне для хранения времени (время последнего исполнения PID). Не пытайтесь делать запись в эту ячейку.
Y остаток (26)	Удерживает остаток для масштабирования интегратора для нулевой установившейся ошибки.
Нижний и верхний диапазон (27/28)	Необязательные INT значения, выраженные в отсчетах PV, которые задают наибольшее и наименьшее отображаемые значения для горизонтальной гистограммы SP и PV, задаваемой клавишей ZOOM Logicmaster, и дисплея лицевой панели ADS PID.
Зарезервировано (29-34) и (35-39)	29-34 зарезервированы для внутреннего использования; 35-39 зарезервированы для внешнего использования. Зарезервировано для использования GE Fanuc и не может использоваться с другими целями.

Внутренние параметры в массиве RefArray

Как следует из приведенной выше таблицы 12-9, блок PID считывает 13 параметров пользователя и использует остаток из 40 слов массива RefArray для хранения внутренних параметров PID. Обычно, вам нет необходимости изменять эти значения. Если вы вызываете блок PID в автоматическом режиме после длительной задержки, то вы можете пожелать применить функцию SVC_REQ#16 для загрузки часов использованного времени ПЛК в %Ref+23 для обновления времени последнего использования PID, чтобы избежать изменения шага на интеграторе. Если младший бит переопределения в управляющем слове (%Ref+14) установлен на 1, то следующие 4 бита управляющего слова должны быть установлены для управления входными контактами блока PID (как уже описано в таблице 12-8 на предыдущей странице), и внутренние SP и PV должны быть установлены так, как если вы хотите взять на себя управление блоком PID, отобрав его у многозвенной логики. Словами внутренних параметров являются:

Выбор алгоритма PID (PIDISA или PIDIND) и коэффициентов усиления

Блок PID можно программировать, выбирая либо версию независимых составляющих (PID_IND), либо стандартную ISA (PID_ISA) версию PID-алгоритма. Различие между ними только в задании коэффициентов усиления по интегралу и по производной. Для понимания отличий вам необходимо понять следующее:

Обе версии вычисляют составляющую ошибки как SP-PV, которую можно перевести в инвертированный режим PV-SP, если установить бит ошибки (младший нулевой бит конфигурационного слова %Ref+12) в состояние 1. Инвертированный режим можно использовать, когда вы хотите, чтобы CV-выход изменялся в противоположном по отношению к PV-входу направлении (CV вниз при PV вверх), а не как обычно (CV вверх при PV вверх).

Ошибка = (SP-PV) или (PV-SP), если младший бит конфигурационного слова равен 1

Дифференцирование обычно основано на изменении составляющей ошибки со времени последнего исполнения алгоритма, которая может вызывать большое изменение выхода, если величина SP изменялась. Если это нежелательно, то можно установить третий бит конфигурационного слова на 1 для вычисления производной на основе изменений PV. Величина dt (приращение времени) определяется вычитанием времени последнего исполнения PID для данного блока и использованного времени ПЛК.

dt = Текущее использованное время ПЛК минус использованное время ПЛК при последнем исполнении PID

Производная = (Ошибка-предыдущее значение ошибки)/dt, или (PV минус предыдущее значение PV)/dt, если 3 бит конфигурационного слова равен 1

PID алгоритм независимых составляющих (PID_IND) вычисляет выходное значение как:

Выход PID = Кр ∗ Ошибка + Ki ∗ Ошибка ∗ dt + Kd ∗ Производная + Смещение CV

Стандартный PID алгоритм ISA (PID_ISA) вычисляет выходное значение как:

Выход PID = Кс ∗ (Ошибка + Ошибка ∗ dt/Ti + Производная ∗ Td) + Смещение CV

где Кс является усилением контроллера, а Ti и Td являются интегральной и дифференциальной постоянными времени. Преимуществом стандартной версии является то, что изменение Кс изменяет вклад всех составляющих, что может упростить настройку следящего контура. Если вы имеете коэффициенты усиления, выраженные в терминах Ti и Td, то используйте следующие выражения для преобразования при использовании в качестве входов пользовательских параметров PID:

Кр = Кс Ki = Кс/Ti Kd = Кс/Td

Составляющая смещения CV является аддитивной составляющей, отделенной от компонент PID. Она может понадобиться, если вы используете только пропорциональное усиление и хотите, чтобы CV имел ненулевое значение, когда PV равно SP и ошибка равна нулю. В этом случае устанавливайте смещение CV к желаемому CV значению, когда PV равно SP. Смещение также можно использовать для прямого управления, когда другой контур PID или управляющий алгоритм используется для настройки выхода CV данного PID-контура.

Если используется интегральное усиление Ki, то смещение CV в норме должно быть равно 0, поскольку интегратор действует в качестве автоматического смещения. Начните работу в ручном режиме и используйте слово Manual Command (Команда Ручной) (%Ref+13) для установки интегратора к желаемому значению CV, а затем перейдите в автоматический режим. Этот метод работает и в том случае, когда Ki = 0, за исключением того, что интегратор не будет перестраиваться в зависимости от ошибки при переходе в автоматический режим.

Следующая диаграмма показывает, как работает PID алгоритм:

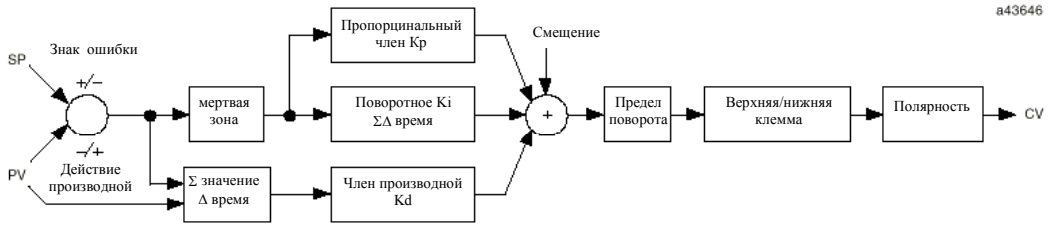


Рисунок 12-4. Алгоритм независимых составляющих (PIDIND)

Алгоритм ISA (PIDISA) аналогичен, за исключением того, что усиление Кр распадается на Ki и Kd так, что интегральное усиление равно Кр∗Ki, а усиление по производной равно Кр∗Kd. Параметры знака ошибки, полярности и дифференцирующего действия устанавливаются битами конфигурационного слова.

Амплитудные и скоростные ограничения CV

Блок не посылает вычисленное значение прямо в CV. Оба PID-алгоритма могут вводить амплитудное ограничение и ограничение по скорости в переменную управления. Максимальная скорость изменения определяется делением максимального 100% значения CV (32000) на минимальное время нарастания, если оно задано большим нуля. Например, если минимальное время нарастания равно 100 сек, то предел скорости составит 320 отсчетов CV в секунду. Если приращение времени решения было равно 50 мсек, то новое значение CV-

значение CV-выхода не может измениться более чем на 320∗50/1000 или на 16 отсчетов CV по отношению к предыдущему значению CV-выхода.

После этого выход CV сравнивается с верхним и нижним уровнями ограничения. Если какой-либо предел превзойден, то выход CV устанавливается равным значению ограничения. Если либо амплитудные, либо скоростные пределы превышены, то внутреннее значение интегратора настраивается так, чтобы соответствовать установленным пределам, чтобы избежать сброса в конце.

В заключение, блок проверяет выходную полярность (второй бит управляющего слова) и изменяет знак выхода, если управляющий бит установлен на 1.

CV = Фиксированному выходу PID или ему же со знаком минус, если установлен бит полярности выхода

Если блок находится в автоматическом режиме, то конечное значение CV помещается в команду Manual %Ref+13. Если блок в ручном режиме, то уравнение PID пропускается, так как CV устанавливается командой Manual, но все амплитудные и скоростные ограничения проверяются. Это означает, что команда Manual не может изменить выход вне установленных амплитудных и скоростных пределов, и выходное значение не может изменяться быстрее, чем это допускается Минимальным временем нарастания.

Период выборки и планирование блока PID

Блок PID является цифровым представлением аналоговой функции управления, поэтому время выборки dt в выходном уравнении PID не является бесконечно малой величиной, доступной для аналогового управления. Большинство управляемых процессов можно аппроксимировать как коэффициент передачи с запаздыванием первого или второго порядка, возможно, характеризуемый чистым временем задержки. Блок PID устанавливает выход CV к значению процесса и использует обратную связь PV для определения ошибки и подстройки следующего значения CV. Ключевым параметром процесса является общая постоянная времени процесса, показывающая, как быстро изменяется PV-отклик при изменении CV. Как показано ниже, общая постоянная времени Tp+Tc для системы первого порядка есть время, необходимое PV для достижения 63% установившегося значения при скачкообразном изменении CV. Блок PID сможет управлять процессом, только если период выборки значительно меньше половины общей постоянной времени. Большие периоды выборки приводят к нестабильной работе.

Период выборки не должен превышать одной десятой общей постоянной времени (в худшем случае одной пятой). Например, если PV достигает 2/3 установившегося значения за 2 сек, то период выборки должен быть менее 0.2 сек или 0.4 сек в худшем случае. С другой стороны, период выборки не должен быть слишком мал, например, менее чем общая постоянная времени, деленная на 1000, иначе составляющая KI∗ошибка∗dt интегратора PID будет округлена до 0. Например, очень медленный процесс, который затрачивает 10 часов или 36000 сек для достижения 63% уровня, должен иметь период выборки в 40 сек и более.

Если процесс не слишком быстрый, обычно нет необходимости использовать нулевой период выборки для исполнения PID-алгоритма при каждом цикле PID. Если несколько контуров PID используются с периодом выборки, большим времени цикла, то возможны значительные изменения времени цикла ПЛК, если несколько контуров заканчивают исполнение алгоритма в одно и то же время. Простым решением является чередование последовательности одного или более единичных битов в массиве нулевыми битами, чтобы разрешить поток энергии к индивидуальным блокам PID.

Определение характеристик процесса

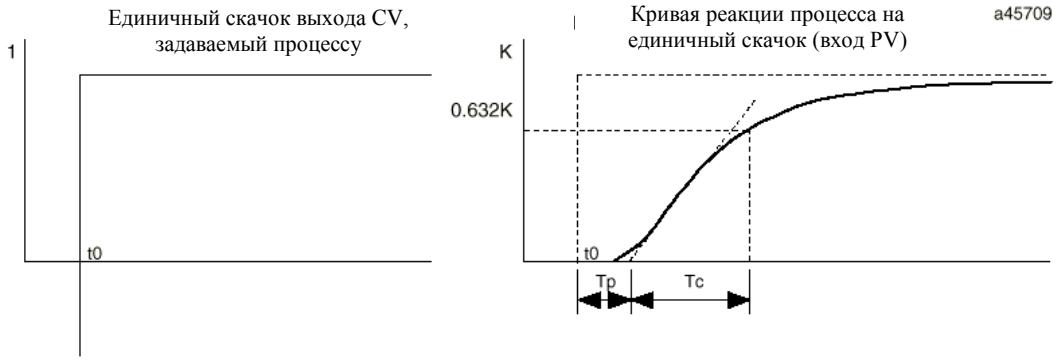
Коэффициенты усиления контура PID определяются характеристиками управляемого процесса. Двумя ключевыми вопросами при установке контура PID являются:

1. Как велико изменение PV при изменении CV на заданную величину или каково усиление разомкнутого контура?
2. Насколько быстр отклик системы или как быстро изменяется PV после скачкообразного изменения CV?

Многие процессы можно аппроксимировать величиной усиления, постоянной запаздывания первого порядка и чистым временем задержки. В частотной области передаточная функция системы с запаздыванием первого порядка и чисто временной задержкой имеет вид:

$$PV(s)/CV(s) = G(s) = K*e^{**}(-Tps)/(1 + Tcs)$$

Изображение реакции на единичный скачок в момент времени t0 во временной области дает переходную характеристику разомкнутого контура:



Следующие параметры модели процесса можно определить по переходной характеристике:

K	Коэффициент усиления разомкнутого контура процесса = конечное изменение PV/ изменение CV в момент времени t0 (Обратите внимание на отсутствие индекса при K)
Tp	Время задержки процесса или линии передачи, или мертвое время, после момента t0 и перед началом изменения PV
Tc	Постоянная времени процесса первого порядка, время после Tp, требуемое для достижения 63,.2% от установившегося значения PV.

Обычно самый быстрый путь измерения этих параметров заключается в переводе блока PID в ручной режим и создании малого скачка выхода CV, путем изменения команды Manual (%Ref+13), и последующего изображения временного отклика PV. Для медленных процессов это можно выполнить вручную, для быстрых процессов потребуется графопостроитель или регистрационный модуль компьютерной графики. Размер шага CV должен быть достаточно велик, чтобы вызвать наблюдаемое изменение PV, но и не слишком велик, чтобы не вызвать разрушение измеряемого процесса. Хорошим приближением будет величина от 2 до 10 % от диапазона между верхним и нижним уровнями ограничения.

Установка параметров пользователя и настройка коэффициентов усиления

Так как все параметры полностью определяются управляемым процессом, то не существует предварительно выбираемых значений, которые могут работать, однако обычно используется простая итеративная процедура выбора приемлемого усиления контура.

1. Установите все параметры пользователя на 0, затем установите значения верхнего и нижнего уровня ограничения, исходя из предполагаемых величин. Установите период выборки в пределах одной десятой/сотой от постоянной времени процесса.
2. Переведите блок в ручной режим и установите команду Manual (%Ref+13) на различные значения, чтобы проверить, не будет ли CV достигать верхнего или нижнего предельных значений. Запишите значение PV для некоторого значения CV и поместите его в SP.
3. Установите небольшое усиление для Kp, например, 100*максимум CV/максимум PV, и выключите ручной режим. Увеличьте скачком SP на 2% - 10% от максимального диапазона PV и просмотрите отклик PV. Увеличьте значение Kp, если отклик PV на ступеньку слишком медленный, или уменьшите значение Kp, если PV имеет заброс или осциллирует, не достигая установившегося состояния.
4. Когда Kp установлен, начинайте увеличивать Ki до достижения заброса, который приближается к установившемуся состоянию за 2-3 цикла осцилляции. При этом может потребоваться уменьшение Kp. Испробуйте при этом различные значения скачка и различные рабочие точки CV.
5. После установки Kp и Ki попробуйте добавить Kd для ускорения отклика, если это не вызовет появления осцилляций. Kd часто не является необходимым и не работает при зашумленном PV.
6. Проверьте усиления в различных рабочих точках SP и при необходимости установите зону нечувствительности и минимальное время нарастания. Некоторые процессы инверсного действия могут потребовать установки в конфигурационном слове битов знака ошибки или полярности.

Установка коэффициентов регулятора – метод Гейнса-Зиглера и Николса

Когда три параметра модели процесса K, Tp и Tc определены, их можно использовать для оценки начальных значений коэффициентов усиления PID-контура. Следующий подход, развитый в 1940 году Зиглером и Николсом, предназначен для обеспечения хорошего отклика на возмущение системы с коэффициентами, обеспечивающими отношение амплитуд 1/4. Отношением амплитуд является отношение второго пика к первому пику отклика замкнутого контура.

1. Вычислите скорость реакции:

   R = K/Tc

2. Для чисто пропорционального управления вычислите Kp:

   Kp = 1/(R∗Tp)=Tc/(K∗Tp)

3. Для пропорционального и интегрального управления используйте соотношения:

   Kp = 0.9/(R∗Tp)=0.9∗Tc/(K∗Tp)

   Ki = 0.3∗Kp/Tp

4. Для управления пропорционального, интегрального и по производной используйте соотношения:

   Kp = G/(R∗Tp) где G=1.2-2.0

   Ki = 0.5∗Kp/Tp

   Kd = 0.5∗Kp∗Tp

5. Проверьте, что период выборки лежит в диапазоне (Tp+Tc)/10 … (Tp+Tc)/1000

Другой подход, называемый процедурой «идеальной настройки», используется для обеспечения отклика с наименьшей задержкой, определяемой только временем задержки процесса Tp или мертвым временем

Kp = 2∗Tc/(3∗K∗Tp)

Ki = Tc

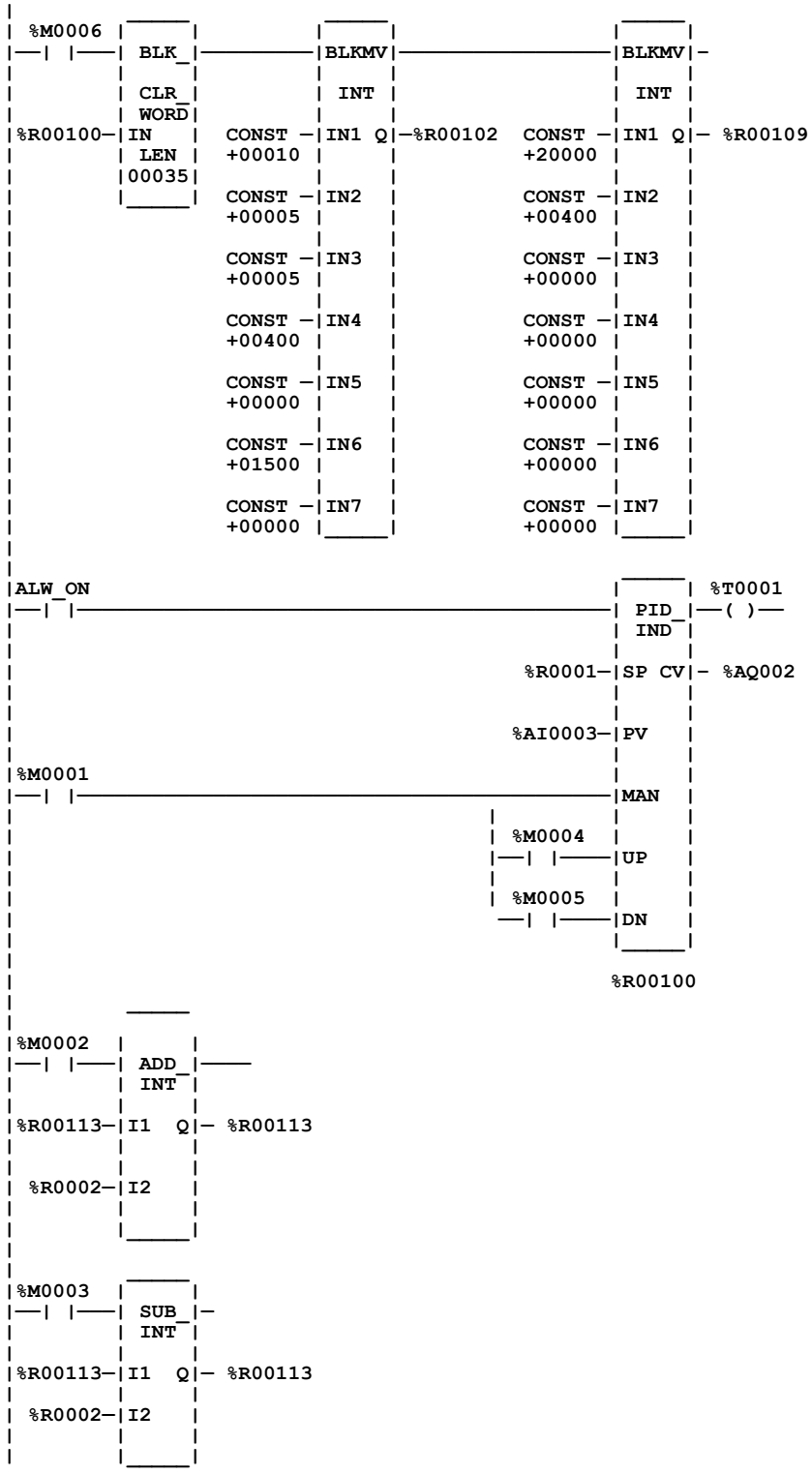
Ki = Ki/4 если используется производная

После определения коэффициентов усиления, их нужно преобразовать в целочисленные параметры пользователя. Чтобы избежать проблемы масштабирования, усиление процесса K должно вычисляться как изменение входных отсчетов PV, деленное на изменение выходного скачка в отсчетах CV, а не в инженерных PV и CV единицах процесса. Все временные интервалы должны задаваться в секундах. После определения коэффициентов Kp, Ki и Kd можно умножить Kp и Kd на 100 и ввести как целочисленные значения, тогда как Ki можно умножить на 1000 и ввести в массив %RefArray.

Пример вызова PID

Следующий пример имеет период выборки 100 мсек, Kp = 4.00 и Ki = 1.500. Уставка хранится в %R1 с выходом CV в %AQ2, а PV - в %AI3. Верхний и нижний пределы CV в данном случае должны быть 20000 и 400, и добавлена необязательная небольшая зона нечувствительности от +5 до -5. 40 слов массива RefArray начинаются в %R100. Обычно параметры пользователя установлены в массиве RefArray с клавишей PID Zoom F10, однако %M6 можно установить для повторной инициализации 14 слов, начиная с %R102 (%Ref+2), содержащих константы, хранящиеся в логике.

Блок можно переключать в ручной режим с %M1, так что можно будет настроить команду Manual ,%R113. Биты %M4 или %M5 можно использовать для увеличения или уменьшения %R113, PID CV и интегратора на 1 при каждом 100-миллисекундном исполнении. Для ускорения ручной работы биты %M2 и %M3 можно использовать для добавления или вычитания значения в %R2 к или из значения %R113 при каждом цикле ПЛК. Выход %T1 находится в состоянии ON, когда функция PID исполнена успешно.


%M0006
BLK_
CLR_
WORD
%R00100
IN
LEN
00035
BLKMV
INT
CONST
+00010
IN1 Q
%R00102
+20000
%R00109
IN2
+00005
+00400
IN3
+00000
IN4
IN5
IN6
+01500
IN7
ALW_ON
PID_
IND
%T0001
%R0001
SP CV
%AQ002
%AI0003
PV
%M0001
MAN
%M0004
UP
%M0005
DN
%R00100
%M0002
ADD_
%M0003
SUB_
I1
I2

Приложение

A

Время выполнения команд

ПЛК серий 90-30, 90-20, и Micro поддерживают большое количество различных функций и функциональных блоков. Это приложение содержит таблицы, отображающие размер памяти в байтах и время выполнения функций в микросекундах. Размер памяти – это число байт, запрашиваемых функцией в пользовательской программе, написанной на языке релейно-контактной логики.

Для каждой функции указывается два времени выполнения:

Время выполнения	Описание
Включено	Время, необходимое для выполнения функции или функционального блока, когда поток энергии входит и выходит в функцию. Обычно, оптимально время, когда данные, используемые в блоке, содержатся в ОЗУ (память адресуется словами) и не в дискретной памяти.
Выключено	Время, необходимое для функции, когда поток энергии не входит в функцию или функциональный блок; тем не менее, это не активное состояние, как когда таймер удерживается в состоянии сброса.

Примечание

Обновление таймеров и счетчиков происходит всякий раз при появлении их в программе, таймеры учитывают время выполнения последнего цикла, а счетчики количество поступивших импульсов.

Примечание

Для моделей ЦП 350, 351, 352, и 360, таймеры времени выполнения команд идентичны за исключением команды MOVE, которая отличается для модели ЦП 350 – смотрите отличия в примечании к таблице на странице A-6.

Таблица A-1. Время выполнения инструкций, стандартные модели

Группа функций	Функция	Включено				Выключено				Увеличение				Размер
		311	313	331	340/41	311	313	331	340/41	311	313	331	340/41	
Таймеры	On-Delay Timer	146	81	80	42	105	39	38	21	–	–	–	–	15
	Off-Delay Timer	98	47	44	23	116	63	58	32	–	–	–	–	9
	Timer	122	76	75	40	103	54	53	30	–	–	–	–	15
Счетчики	Up Counter	137	70	69	36	130	63	62	33	–	–	–	–	11
	Down Counter	136	70	69	37	127	61	61	31	–	–	–	–	11
Математические операции	Addition (INT)	76	47	46	24	41	0	1	0	–	–	–	–	13
	Addition (DINT)	90	60	60	34	41	1	0	0	–	–	–	–	13
	Subtraction (INT)	75	46	45	25	41	0	1	0	–	–	–	–	13
	Subtraction (DINT)	92	62	62	34	41	1	0	0	–	–	–	–	13
	Multiplication (INT)	79	49	50	28	41	0	1	0	–	–	–	–	13
	Multiplication (DINT)	108	80	101	43	41	1	0	0	–	–	–	–	13
	Division (INT)	79	51	50	27	41	0	1	0	–	–	–	–	13
	Division (DINT)	375	346	348	175	41	1	0	0	–	–	–	–	13
	Modulo Division (INT)	78	51	49	27	41	0	1	0	–	–	–	–	13
	Modulo Div (DINT)	134	103	107	54	41	1	0	0	–	–	–	–	13
	Square Root (INT)	153	124	123	65	42	0	1	0	–	–	–	–	9
	Square Root (DINT)	268	239	241	120	42	0	0	1	–	–	–	–	9
Операции отношений	Equal (INT)	66	35	36	19	41	1	1	0	–	–	–	–	9
	Equal (DINT)	86	56	54	29	41	1	0	0	–	–	–	–	9
	Not Equal (INT)	67	39	35	22	41	1	1	0	–	–	–	–	9
	Not Equal (DINT)	81	51	51	28	41	1	0	0	–	–	–	–	9
	Greater Than (INT)	64	33	35	20	41	1	1	0	–	–	–	–	9
	Greater Than (DINT)	89	59	58	32	41	1	0	0	–	–	–	–	9
	Greater Than/Eq (INT)	64	36	34	19	41	1	1	0	–	–	–	–	9
	Greater Than/Eq (DINT)	87	58	57	30	41	1	0	0	–	–	–	–	9
	Less Than (INT)	66	35		19	41	1	1	0	–	–	–	–	9
	Less Than (DINT)	87	57		30	41	1	1	0	–	–	–	–	9
	Less Than/Equal (INT)	66	36	34	21	41	1	1	0	–	–	–	–	9
	Less Than/Equal (DINT)	86	57	56	31	41	1	1	0	–	–	–	–	9
	Range (INT)	92	58	54	29	46	1	0	1	–	–	–	–	15
	Range(DINT)	106	75	57	37	45	0	0	0	–	–	–	–	15
	Range(WORD)	93	60	54	29	0	0	0	0	–	–	–	–	15

Примечания:

1. Время (в микросекундах) является базовым для реализаций 5.01 ПО Logicmaster 90-30/20 для моделей ЦП 311, 313, 340 и 341 (Release 7 для 331).
2. Для табличных функций, увеличение длины памяти происходит поблочно; для битовых операций единица измерений микросекунд/бит; для функций перемещения данных, микросекунд/количество бит (слов)..
3. Включая время для единиц одинарной длины типов памяти %R, %AI, и %AQ.
4. Время выполнения COMMREQ измерено между ЦП и HSC (высокоскоростным счетчиком)..
5. Для DOIO это время передачи значений в дискретный выходной модуль.
6. В тех случаях, когда имеется более чем одна возможность выбора, время указывается по наихудшему варианту выбора.

Таблица A-1. Время выполнение инструкций, стандартные модели - продолжение

Группа фукций	Функция	Включено				Выключено				Увеличение				Размер
		311	313	331	340/341	311	313	331	340/341	311	313	331	340/341	
Битовые Операции	Logical AND	67	37	37	22	42	0	0	1	–	–	–	–	13
	Logical OR	68	38	38	21	42	0	0	1	–	–	–	–	13
	Logical Exclusive OR	66	38	37	20	42	0	1	1	–	–	–	–	13
	Logical Invert, NOT	62	32	31	17	42	0	1	1	–	–	–	–	9
	Shift Bit Left	139	89	90	47	74	26	23	13	11.61	11.61	12.04	6.29	15
	Shift Bit Right	135	87	85	45	75	26	24	13	11.63	11.62	12.02	6.33	15
	Rotate Bit Left	156	127	126	65	42	1	1	0	11.70	11.78	12.17	6.33	15
	Rotate Bit Right	146	116	116	62	42	1	1	0	11.74	11.74	12.13	6.27	15
	Bit Position	102	72	49	38	42	1	0	0	–	–	–	–	13
	Bit Clear	68	38	35	21	42	1	1	1	–	–	–	–	13
	Bit Test	79	49	51	28	41	0	0	1	–	–	–	–	13
	Bit Set	67	37	37	20	42	0	0	0	–	–	–	–	13
	Masked Compare (WORD)	217	154	141	74	107	44	39	21	–	–	–	–	25
	Masked Compare (DWORD)	232	169	156	83	108	44	39	22	–	–	–	–	25
Переме-щение данных	Move (INT)	68	37	39	20	43	0	0	0	1.62	1.62	5.25	1.31	13
	Move (BIT)	94	62	64	35	42	0	0	0	12.61	12.64	12.59	6.33	13
	Move (WORD)	67	37	40	20	41	0	0	0	1.62	1.63	5.25	1.31	13
	Block Move (INT)	76	48	50	28	59	30	30	16	–	–	–	–	27
	Block Move (WORD)	76	48	49	29	59	29	28	15	–	–	–	–	27
	Block Clear	56	28	27	14	43	0	0	0	1.35	1.29	1.40	0.78	9
	Shift Register (BIT)	201	153	153	79	85	36	34	18	0.69	0.68	0.71	0.37	15
	Shift Register (WORD)	103	53	52	29	73	25	23	12	1.62	1.62	2.03	1.31	15
	Bit Sequencer	165	101	99	53	96	31	29	16	0.07	0.07	0.08	0.05	15
	COMM_REQ	1317	1272	1489	884	41	2	0	0	–	–	–	–	13
Табличные операции	Array Move													
	INT	230	201	177	104	72	41	40	20	1.29	1.15	10.56	2.06	21
	DINT	231	202	181	105	74	44	42	23	3.24	3.24	10.53	2.61	21
	BIT	290	261	229	135	74	43	42	23	–.03	–.03	-0.01	0.79	21
	BYTE	228	198	176	104	74	42	42	23	0.81	0.82	8.51	1.25	21
	WORD	230	201	177	104	72	41	40	20	1.29	1.15	10.56	2.06	21
	Search Equal													
	INT	197	158	123	82	78	39	37	20	1.93	1.97	2.55	1.55	19
	DINT	206	166	135	87	79	38	36	21	4.33	4.34	4.55	2.44	19
	BYTE	179	141	117	74	78	38	36	21	1.53	1.49	1.83	1.03	19
	WORD	197	158	123	82	78	39	37	20	1.93	1.97	2.55	1.55	19

Примечания:

1. Время (в микросекундах) является базовым для реализаций 5.01 ПО Logicmaster 90-30/20 для моделей ЦП 311, 313, 340 и 341 (Release 7 для 331).
2. Для табличных функций, увеличение длины памяти происходит поблочно; для битовых операций единица измерений микросекунд/бит; для функций перемещения данных, микросекунд/количество бит (слов)..
3. Включая время для единиц одинарной длины типов памяти %R, %AI, и %AQ.
4. Время выполнения COMMREQ измерено между ЦП и HSC (высокоскоростным счетчиком)..
5. Для DOIO это время передачи значений в дискретный выходной модуль.
6. В тех случаях, когда имеется более чем одна возможность выбора, время указывается по наихудшему варианту выбора.
7. Для инструкций, которые имеют увеличивающиеся значения, время выполнения равно времени увеличения умноженному на обрабатываемый массив (длина –1) плюс базовое время.

Таблица A-1. Время выполнение инструкций, стандартные модели - продолжение

Группа функций	Функция	Включено				Выключено				Увеличение				Размер
		311	313	331	340/41	311	313	331	340/41	311	313	331	340/41	
	Search Not Equal													
	INT	198	159	124	83	79	39	36	21	1.93	1.93	2.48	1.52	19
	DINT	201	163	132	84	79	37	35	21	6.49	6.47	6.88	3.82	19
	BYTE	179	141	117	73	79	38	36	19	1.54	1.51	1.85	1.05	19
	WORD	198	159	124	83	79	39	36	21	1.93	1.93	2.48	1.52	19
	Search Greater Than													
	INT	198	160	125	82	79	37	38	19	3.83	3.83	4.41	2.59	19
	DINT	206	167	135	88	78	38	36	20	8.61	8.61	9.03	4.88	19
	BYTE	181	143	118	73	79	37	36	19	3.44	3.44	3.75	2.03	19
	WORD	198	160	125	82	79	37	38	19	3.83	3.83	4.41	2.59	19
	Search Greater Than/Eq													
	INT	197	160	124	83	77	38	36	20	3.86	3.83	4.45	2.52	19
	DINT	205	167	136	87	80	39	36	21	8.62	8.61	9.02	4.87	19
	BYTE	180	142	118	75	79	37	37	20	3.47	3.44	3.73	2.00	19
	WORD	197	160	124	83	77	38	36	20	3.86	3.83	4.45	2.52	19
	Search Less Than													
	INT	199	159	124	84	78	38	36	20	3.83	3.86	4.48	2.48	19
	DINT	206	168	135	87	79	38	38	19	8.62	8.60	-1.36	4.88	19
	BYTE	181	143	119	75	80	38	37	20	3.44	3.44	3.75	2.00	19
	WORD	199	159	124	84	78	38	36	20	3.83	3.86	4.45	2.48	19
	Search Less Than/Equal													
	INT	200	158	124	82	79	38	37	21	3.79	3.90	4.45	2.55	19
	DINT	207	167	137	88	78	39	37	19	8.60	8.61	9.01	4.86	19
	BYTE	180	143	119	74	78	40	37	19	3.46	3.44	3.73	2.02	19
	WORD	200	158	124	82	79	38	37	21	3.79	3.90	4.45	2.55	19
Функции	Convert to INT	74	46	39	25	42	1	1	1	–	–	–	–	9
преобразо-вания	Convert to BCD–4	77	50	34	25	42	1	1	1	–	–	–	–	9

Примечания:

1. Время (в микросекундах) является базовым для реализаций 5.01 ПО Logicmaster 90-30/20 для моделей ЦП 311, 313, 340 и 341 (Release 7 для 331).
2. Для табличных функций, увеличение длины памяти происходит поблочно; для битовых операций единица измерений микросекунд/бит; для функций перемещения данных, микросекунд/количество бит (слов)..
3. Включая время для единиц одинарной длины типов памяти %R, %AI, и %AQ.
4. Время выполнения COMMREQ измерено между ЦП и HSC (высокоскоростным счетчиком)..
5. Для DOIO это время передачи значений в дискретный выходной модуль.
6. В тех случаях, когда имеется более чем одна возможность выбора, время указывается по наихудшему варианту выбора.
7. Для инструкций, которые имеют увеличивающиеся значения, время выполнения равно времени увеличения умноженному на обрабатываемый массив (длина –1) плюс базовое время.

Таблица A-1. Время выполнение инструкций, стандартные модели - продолжение

Группа функций	Функция	Включено				Выключено				Увеличение				Размер
		311	313	331	340/41	311	313	331	340/41	311	313	331	340/41	
Функции управления	Call a Subroutine	155	93	192	85	41	0	0	0	–	–	–	–	7
	Do I/O	309	278	323	177	38	1	0	0	–	–	–	–	12
	PID – ISA Algorithm	1870	1827	1812	929	91	56	82	30	–	–	–	–	15
	PID – IND Algorithm	2047	2007	2002	1017	91	56	82	30	–	–	–	–	15
	End Instruction	–	–	–	–	–	–	–	–	–	–	–	–	–
	Сервисные запросы													
	# 6	93	54	63	45	41	2	0	0	–	–	–	–	9
	# 7 (Read)	–	37	309	161	–	2	0	0	–	–	–	–	9
	# 7 (Set)	–	37	309	161	–	2	0	0	–	–	–	–	9
	#14	447	418	483	244	41	2	0	0	–	–	–	–	9
	#15	281	243	165	139	41	2	0	0	–	–	–	–	9
	#16	131	104	115	69	41	2	0	0	–	–	–	–	9
	#18	–	56	300	180	–	2	0	0	–	–	–	–	9
	#23	1689	1663	1591	939	43	1	0	0	–	–	–	–	9
	#26//30*	1268	1354	6680	3538	42	0	0	0	–	–	–	–	9
	#29	–	–	55	41	–	–	1	0	–	–	–	–	9
	Nested MCR/ENDMCR Combined	135	73	68	39	75	25	21	12	–	–	–	–	8

***** Служебный запрос #26/30 измеряется с использованием высокоскоростного счетчика, 16-канального выхода, в 5-слотовой стойке.

Примечания:

1. Время (в микросекундах) является базовым для реализаций 5.01 ПО Logicmaster 90-30/20 для моделей ЦП 311, 313, 340 и 341 (Release 7 для 331).
2. Для табличных функций, увеличение длины памяти происходит поблочно; для битовых операций единица измерений микросекунд/бит; для функций перемещения данных, микросекунд/количество бит (слов)..
3. Включая время для единиц одинарной длины типов памяти %R, %AI, и %AQ.
4. Время выполнения COMMREQ измерено между ЦП и HSC (высокоскоростным счетчиком).
5. Для DOIO это время передачи значений в дискретный выходной модуль.
6. В тех случаях, когда имеется более чем одна возможность выбора, время указывается по наихудшему варианту выбора.
7. Для инструкций, которые имеют увеличивающиеся значения, время выполнения равно времени увеличения умноженному на обрабатываемый массив (длина –1) плюс базовое время.

Таблица A-2. Время выполнения инструкций, высокопроизводительные модели

Группа Функций	Функция	Включено 350/351/36x	Выключено 350/351/36x	Увеличение 350/351/36x	Включено 352	Выключено 352	Увеличение 352	Размер
Таймеры	On-Delay Timer	4	6	–	4	5	–	15
	Timer	3	3	–	2	2	–	15
	Off-Delay Timer	3	3	–	3	2	–	15
Счетчики	Up Counter	1	3	–	2	2	–	13
	Down Counter	3	3	–	1	2	–	13
Математические	Addition (INT)	2	0	–	1	0	–	13
Функции	Addition (DINT)	2	0	–	2	0	–	19
	Addition (REAL)	52	0	–	33	0	–	17
	Subtraction (INT)	2	0	–	1	0	–	13
	Subtraction (DINT)	2	0	–	2	0	–	19
	Subtraction (REAL)	53	0	–	34	0	–	17
	Multiplication (INT)	21	0	–	21	0	–	13
	Multiplication (DINT)	24	0	–	24	0	–	19
	Multiplication (REAL)	68	1	–	38	1	–	17
	Division (INT)	22	0	–	22	0	–	13
	Division (DINT),	25	0	–	25	0	–	19
	Division (REAL)	82	2	–	36	2	–	17
	Modulo Division (INT)	21	0	–	21	0	–	13
	Modulo Div (DINT)	25	0	–	25	0	–	19
	Square Root (INT)	42	1	–	41	1	–	10
	Square Root (DINT)	70	0	–	70	0	–	13
	Square Root (REAL)	137	0	–	35	0	–	11
Тригонометрические функции	SIN (REAL)	360	0	–	32	0	–	11
	COS (REAL)	319	0	–	29	0	–	11
	TAN (REAL)	510	1	–	32	1	–	11
	ASIN (REAL)	440	0	–	45	0	–	11
	ACOS (REAL)	683	0	–	63	0	–	11
	ATAN (REAL)	264	1	–	33	1	–	11
Логарифмы	LOG (REAL)	469	0	–	32	0	–	11
	LN (REAL)	437	0	–	32	0	–	11
Экспонента	EXP	639	0	–	42	0	–	11
	EXPT	89	1	–	54	1	–	17
Перевод град.\рад.	Convert RAD to DEG	65	1	–	32	1	–	11
	Convert DEG to RAD	59	0		32	0		11

Примечания:

1. Время (в микросекундах) является базовым для реализаций 7 ПО Logicmaster 90-30/20 для моделей ЦП 351 и 352.
2. Для табличных функций, увеличение длины памяти происходит поблочно; для битовых операций единица измерений микросекунд/бит; для функций перемещения данных, микросекунд/количество бит (слов)..
3. Включая время для единиц одинарной длины типов памяти %R, %AI, и %AQ.
4. Время выполнения COMMREQ измерено между ЦП и HSC (высокоскоростным счетчиком).
5. Для DOIO это время передачи значений в дискретный выходной модуль.
6. В тех случаях, когда имеется более чем одна возможность выбора, время указывается по наихудшему варианту выбора.

Таблица A-2. Время выполнения инструкций, высокопроизводительные модели - продолжение

Группа функций	Функция	Включено 350/351/36x	Выключено 350/351/36x	Увеличение 350/351/36x	Включено 352	Выключено 352	Увеличение 352	Размер
Функции отношений	Equal (INT)	1	0	–	1	0	–	10
	Equal (DINT)	2	0	–	2	0	–	16
	Equal (REAL)	57	0	–	28	0	–	14
	Not Equal (INT)	1	0	–	1	0	–	10
	Not Equal (DINT)	1	0	–	1	0	–	16
	Not Equal (REAL)	62	0	–	31	0	–	14
	Greater Than (INT)	1	0	–	1	0	–	10
	Greater Than (DINT)	1	0	–	1	0	–	16
	Greater Than (REAL)	57	0	–	32	0	–	14
	Greater Than/Equal (INT)	1	0	–	1	0	–	10
	Greater Than/Equal (DINT)	1	0	–	1	0	–	10
	Greater Than/Equal (REAL)	57	1	–	31	1	–	14
	Less Than (INT)	1	0	–	1	0	–	10
	Less Than (DINT)	1	0	–	1	0	–	16
	Less Than (REAL)	58	1	–	36	1	–	14
	Less Than/Equal (INT)	1	0	–	1	0	–	10
	Less Than/Equal (DINT)	3	0	–	3	0	–	16
	Less Than/Equal (REAL)	37	0	–	37	0	–	14
	Range (INT)	2	1	–	2	1	–	13
	Range (DINT)	2	1	–	2	1	–	22
	Range (WORD)	1	0	–	1	0	–	13
Битовые операции	Logical AND	2	0	–	2	0	–	13
	Logical OR	2	0	–	2	0	–	13
	Logical Exclusive OR	1	0	–	1	0	–	13
	Logical Invert, NOT	1	0	–	1	0	–	10
	Shift Bit Left	31	1	1.37	31	1	1.37	16
	Shift Bit Right	28	0	3.03	28	0	3.03	16
	Rotate Bit Left	25	0	3.12	25	0	3.12	16
	Rotate Bit Right	25	0	4.14	25	0	4.14	16
	Bit Position	20	1	–	20	1	–	13
	Bit Clear	20	0	–	20	0	–	13
	Bit Test	20	0	–	20	0	–	13
	Bit Set	19	1	–	19	1	–	13
	Mask Compare (WORD)	52	0	–	52	0	–	25
	Mask Compare (DWORD)	50	0	–	49	0	–	25

Примечания:

1. Время (в микросекундах) является базовым для реализаций 7 ПО Logicmaster 90-30/20 для моделей ЦП 351 и 352.
2. Для табличных функций, увеличение длины памяти происходит поблочно; для битовых операций единица измерений микросекунд/бит; для функций перемещения данных, микросекунд/количество бит (слов)..
3. Включая время для единиц одинарной длины типов памяти %R, %AI, и %AQ.
4. Время выполнения COMMREQ измерено между ЦП и HSC (высокоскоростным счетчиком)..
5. Для DOIO это время передачи значений в дискретный выходной модуль.
6. В тех случаях, когда имеется более чем одна возможность выбора, время указывается по наихудшему варианту выбора.
7. Для инструкций, которые имеют увеличивающиеся значения, время выполнения равно времени увеличения умноженному на обрабатываемый массив (длина –1) плюс базовое время.

Таблица A-2. Время выполнения инструкций, высокопроизводительные модели - продолжение

Группа Функций	Функция	Включено 350/351/36X	Выключено 350/351/36X	Увеличение 350/351/36X	Включено 352	Выключено 352	Увеличение 352	Размер
Функции пересылки данных	Move (INT)	2	0	0.41	2	0	0.41	10
	Move (BIT)	28	0	4.98	28	0	4.98	13
	Move (WORD)	2	0	0.41	2	0	0.41	10
	Move (REAL)	24	1	0.82	24	1	0.82	13
	Block Move (INT)	2	0	–	2	0	–	28
	Block Move (WORD)	4	4	–	3	0	–	28
	Block Move (REAL)	41	0	–	41	0	–	13
	Block Clear	1	0	0.24	1	0	0.24	11
	Shift Register (BIT)	49	0	0.23	46	0	0.23	16
	Shift Register (WORD)	27	0	0.41	27	0	0.41	16
	Bit Sequencer	38	22	0.02	38	22	0.02	16
	COMM_REQ	765	0	–	765	0	–	13
Табличные функции	Array Move							
	INT	54	0	0.97	54	0	0.97	22
	DINT	54	0	0.81	54	0	0.81	22
	BIT	69	0	0.36	69	0	0.36	22
	BYTE	54	1	0.64	54	1	0.64	22
	WORD	54	0	0.97	54	0	0.97	22
	Search Equal							
	INT	37	0	0.62	37	0	0.62	19
	DINT	41	1	1.38	41	1	1.38	22
	BYTE	35	0	0.46	35	0	0.46	19
	WORD	37	0	0.62	37	0	0.62	19
	Search Not Equal							
	INT	37	0	0.62	37	0	0.62	19
	DINT	38	0	2.14	38	0	2.14	22
	BYTE	37	0	0.47	37	0	0.47	19
	WORD	37	0	0.62	37	0	0.62	19
	Search Greater Than							
	INT	37	0	1.52	37	0	1.52	19
	DINT	39	0	2.26	39	0	2.26	22
	BYTE	36	1	1.24	36	1	1.24	19
	WORD	37	0	1.52	37	0	1.52	19
	Search Greater Than/Equal							
	INT	37	0	1.48	37	0	1.48	19
	DINT	39	0	2.33	39	0	2.33	22
	BYTE	37	1	1.34	37	1	1.34	19
	WORD	37	0	1.48	37	0	1.48	19

Примечания:

1. Время (в микросекундах) является базовым для реализаций 7 ПО Logicmaster 90-30/20 для моделей ЦП 351 и 352.
2. Для табличных функций, увеличение длины памяти происходит поблочно; для битовых операций единица измерений микросекунд/бит; для функций перемещения данных, микросекунд/количество бит (слов)..
3. Включая время для единиц одинарной длины типов памяти %R, %AI, и %AQ.
4. Время выполнения COMMREQ измерено между ЦП и HSC (высокоскоростным счетчиком)..
5. Для DOIO это время передачи значений в дискретный выходной модуль.
6. В тех случаях, когда имеется более чем одна возможность выбора, время указывается по наихудшему варианту выбора.
7. Для инструкций, которые имеют увеличивающиеся значения, время выполнения равно времени увеличения умноженному на обрабатываемый массив (длина –1) плюс базовое время.

Таблица A-2. Время выполнения инструкций, высокопроизводительные модели - продолжение

Группа функций	Функция	Включено 350/351/36x	Выключено 350/351/36x	Увеличение 350/351/36x	Включено 352	Выключено 352	Увеличение 352	Размер
	Search Less Than							
	INT	37	0	1.52	37	0	1.52	19
	DINT	41	1	2.27	41	1	2.27	22
	BYTE	37	0	1.41	37	0	1.41	19
	WORD	37	0	1.52	37	0	1.52	19
	Search Less Than/Equal							
	INT	38	0	1.48	38	0	1.48	19
	DINT	40	1	2.30	40	1	2.30	22
	BYTE	37	0	1.24	37	0	1.24	19
	WORD	38	0	1.48	38	0	1.48	19
Функции преобразования	Convert to INT	19	1	–	19	1	–	10
	Convert to BCD-4	21	1	–	21	1	–	10
	Convert to REAL	27	0	–	21	0	–	8
	Convert to WORD	28	1	–	30	1	–	11
	Truncate to INT	32	0	–	32	0	–	11
	Truncate to DINT	63	0	–	31	0	–	11
Функции управления	CALL	72	1	–	73	1	–	7
	DO I/O	114	1	–	115	1	–	13
	PID – ISA алгоритм*	162	34	–	162	34	–	16
	PID – IND алгоритм*	146	34	–	146	34	–	16
	Функция End	–	–	–	–	–	–	–
	Сервисные запросы							
	#6	22	1	–	22	1	–	10
	#7 (Read)	75	1	–	75	1	–	10
	#7 (Set)	75	1	–	75	1	–	10
	#14	121	1	–	121	1	–	10
	#15	46	1	–	46	1	–	10
	#16	36	1	–	36	1	–	10
	#18	261	1	–	261	1	–	10
	#23	426	0	–	426	0	–	10
	#26//30**	2260	1	–	2260	1	–	10
	#29	20	0	–	20	0	–	10
	#43							
	Nested MCR/ENDMCR Combined	1	1	–	1	1	–	4
	Sequential Event Recorder (SER)	См. табл. A-3	26.50	См. табл A-3				

* Функция ПИД регулирования выполняется показанное время в модуле ЦП 351 release 6.5.

** Служебный запрос #26/30 измеряется с использованием высокоскоростного счетчика, 16-канального выхода, в 5-слотовой стойке.

Примечания:

1. Время (в микросекундах) является базовым для реализаций 7 ПО Logicmaster 90-30/20 для моделей ЦП 351 и 352.
2. Для табличных функций, увеличение длины памяти происходит поблочно; для битовых операций единица измерений микросекунд/бит; для функций перемещения данных, микросекунд/количество бит (слов)..
3. Включая время для единиц одинарной длины типов памяти %R, %AI, и %AQ.
4. Время выполнения COMMREQ измерено между ЦП и HSC (высокоскоростным счетчиком)..
5. Для DOIO это время передачи значений в дискретный выходной модуль.
6. В тех случаях, когда имеется более чем одна возможность выбора, время указывается по наихудшему варианту выбора.
7. Для инструкций, которые имеют увеличивающиеся значения, время выполнения равно времени увеличения умноженному на обрабатываемый массив (длина –1) плюс базовое время.

Таблица A-3. Время выполнения функции SER

Конфигурация	Пример	Время (мкс)
Нет потока энергии (Выключено)	—	26.50
Последовательно		
8 каналов	%I1—8	79.94
16 каналов	%I1—16	80.58
24 канала	%I1—24	81.56
32 канала	%I1—32	81.73
8 + 8 последовательных каналов	%I1—8 и %Q1—8	111.03
8 + 8 + 8 последовательных каналов	%I1—8, %Q1—8 и %M1—8	143.38
8 + 8 + 8 + 8 последовательных каналов	%I1—8, %Q1—8 и %M1—8 и %T1—8	175.79
Не последовательно	%I1, %M10, %Q3, и т.д.	
8 каналов		299.64
16 каналов		552.83
24 канала		806.35
32 канала		1059.85
Сброс		
с 8 каналами	—	162.63
с 16 каналами	—	267.51
с 24 каналами	—	372.73
с 32 каналами	—	477.95

Примечание: Если задан слот с входным модулем, прибавьте дополнительно 46 мкс к каждому времени **последовательной** и **непоследовательной** работы.

При срабатывании триггера, ко времени выполнения блока SER добавляется 29 мкс при использовании формата BCD или 148 мкс при использовании Posix формата.

Время, указанное для сброса вычислялось при использовании размера буфера выборки - 1024. (Сброс полностью очищает буфера выборки.)

Размер инструкций для высокопроизводительных моделей ЦП

Размер памяти в байтах – это число байт занимаемое инструкцией в памяти прикладной программы. В моделях ЦП 351 и 352 для большинства булевых функций требуется 3 байта – смотри таблицу A-4.

Таблица A-4. Размер инструкций для ЦП 350—352, 360, 363, и 364

Инструкция	Размер
No operation	1
Pop stack and AND to top	1
Pop stack and OR to top	1
Duplicate top of stack	1
Pop stack	1
Initial stack	1
Label	5
Jump	5
Все остальные инструкции	3
Функциональные блоки—см. Таблицу A-2	–

Время выполнения булевых операций

Следующая таблица содержит время выполнения операций над катушками и контактами для ЦП контроллеров Series 90-30.

Таблица A-5. Время выполнения булевых операций

Модель ЦП	Время выполнения для 1 000 булевых контактов/обмоток
Модель 350 и 360-е	0.22 миллисекунд
Модель 340/341	0.3 миллисекунд
Модель 331	0.4 миллисекунд
Модель 313/323	0.6 миллисекунд
Модель 311	18.0 миллисекунд

Приложение

B

Интерпретация таблиц сбоев

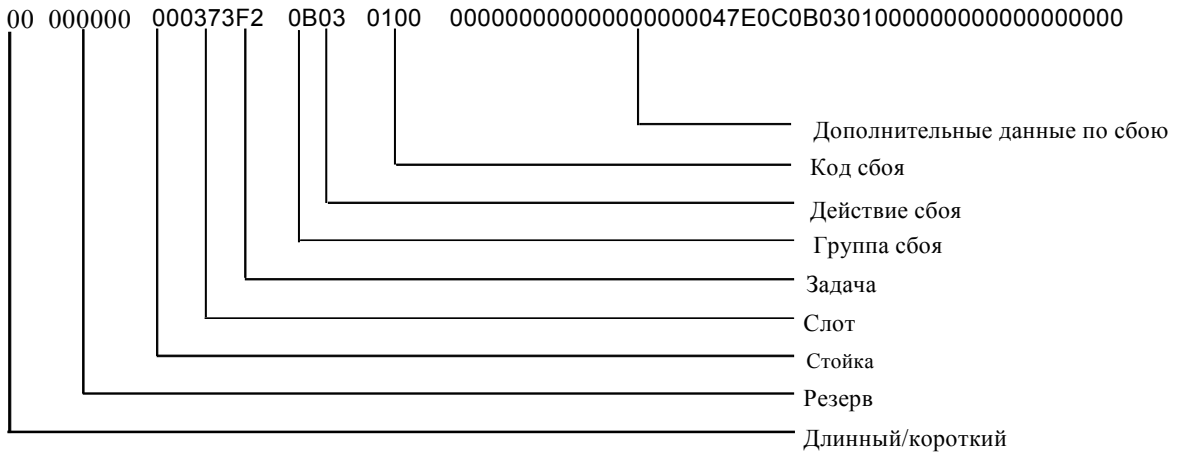
ПЛК Series 90-30, Series 90-20 и Series 90 Micro поддерживают две таблицы сбоев: таблицу сбоев модулей В/В для регистрации сбоев системы В/В (включая контроллеры В/В) и таблицу сбоев ПЛК для регистрации сбоев, происходящих в ПЛК. Информация в данном приложении позволяет Вам получить объяснение записей в этих таблицах. Таблицы сбоев содержат похожую информацию.

- Таблица сбоев ПЛК содержит:
  - Место нахождения сбоя.
  - Описание сбоя.
  - Дату и время, когда произошел сбой.
- Таблица сбоев В/В содержит:
  - Место нахождения сбоя.
  - Ссылку на адрес.
  - Категорию сбоя.
  - Тип сбоя.
  - Дату и время, когда произошел сбой.

Доступ к таблицам сбоев осуществляется посредством ПО для программирования. Для информации относительно доступа к таблицам, обратитесь к документу GFK-0466, *Logicmaster 90 Руководство пользователя*.

Таблица сбоев ПЛК

На следующем рисунке приведены все поля записи о сбое «Несоответствие системной конфигурации»:



Ниже все поля записи о сбое «Несоответствие системной конфигурации» представлены в виде таблицы. (Все данные приведены в шестнадцатеричном формате.)

Поле	Значение	Описание
Длинный/короткий	00	Данный сбой содержит 8 байт дополнительных данных
Стойка	00	Главная стойка (ведущая)
Слот	03	3 слот
Задача	44	
Группа сбоя	0B	Несоответствие системной конфигурации
Действие сбоя	03	Фатальный сбой
Код ошибки	01	

В следующих параграфах описываются все поля. Включая таблицы описывающие диапазон значений каждого поля.

Признак «Длинный/короткий»

Этот байт указывает на объем памяти, отводящийся для дополнительной информации о сбое: либо 8 байт, либо 24 байт.

Тип	Код	Длина данных
Короткий	00	8 байт
Длинный	01	24 байта

Резерв

Шесть байт данного поля – это «холостые» байты. Они нужны лишь для того, чтобы длина записи в таблице сбоев ПЛК была равна длине записи в таблице сбоев В/В.

Стойка

Номер стойки может варьироваться от 0 до 7. Нулевая стойка – это главная стойка, содержащая ЦП. Стойки от 1 до 7 – это стойки расширения, подсоединяемые к главной стойке через кабель расширения.

Слот

Слот – место модуля в стойке. Номер слота может варьироваться от 0 до 9. Модуль ЦП ПЛК обязательно должен размещаться в слоте 1 в главной стойке (нулевая стойка).

Задача

Номер задачи может варьироваться от 0 до +65535. Иногда номер задачи содержит дополнительную информацию для инженеров поддержки систем ПЛК; обычно, это поле игнорируется

Группа сбоя ПЛК

Группа сбоя это самый высокий уровень классификации сбоев. Он определяет общую категорию сбоев. Описание сбоя отображается в программном обеспечении Logicmaster 90-30/20/Micro и определяется по группе и коду сбоя.

Таблица B-1 содержит возможные группы сбоев для таблицы сбоев ПЛК.

Последняя немаскируемая группа сбоев - ***Additional PLC Fault Codes,*** введена для управления неизвестными сбоями, появляющимися в системе. Все нераспознанные коды сбоев В/В представлены в этой группе.

Таблица B-1. Группы сбоев ПЛК

Номер группы		Имя группы	Действие сбоя
Десятичный вид	Шестнадцатеричный вид		
1	1	Потеря или отсутствие стойки	Фатальное
4	4	Потеря или отсутствие дополнительного модуля	Диагностическое
5	5	Добавленная или новая стойка	Диагностическое
8	8	Добавленный или новый дополнительный модуль	Диагностическое
11	B	Несоответствие системной конфигурации	Фатальное
12	C	Ошибка системной шины	Диагностическое
13	D	Аппаратный сбой модуля ЦП	Фатальное
14	E	Не фатальный аппаратный сбой модуля	Диагностическое
16	10	Сбой программного обеспечения дополнительного модуля	Диагностическое
17	11	Ошибка контрольной суммы программы	Фатальное
18	12	Низкое напряжение батареи	Диагностическое
19	13	Превышено время цикла	Диагностическое
20	14	Заполнена таблица сбоев ПЛК	Диагностическое
21	15	Заполнена таблица сбоев модулей В/В	Диагностическое
22	16	Ошибка приложения	Диагностическое
–	–	Дополнительные коды сбоев ПЛК	Как оговорено
128	80	Сбой системной шины	Фатальное
129	81	Нет пользовательской программы	Информационное
130	82	Повреждена память	Фатальное
132	84	Отказ в доступе; неправильный пароль	Информационное
135	87	Сбой программного обеспечения ЦП	Фатальное
137	89	Сбой при записи в ПЛК	Фатальное

Действия сбоя

Каждому сбою соответствует одно из трех действий, которое он может произвести в системе. Эти действия фиксированы в ПЛК Series 90-30 и не могут быть изменены пользователем.

Таблица B-2. Действия сбоя ПЛК

Действие сбоя	Действия, выполняемые ЦП	Код
Информационное	Запись сбоя в таблицу	1
Диагностическое	Запись сбоя в таблицу Установка ссылок на сбой	2
Фатальное	Запись сбоя в таблицу Установка ссылок на сбой Переход в режим STOP	3

Код ошибки

Код ошибки способствует дальнейшему определению сбоя. Для каждой группы сбоев имеется свой набор кодов ошибок. Таблица B-3 содержит коды ошибок для группы сбоев программного обеспечения ПЛК (группа 87H).

Таблица B-3. Коды ошибок для группы сбоев программного обеспечения ПЛК

Десятичный	Шестнадцатеричный	Имя
20	14	Нарушена целостность памяти программы
39	27	Нарушена целостность памяти программы
82	52	Сбой связи по системной шине
90	5A	ПЛК остановлен по требованию пользователя
Все остальные		Внутренняя ошибка ПО ЦП

Таблица B-4 отображает коды ошибок для всех остальных групп сбоев.

Таблица B-4. Коды ошибок для сбоев ПЛК

Десятичный	Шестнадцатеричный	Имя
Коды ошибок ПЛК для группы «Потеря дополнительного модуль» (4)		
44	2C	Сбой при перезапуске дополнительного модуля
45	2D	Сбой при перезапуске дополнительного модуля
255	FF	Сбой связи с дополнительным модулем
Коды ошибок для группы «Сброс, добавления, или новый дополнительный модуль» (8)		
2	2	Перезапуск модуля завершен
	Все остальные	Сброс, добавление или новый дополнительный модуль
Коды ошибок для группы «Сбой в ПО дополнительного модуля» (10 hex)		
1	1	Неподдерживаемый тип платы
2	2	COMREQ – буфер полон выходящими сообщениями для старта COMREQ
3	3	COMREQ – буфер заполнен ответами
5	5	Потеря запроса при связи по системной шине ПЛК
11	B	Ошибка использования ресурсов
13	D	Ошибка пользовательской программы
401	191	Повреждено ПО модуля; требуется перезагрузка
Коды ошибок для группы «Несоответствие системной конфигурации» (B hex)		
8	8	Несоответствие расширения
10	A	Неподдерживаемые возможности
23	17	Программа превышает лимит памяти
Коды ошибок для группы «Ошибка системной шины» (C hex)		
Все		Ошибка системной шины
Коды ошибок для группы «Несоответствие контрольной суммы программы» (11 hex)		
3	3	Ошибка контрольной суммы программы или блока программы
Коды ошибки для сообщения «Низкое напряжение батареи»		
0	0	Разряженная батарея в модуле ЦП ПЛК или другом модуле
1	1	Низкое напряжение на батарее модуля ЦП ПЛК или другом модуле
Коды ошибок для группы «Ошибка пользовательской программы» (16 hex)		
2	2	Сработал сторожевой таймер ПЛК
5	5	COMREQ – режим WAIT не доступен для этой команды
6	6	COMREQ – неверный идентификатор задачи
7	7	Стек программы переполнен
Коды ошибок для группы «Ошибка системной шины» (80 hex)		
1	1	Операционная система
Коды ошибок для группы «Повреждение ОЗУ пользователя при включении» (82 hex)		
1	1	Обнаружена поврежденная ОЗУ пользователя при включении питания
2	2	Обнаружена некорректная булева операция
3	3	PLC_ISCP_PC_OVERFLOW
4	4	PRG_SYNTAX_ERR
Коды ошибок для группы «Аппаратный сбой ЦП» (D hex)		
Все коды		Аппаратный сбой ЦП

Дополнительный код сбоя

Это поле содержит дополнительное описание сбоя. Например, поле может содержать:

Corrupted User RAM Group:

Для сбоев группы «Несоответствие системной конфигурации» это поле означает следующее:

Таблица B-5. Дополнительные данные о сбое «Illegal Boolean Opcode Detected»

Значение	Описание
[0]	Ошибка в содержимом регистра ISCP
[1]	Ошибка в OPCODE
[2,3]	Счетчик программ ISCP
[4,5]	Номер функции

PLC CPU Hardware Failure (RAM Failure):

При сбоях ОЗУ в модуле ЦП (один из сбоев, регистрируемых как «аппаратный сбой ЦП»), первые четыре байта этого поля сохраняют адрес поврежденной памяти.

Время сбоя ПЛК

Шесть байт текущего времени - это значение системных часов ПЛК, когда был зарегистрирован сбой. (Значения – в формате BCD)

Таблица B-6. Время сбоя ПЛК

Номер байта	Описание
1	Секунды
2	Минуты
3	Часы
4	День
5	Месяц
6	Год

Таблица сбоев B/B

На следующем рисунке в шестнадцатеричном виде приведены все поля записи о сбое:

В следующих параграфах описываются все поля, включая таблицы, описывающие диапазон значений каждого поля.

Признак «Длинный/короткий»

Этот байт указывает на объем памяти отводящийся для дополнительной информации о сбое: либо 5 байт, либо 21 байт.

Таблица B-7. Формат байта «Длинный/короткий»

Тип	Код	Кол-во байт
короткий	02	5 байт
длинный	03	21 байт

Начальный адрес

Начальный адрес – это трехбайтовый адрес, содержащие в себе тип памяти B/B и расположение (или смещение) в памяти, которые соответствуют каналу, вызвавшему этот сбой. Или, когда происходит сбой блока Genius или аналогового модуля, начальный адрес указывает первый канал блока, в котором произошел сбой.

Таблица B-8. Начальный адрес сбоев B/B

Байт	Описание	Диапазон значений
0	Тип памяти	0 – FF
1–2	Смещение	0 – 7FF

Байт типа памяти первый и может содержать следующие значения.

Таблица B-9. Типы памяти адресов ссылок B/B

Имя	Значение (шестнадцатеричное)
Аналоговый вход	0A
Аналоговый выход	0C
Группа аналоговых сигналов	0D
Дискретный вход	10 or 46
Дискретный выход	12 or 48
Группа дискретных сигналов	1F

Адрес сбоя B/B

Адрес сбоя B/B представлен шестью байтами и содержит в себе адреса стойки, слота, шины, канала и модуля B/B, который сгенерировал сбой. Адрес канала – WORD; все другие адреса имеют тип байт. Все пять значений могут быть не представлены в сбое.

Когда адрес сбоя B/B не представлен всеми пятью адресами, то отсутствующий адрес заменяется числом 7F. Например, если 7F появляется в байте шины, значит, сбой произошел в модуле шинного контроллера. Только переменные стойки и слота являются значащими.

Стойка

Номер стойки может варьироваться от 0 до 7. Нулевая стойка – это главная стойка, содержащая ЦП. Стойки от 1 до 7 – это стойки расширения, подсоединяемые к главной стойке через кабель расширения.

Слот

Слот – место модуля в стойке. Номер слота может варьироваться от 0 до 9. Модуль ЦП ПЛК обязательно должен размещаться в слоте 1 в главной стойке (нулевая стойка).

Канал

Значение может варьироваться от 1 до 1024 (десятичные значения). Оно говорит о том, какой канал в модуле привел к сбою.

Группа сбоя B/B

Группа сбоя это самый высокий уровень классификации сбоев. Он определяет общую категорию сбоев. Описание сбоя отображается в программном обеспечении Logicmaster 90-30/20/Micro и определяется по группе и коду сбоя.

Таблица B-10 содержит возможные группы сбоев для таблицы сбоев B/B. Группы с номерами больше чем 80 (шестнадцатеричное значение) – это маскируемые сбои.

Последняя немаскируемая группа сбоев - ***Additional PLC Fault Codes,*** введена для управления неизвестными сбоями, появляющимися в системе. Все нераспознанные коды сбоев B/B представлены в этой группе.

Таблица B-10. Группы сбоев B/B

Номер группы	Имя группы	Действие сбоя
3	Потеря или отсутствие модуля B/B	Диагностическое
7	Добавление или новый модуль B/B	Диагностическое
9	сбой IOC или шины B/B	Диагностическое
A	Сбой модуля B/B	Диагностическое
–	Дополнительные коды сбоев B/B	Как оговорено

Действия сбоя

Каждому сбою соответствует одно из трех действий, которое он может произвести в системе. Эти действия фиксированы в ПЛК Series 90-30 и не могут быть изменены пользователем.

Таблица B-11. Действия сбоев B/B

Действие сбоя	Действия, выполняемые ЦП	Код
Информационное	Запись сбоя в таблицу	1
Диагностическое	Запись сбоя в таблицу Установка ссылок на сбой	2
Фатальное	Запись сбоя в таблицу Установка ссылок на сбой Переход в режим STOP	3

Специальные данные сбоя B/B

Запись сбоя в таблице сбоев B/B может содержать до 5 байт специальных данных B/B.

Символические специальные данные сбоя

Таблица B-12 содержит данные, которые запрашиваются для блока согласования конфигурации.

Таблица B-12. Специальные данные сбоя B/B

Десятичный номер	Шестнадцатеричный код	Описание
Согласование конфигурации		
	1	Согласование на входе – тристабильное
	2	Согласование на входе
	3	Согласование на выходе

Действия сбоев для специальных сбоев

Вынужденное/не вынужденное согласование сбоев регистрируется как информационные сбои. Все остальные либо диагностические, либо фатальные.

Такие сбои как несоответствие номера модели, несоответствие типа B/B и несуществующий модуль B/B регистрируются в таблице сбоев ПЛК в группе «Несоответствие системной конфигурации». Они не регистрируются в таблице сбоев B/B.

Время сбоя B/B

Шесть байт текущего времени это значение системных часов ПЛК, когда сбой был зарегистрирован модулем ЦП. (Значение в формате BCD)

Таблица B-13. Время сбоя B/B

Номер байта	Описание
1	Секунды
2	Минуты
3	Часы
4	День
5	Месяц
6	Год

Приложение

C

Мнемокоды команд

При программировании в режиме Display/Edit, вы можете быстро ввести или найти инструкции программирования: по типу символ «амперсант» (&) и следующим за ним мнемоникой инструкции. Для некоторых инструкций, вы можете также установить ссылку на адрес или имя, метку, или адрес расположения ссылки.

Это приложение содержит список мнемокодов программирования для ПО Logicmaster 90-30/20/Micro. Перечень мнемокодов отображен в таблице.

В любой момент в процессе программирования, вы можете отобразить экран помощи, содержащий эти мнемокоды при нажатии клавиш ALT-I.

Группа функций	**Инструкция**	**Мнемокоды**						
		Все	**INT**	**DINT**	**BIT**	**BYTE**	**WORD**	**REAL**
Контакты	Any Contact	&CON	&CON					
	Normally Open Contact	&NOCON	&NOCON					
	Normally Closed Contact	&NCCON	&NCCON					
	Continuation Contact	&CONC	&CONC					
Обмотки	Any Coil	&COI	&COI					
	Normally Open Coil	&NOCOI	&NOCOI					
	Negated Coil	&NCCOI	&NCCOI					
	Positive Transition Coil	&PCOI	&PCOI					
	Negative Transition Coil	&NCOI	&NCOI					
	SET Coil	&SL	&SL					
	RESET Coil	&RL	&RL					
	Retentive SET Coil	&SM	&SM					
	Retentive RESET Coil	&RM	&RM					
	Retentive Coil	&NOM	&NOM					
	Negated Retentive Coil	&NCM	&NCM					
	Continuation Coil	&COILC	&COILC					
Связи	Horizontal Link	&HO	&HO					
	Vertical Link	&VE	&VE					
Таймеры	On Delay Timer	&ON	&ON					
	Elapsed Timer	&TM	&TM					
	Off Delay Timer	&OF	&OF					
Счетчики	Up Counter	&UP	&UP					
	Down Counter	&DN	&DN					

Группа функций	Инструкция	Мнемокоды							
		Все	BCD-4	INT	DINT	BIT	BYTE	WORD	REAL
Математические	Addition Subtraction Multiplication Division Modulo Square Root Sine Cosine Tangent Inverse Sine Inverse Cosine Inverse Tangent Base 10 Logarithm Natural Logarithm Power of e Power of x	&AD &SUB &MUL &DIV &MOD &SQ &SIN &COS &TAN &ASIN &ACOS &ATAN &LOG &LN &EXP &EXPT		&AD_I &SUB_I &MUL_I &DIV_I &MOD_I &SQ_I	&AD_DI &SUB_DI &MUL_DI &DIV_DI &MOD_DI &SQ_DI				&AD_R &SUB_R &MUL_R &DIV_R &MOD_R&SQ_R
Функции отношений	Equal Not Equal Greater Than Greater or Equal Less Than Less Than or Equal	&EQ &NE &GT &GE &LT &LE		&EQ_I &NE_I &GT_I &GE_I &LT_I &LE_I	&EQ_DI &NE_DI &GT_DI &GE_DI &LT_DI &LE_DI				&EQ_R &NE_R &GT_R &GE_R &LT_R &LE_R
Битовые операции	AND OR Exclusive OR NOT Bit Shift Left Bit Shift Right Bit Rotate Left Bit Rotate Right Bit Test Bit Set Bit Clear Bit Position Masked Compare	&AN &OR &XO &NOT &SHL &SHR &ROL &ROR &BT &BS &BCL &BP &MCMP						&AN_W &OR_W &XO_W &NOT_W &SHL_W &SHR_W &ROL_W &ROR_W &BT_W &BS_W &BCL_W &BP_W &MCM_W	
Функции преобразования	Convert to Integer Convert to Double Integer Convert to BCD–4 Convert to REAL Convert to WORD Truncate to Integer Truncate to Double Integer	&TO_INT &TO_DINT &BCD4 &TO_REAL &TO_W &TRINT &TRDINT	&TO_INT_BCD4	&MOV &BLKM &BLKC &SHF &BI &COMMR	&TO_REAL_DI			&TO_REAL_W	&BCD4_R

Группа функций	Инструкция	Мнемокоды						
		Все	INT	DINT	BIT	BYTE	WORD	REAL
Пересылка данных	Move Block Move Block Clear Shift Register Bit Sequencer Communications Request	&MOV &BLKM &BLKC &SHF &BI &COMMR	&MOV_I &BLKM_I		&MOV_BI &SHF_BI		&MOV_W &BLKM_W &AR_W	&MOV_R &BLKM_R
Операции с таблица-ми	Array Move Search Equal Search Not Equal Search Greater Than Search Greater Than or Equal Search Less Than Search Less Than or Equal	&AR &SRCHE &SRCHN &SRCHGT &SRCHGE &SRCHLT &SRCHLE	&AR_I &SRCHE_I &SRCHN_I &SRCHGT_I &SRCHGE_I &SRCHLT_I &SRCHLE_I	&AR_DI &SRCHE_DI &SRCHN_DI &SRCHGT_DI &SRCHGE_DI &SRCHLT_DI &SRCHLE_DI	&AR_BI	&AR_BY &SRCHE_BY &SRCHN_BY &SRCHGT_BY &SRCHGE_BY &SRCHLT_BY &SRCHLE_BY	&AR_W &SRCHE_W &SRCHN_W &SRCHGT_W &SRCHGE_W &SRCHLT_W &SRCHLE_W_	
Функции управле-ния	Call a Subroutine Do I/O SER PID – ISA Algorithm PID – IND Algorithm SFC Reset End Rung Explanation System Services Request Master Control Relay End Master Control Relay Nested Master Control Relay Nested End Master Cntl Relay Jump Nested Jump Label Nested Label	&CA &DO &SER &PIDIS &PIDIN &SFCR &END &COMME &SV &MCR &ENDMCR &MCRN &ENDMCRN &JUMP &JUMPN &LABEL &LABELN							

Приложение

D

Быстрые клавиши

Это приложение содержит перечень функций клавиатуры, которые активны в среде программирования LogicMaster. На дисплее эту информацию можно просмотреть, нажав комбинацию клавиш ALT-K.

Последовательность клавиш	**Описание**	**Последовательность клавиш**	**Описание**
Быстрые клавиши, доступные в режиме Software Package			
ALT-A	Прервать.	CTRL-Break	Выход из режима Software Package.
ALT-C	Очистит поле.	Esc	Уменьшить изображение.
ALT-M	Изменить режим программирования.	CTRL-Home	Предыдущее содержание ком. строки.
ALT-R	Изменить статус ПЛК Run/Stop.	CTRL-End	Следующее содержание ком. строки.
ALT-E	Переключить в статусную область.	CTRL- ←	Курсор налево через поле.
ALT-J	Переключить в командную строку.	CTRL- →	Курсор направо через поле.
ALT-L	Список файлов в директории.	CTRL-D	Уменьшить адресную ссылку.
ALT-P	Печать экрана.	CTRL-U	Увеличить адресную ссылку.
ALT-H	Помощь.	Tab	Изменить/увеличить значение поля.
ALT-K	Клавиши помощи.	Shift-Tab	Изменить/уменьшить значение поля.
ALT-I	Помощь по мнемоникам.	Enter	Принять содержимое поля.
ALT-N	Переключение опций дисплея.	CTRL-E	Вывести последнюю ошибку.
ALT-T	Старт режима Teach.	F12 или Keypad -	Переключить дискретную ссылку.
ALT-Q	Остановить режим Teach.	F11 или Keypad *	Аннулировать дискретную ссылку.
ALT-n	Проиграть файл n (n = от 0 до 9).		
Быстрые клавиши доступные только в режиме Program Editor			
ALT-B	Переключить текстовый редактор.	Keypad +	Принять шаг.
ALT-D	Удалить элемент шага/удалить шаг.	Enter	Принять шаг.
ALT-S	Сохранить блок в ПЛК и на диске.	CTRL-PgUp	Предыдущий шаг.
ALT-X	Уровень отображения.	CTRL-PgDn	Следующий шаг.
ALT-U	Обновить на диске.	~	Передвинуть по горизонтали.
ALT-V	Окно таблицы переменных.	|	Передвинуть по вертикали.
ALT-F2	Перейти к операнду таблице ссылок.	Tab	Перейти к следующему полю операнда.
Специальные клавиши			
ALT-O	Пароль аннулируется. Доступно только в экране Password в конфигурации		

Приложение

E

Использование чисел с плавающей точкой

В настоящем приложении приведены сведения о том, как использовать числа с плавающей точкой. Первая часть посвящена основным понятиям. Перейдите к странице E-5 и далее для просмотра инструкций по вводу и отображению чисел с плавающей точкой.

Примечание

Числа с плавающей точкой поддерживаются ***только*** в моделях ЦП 35x и 36x, Версии 9 или старше, и во всех версиях модуля CPU352.

Числа с плавающей точкой

ПО для программирования ПЛК обеспечивает возможности редактирования, отображения, хранения и обработки чисел с плавающей точкой. Некоторые функции оперируют числами с плавающей точкой. Однако, при использовании таких чисел в ПО для программирования, Вы должны иметь модуль ЦП 35x или 36x (см. примечание выше). Числа с плавающей точкой представлены в экспоненциальном виде, с отображением шести значащих цифр.

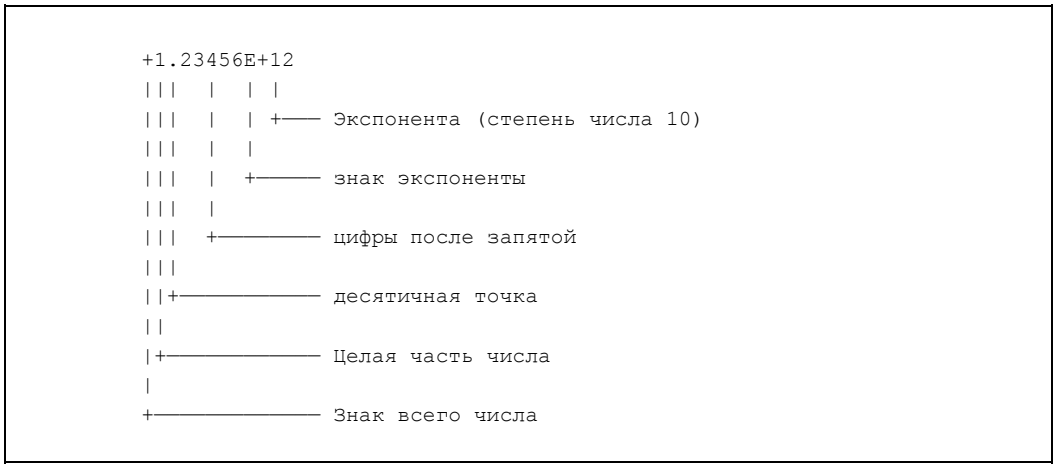
Примечание

В этом руководстве, термины “плавающая точка” и “вещественное число” взаимозаменяют друг друга при описании чисел с плавающей точкой в ПО для программирования.

Рассмотрим используемые форматы представления. Число в диапазоне от 0.0001 до 9999999, отображается без экспоненты и до шести или семи цифр после запятой. Например:

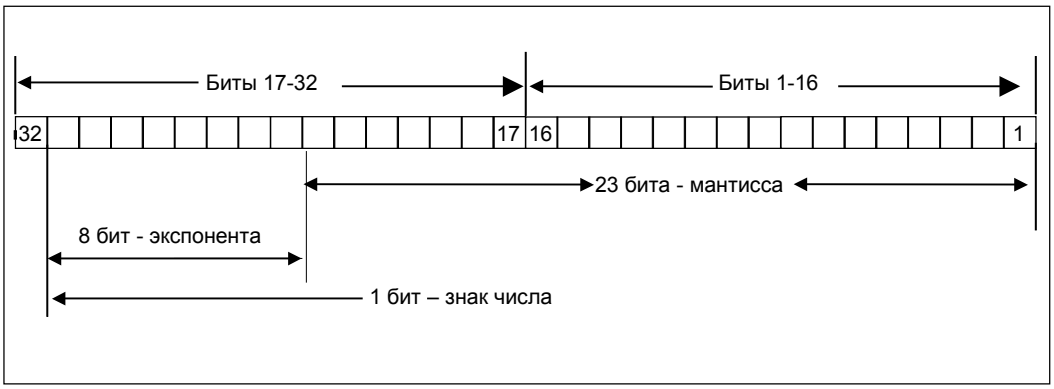
Введенное	Отображаемое	Описание
.000123456789	+.0001234567	10 цифр, шесть или семь после запятой.
–12.345e-2	–.1234500	7 цифр, шесть или семь после запятой.
1234	+1234.000	7 цифр, шесть или семь после запятой.

С другой стороны в диапазоне, представленном выше, только шесть цифр после запятой, отображается на дисплее в следующем виде:

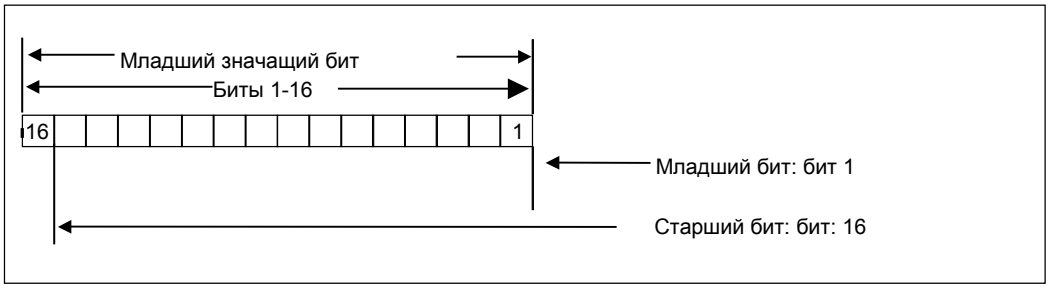


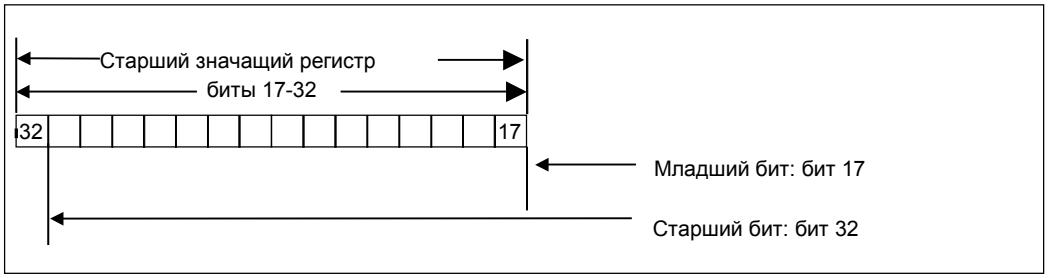
Внутренний формат представления чисел с плавающей точкой

Числа с плавающей точкой хранятся в формате стандарта IEEE одинарной точности. Этот формат использует 32 бита, которые занимают два 16-битных регистра ПЛК. Значения этих бит, приведены на рисунке ниже:



Ниже на диаграмме показано использование регистров при хранении одного числа с плавающей точкой. На этой диаграмме, если вещественное число занимает, например, регистры R5 и R6, регистр R5 - младший регистр и регистр R6 - старший.





Значения чисел с плавающей точкой

Используя следующую таблицу для определения значения числа с плавающей точкой из двоичных чисел, хранящихся в двух регистрах.

Экспонента (e)	Мантисса (f)	Значение числа
255	Ненулевая	Неправильное число (NaN).
255	0	$-1^s * \infty$
0 < e < 255	Любое значение	$-1^s * 2^{e-127} * 1.f$
0	Ненулевая	$-1^s * 2^{-126} * 0.f$
0	0	0

f = это мантисса. Мантисса это дробная двоичная часть.
e = это экспонента. То есть целое число E, такое как E+127, число в степени 2, которое умножается с мантиссой для получения значения числа.
s = знаковый бит.
* = умножение.

Например, рассмотрим вещественное число 12.5. В формате IEEE в двоичном виде оно будет представлено следующим образом:

```
01000001 01001000 00000000 00000000
```

или 41480000 в шестнадцатеричной системе счисления. Старший бит (знаковый бит) равен нулю (s=0). Следующие восемь значащих бит это 10000010, или 130 в десятичной системе счисления (e=130).

Мантисса хранится как двоично-десятичное число с десятичной точкой в 23 значащих битах. Таким образом, старший значащий бит в мантиссе умножается на 2^{-1}, следующий бит на 2^{-2}, и так далее до последнего значащего бита, который надо умножить на 2^{-23}. Последние 23 бита (мантисса) это:

```
1001000 00000000
```

значение мантиссы, далее, есть 0.5625.

Так как e > 0 и e < 255, мы используем третью формулу из таблицы:

$$\begin{aligned} \text{число} &= -1^s * 2^{e-127} * 1.f \\ &= -1^0 * 2^{130-127} * 1.5625 \\ &= 1 * 2^3 * 1.5625 \\ &= 8 * 1.5625 \\ &= 12.5 \end{aligned}$$

Диапазон чисел, которые можно хранить в этом формате, ограничен от ± 1.401298E–45 до ± 3.402823E+38 и нуль.

Ввод и отображение чисел с плавающей точкой

В мантиссе, можно вводить и сохранять шести- или семизначные числа; однако, ПО для программирования будет отображать только первых шесть цифр. Мантисса может иметь положительный или отрицательный знак. Если знака нет, число с плавающей точкой считается положительным.

Если используется экспонента, можно использовать букву **E** или **e**, и мантисса должна содержать десятичную точку: это освобождает от ошибки в шестнадцатеричном представлении числа. Экспонента может содержать знак; но, если его нет, это считается как положительное число. Если экспонента равна нулю, то допускается нулевое значение. При вводе числа с плавающей точкой пробелы недопустимы.

Для обеспечения простоты использования в командной строке и при вводе в поле, поддерживаются несколько общепринятых форматов. Эти форматы включают в себя также целые числа, десятичные дроби, или десятичные дроби с экспонентой. Эти числа переводятся в стандартный вид на дисплее, после того, как пользователь введет данные и нажмет клавишу **Enter**.

Примеры правильного ввода чисел с плавающей точкой и приведенных к нормализованному виду приведены ниже.

Введено	Отображено
250	+250,0000
+4	+4.000000
–2383019	–2383019.
34.	+34.00000
–.0036209	–.003620900
12.E+9	+1.20000E+10
–.0004E–11	–4.00000E–15
731.0388	+731.0388
99.20003e–29	+9.92000E–28

Примеры ошибочного ввода вещественных чисел приведено ниже.

Ошибочное значение	Пояснение
–433E23	Отсутствует десятичная точка
10e-19	Отсутствует десятичная точка
10.e19	Мантисса не может содержать пробелы между цифрами или символами. Это распространяется на 10.e0, и отобразится сообщение об ошибке.
4.1e19	Мантисса не может содержать пробелы между цифрами или символами. Это распространяется на 10.e0, и отобразится сообщение об ошибке.

Ошибки в числах с плавающей точкой и операциях над ними

В модуле ЦП 352 переполнение возникает, когда число, полученное в результате вычислений функций, превышает значение 3.402823E+38 или становится меньше чем -3.402823E+38. Все другие модели ЦП серии 90-30, которые поддерживают операции с плавающей точкой, этот диапазон представлен больше чем 2^{16} или меньше чем -2^{16}. когда Ваше число превышает этот диапазон, выход функции «OK» устанавливается в ноль, (то есть выключен); и результат устанавливается в положительную бесконечность (для чисел больше чем 3.402823E+38 в модуле ЦП 352 или 2^{16} для всех других модулей) или в отрицательную бесконечность (для чисел меньше чем –3.402823E+38 или -2^{16} для всех остальных модулей). Вы можете установить, когда это произошло, по состоянию в направлении выхода «OK».

POS_INF	= 7F800000h	– в формате IEEE положительная бесконечность в шестнадцатеричном представлении.
NEG_INF	= FF800000h	– в формате IEEE отрицательная бесконечность в шестнадцатеричном представлении.

Примечание

Если используется программная поддержка плавающей точки (все модели, поддерживающие операции с плавающей точкой, кроме модели CPU352), числа округляются до нуля (0) на значениях ±1.175494E–38.

Если бесконечности вырабатываются при переполнении при использовании функций REAL, они могут служить причиной неопределенного результата. Этот неопределенный результат передается как NaN (не число). Например, результат сложения положительной бесконечности с отрицательной бесконечностью будет неопределен. Когда функция ADD_REAL будет осуществляться с операндами положительной бесконечностью и отрицательной бесконечностью, результатом будет NaN.

В модуле ЦП 352, любая функция REAL поддерживает результат NaN, вырабатывая специальные коды NaN, которые распознаются функцией:

NaN_SW	= FFFFFFFFh	– NaN ошибка программного обеспечения
NaN_ADD.	= 7F81FFFFh	– ошибка сложения вещественных чисел в шестнадцатеричном формате.
NaN_SUB	= 7F81FFFFh	– ошибка вычитания вещественных чисел в шестнадцатеричном формате.
NaN_MUL	= 7F82FFFFh	– ошибка умножения вещественных чисел в шестнадцатеричном формате.
NaN_DIV	= 7F83FFFFh	– ошибка деления вещественных чисел в шестнадцатеричном формате.
NaN_SQRT	= 7F84FFFFh	– ошибка извлечения квадратного корня вещественного числа в шестнадцатеричном формате.
NaN_LOG	= 7F85FFFFh	– ошибка вычисления логарифма вещественного числа в шестнадцатеричном формате.
NaN_POW0	= 7F86FFFFh	– ошибка вычисления экспоненты вещественного числа в шестнадцатеричном формате.

NaN_SIN	= 7F87FFFFh	– ошибка вычисления синуса вещественного числа в шестнадцатеричном формате.
NaN_COS	= 7F88FFFFh	– ошибка вычисления косинуса вещественного числа в шестнадцатеричном формате.
NaN_TAN	= 7F89FFFFh	– ошибка вычисления тангенса вещественного числа в шестнадцатеричном формате.
NaN_ASIN	= 7F8AFFFFh	– ошибка вычисления арксинуса вещественного числа в шестнадцатеричном формате.
NaN_ACOS	= 7F8BFFFFh	– ошибка вычисления арккосинуса вещественного числа в шестнадцатеричном формате.
NaN_BCD	= 7F8CFFFFh	– ошибка вещественного числа в формате BCD-4.
REAL_INDEF	= FFC00000h	– ошибка деления 0 на 0.

Все другие модули ЦП, которые поддерживают операции с вещественными числами, имеют один формат NaN: FFFF FFFF.

Когда результат NaN поступает на вход другой функции, он передается на ее выход. Например, если NaN_ADD это первый операнд для функции SUB_REAL, то результатом работы функции SUB_REAL будет значение NaN_ADD. Если оба операнда в функции будут NaN, первый операнд передается на выход. Так как эта особенность распространяется через функции, вы можете идентифицировать функцию, которая выдала NaN.

Примечание

Если в результате операции получено значение NaN, выход функции «ОК» будет установлен в OFF.

Следующая таблица объясняет, когда поток энергии проходит на выход функций ADD, MUL, и тому подобных, при значениях операндов вне диапазона, либо при значениях выхода функции вне диапазона. Как было показано ранее, выходы, которые превышают положительные или отрицательные пределы принимаются равными POS_INF или NEG_INF соответственно.

Таблица E-1. Прохождение потока энергии при операциях с вещественными числами

Операция	Вход 1	Вход 2	Выход	Поток энергии
Все	Число	Число	Положительная или отрицательная бесконечность	Нет
Все ошибки деления	бесконечность	число	бесконечность	Да
Все	Число	Бесконечность	Бесконечность	Да
Деление	Бесконечность	Число	Бесконечность	Нет
Все	Число	Число	NaN	Нет

П

Р

С

T

У

Ф