

Основы программирования CPU S7-200

5

Прежде чем начинать программирование приложений для Вашего CPU S7-200, Вы должны познакомиться с некоторыми важными функциональными возможностями CPU.

Обзор главы

Раздел	Описание	Страница
5.1	Указания по разработке решения задачи автоматизации с применением микро-ПЛК	5-2
5.2	Программа S7-200	5-4
5.3	Языки программирования S7-200	5-5
5.4	Основные элементы для разработки программы	5-8
5.5	Цикл CPU	5-10
5.6	Установка режима работы CPU	5-12
5.7	Установка пароля для CPU	5-13
5.8	Тестирование и контроль программы	5-15
5.9	Устранение ошибок в CPU S7-200	5-18

5.1 Указания по разработке решения по автоматизации с применением микро-ПЛК

Существует много методов проектирования системы автоматизации. Данный раздел объясняет некоторые основные правила, которые Вы можете применять в любом проекте. При этом, естественно, Вы должны следовать технологическим директивам, действующим на Вашем предприятии и учитывать Ваш собственный опыт. На рис. 5-1 показаны некоторые из важных шагов при проектировании системы автоматизации.

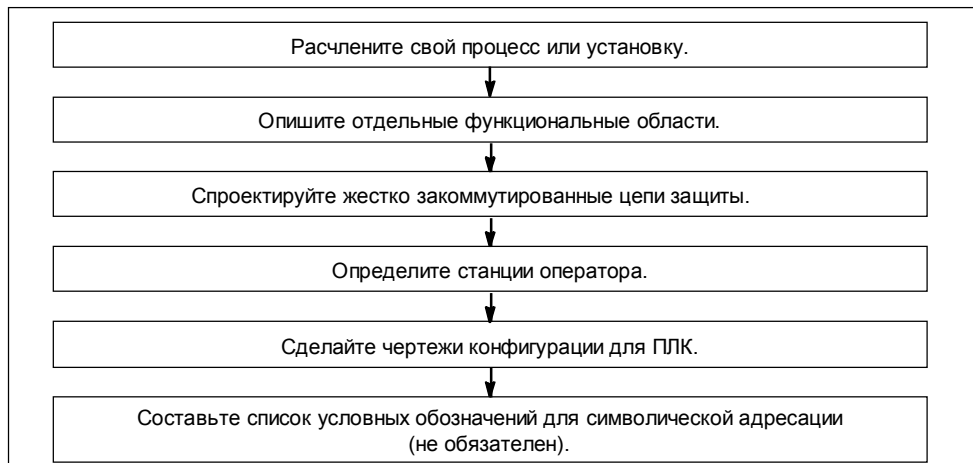


Рис. 5-1. Основные шаги при проектировании системы автоматизации

Расчленение процесса или установки

Расчлените Ваш процесс или Вашу установку на части, не зависящие друг от друга. Такие части устанавливают границы между несколькими системами автоматизации и влияют на описание функциональных областей, а также на распределение оборудования.

Описание функциональных областей

Опишите принцип действия каждой части процесса или установки. Укажите при этом следующие пункты:

- входы/выходы (I/O)
- описание принципа действия
- условия разблокировки (то есть состояния, которые должны достигаться, прежде чем становится возможным управление) для каждого исполнительного устройства (магнитные вентили, электродвигатели, приводы и т.д.)
- описание системы управления и контроля
- интерфейсы с другими частями процесса или установки.

Проектирование цепей защиты

Определите устройства, которые по соображениям безопасности нуждаются в жестко закоммутированных схемах. Управляющие устройства могут принимать опасные рабочие состояния, что может привести к неожиданному пуску или изменению процессов функционирования установки. Если существует опасность того, что при неожиданной или ошибочной работе установки приведет к тяжелым телесным повреждениям или материальному ущербу, то Вы должны предотвратить опасные рабочие состояния с помощью электромеханических средств вмешательства в программу, работающих независимо от CPU.

Для проектирования цепей защиты действуйте следующим образом:

- Выявите ненадлежащую или неожиданную работу исполнительных устройств, которая заключает в себе потенциальную опасность.
- Определите условия, при которых работа является безопасной, и установите, как эти условия распознаются независимо от CPU.
- Определите, как CPU и модули расширения влияют на процесс, когда включается и снова выключается напряжение и когда обнаруживаются ошибки. Такая информация должна использоваться только при проектировании нормальных и ожидаемых ненормальных режимов работы и не должна вводиться по соображениям безопасности.
- Спроектируйте коррекцию через ручное вмешательство или электромеханические средства вмешательства в программу, с помощью которых опасные процессы блокируются независимо от CPU.
- Предоставьте возможность передавать статусную информацию от независимых электрических цепей в CPU, так чтобы программа и каждый интерфейс оператора располагал требуемой информацией.
- Определите другие требования к безопасности, чтобы процесс мог протекать надежно.

Определение станций оператора

Составьте планы станций оператора на основании требований из описаний функциональных областей. Включите следующие пункты:

- Расположение всех станций оператора относительно процесса или установки.
- Механическая компоновка устройств станции оператора (дисплей, переключатели, лампы и т.д.)
- Схемы проводных соединений с соответствующими входами и выходами CPU или модулей расширения.

Вычерчивание конфигурационных планов для ПЛК

Составьте конфигурационные планы системы автоматизации на основании требований из описаний функциональных областей. Включите следующие пункты:

- Обзор расположения всех CPU относительно процесса или установки.
- Механическая компоновка CPU и модулей расширения (включая шкафы и т.д.)
- Схемы проводных соединений для всех CPU и модулей расширения (включая номера устройств, коммуникационные адреса и адреса входов и выходов).

Составление списка символических имен

Если Вы выбираете символическую адресацию, то Вы должны поставить в соответствие абсолютным адресам символические имена. Задайте не только физические входы и выходы, но также и все элементы, используемые в Вашей программе.

5.2 Программа S7-200

Ссылки в программе на входы и выходы

Основной принцип действия CPU S7-200 очень прост:

- CPU считывает состояния сигналов входов.
- Программа, записанная в память CPU, используя эти входы, анализирует логику управления. Во время обработки программы CPU обновляет данные.
- CPU записывает данные на выходы.

На рис. 5-2 показана связь между простой коммутационной схемой и CPU S7-200. В этом примере состояние сигнала переключателя на станции оператора, который открывает сток, складывается с состояниями других входов. Затем итоги вычисления этих состояний определяют состояние сигнала выхода для магнитного вентиля, закрывающего сток. CPU циклически обрабатывает программу, которая считывает и записывает данные.

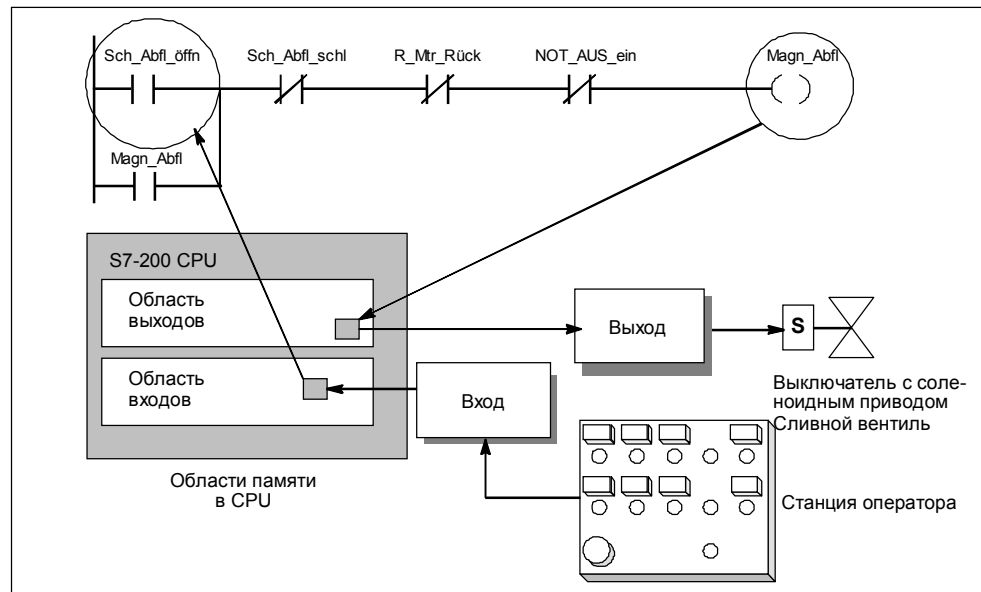


Рис. 5-2. Ссылки в программе на входы и выходы

Доступ к данным в областях памяти

CPU сохраняет состояния сигналов входов и выходов в определенных областях памяти. На рис. 5-2 показан упрощенный информационный поток: Вход ⇒ Область памяти ⇒ Программа ⇒ Область памяти ⇒ Выход. Каждой области памяти поставлен в соответствие мнемонический идентификатор (в частности, "E" для входа и "A" для выхода), через который можно производить доступ к данным в соответствующей области памяти.

STEP 7-Micro/WIN предоставляет в распоряжение "абсолютные" адреса для всех областей памяти. Вы обращаетесь к определенным адресам, задавая операнд (например, E0.0 для первого входа). В STEP 7-Micro/WIN Вы можете также сопоставить абсолютным адресам символические адреса. Абсолютный адрес области памяти состоит не только из признака области (например, "V"), но также из указателя размера данных (максимум 4 байта или 32 бита), к которым нужно произвести доступ: B (байт), W (слово или два байта) или D (двойное слово 4 байта). Кроме того, абсолютный адрес включает в себя числовое значение. Это либо количество байтов от начала области памяти (смещение) или номер элемента (это значение ориентируется на идентификатор области, см. раздел 6.1).

5.3 Языки программирования S7–200

CPU S7–200 (и STEP 7–Micro/WIN) поддерживает следующие языки программирования:

- Список команд (AWL) состоит из нескольких операций, мнемоника которых представляет функцию CPU.
- Контактный план (KOP) является графическим языком программирования, который похож на электрические схемы.

Кроме того, STEP 7–Micro/WIN предоставляет два способа отображения адресов и операций в программе: международный и SIMATIC. Оба способа представления – международный и SIMATIC – относятся к одному и тому же набору операций для S7–200. Между этими двумя способами представления существует прямое соответствие, и функциональные возможности обоих способов представления идентичны.

Основные элементы контактного плана

Если Вы проектируете программу в форме контактного плана, то Вы работаете с графическими компонентами, с помощью которых строятся логические сети. Для проектирования программы Вы можете использовать следующие элементы

□ см. рис. 5–3):

- Контакты: Каждый контакт представляет собой переключатель, через который при замкнутом состоянии может протекать ток.
- Катушки: Каждая катушка представляет собой реле, которое включается при протекании тока.
- Блоки: Каждый блок представляет собой функцию, которая выполняется, когда к блоку течет ток.
- Сети: Сеть образует полную цепь тока. Ток течет от левой шины тока через замкнутые контакты к катушкам или блокам, которые за счет этого активизируются.

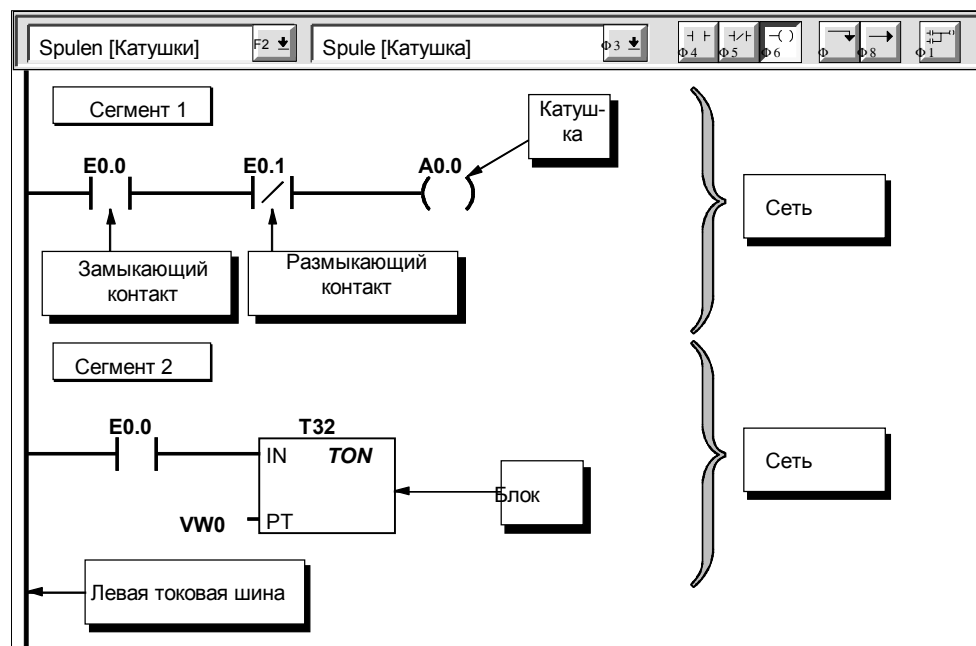


Рис. 5-3. Основные элементы контактного плана

Операторы списка команд

Список команд представляет собой язык программирования, в котором каждая команда Вашей программы содержит операцию, мнемоника которой представляет функцию CPU. Вы связываете эти операции в одну программу так, чтобы создать систему управления Вашим приложением.

На рис. 5-4 показаны основные элементы программы в форме списка команд.

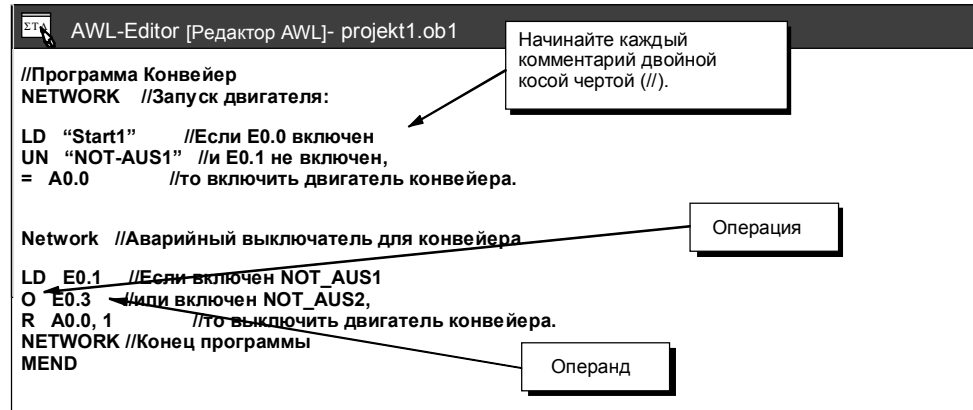


Рис. 5-4. Окно редактора AWL с примером программы

Для решения логических задач операции AWL работают с логическим стеком в CPU. Этот логический стек имеет глубину девять битов и разрядность один бит (см. рис. 5-5). Большинство операций AWL работает либо с первым битом, либо с первым и вторым битами стека. Новые значения могут вдвигаться (или добавляться) в стек. Если два самых верхних бита стека логически сопрягаются, то стек уменьшается на один бит.

Биты логического стека	S0	Stack 0 - Первый уровень стека или вершина стека
	S1	Stack 1 - Второй уровень стека
	S2	Stack 2 - Третий уровень стека
	S3	Stack 3 - Четвертый уровень стека
	S4	Stack 4 - Пятый уровень стека
	S5	Stack 5 - Шестой уровень стека
	S6	Stack 6 - Седьмой уровень стека
	S7	Stack 7 - Восьмой уровень стека
	S8	Stack 8 - Девятый уровень стека

Рис. 5-5. Логический стек CPU S7-200

В то время, как большинство операций AWL только считывают значение из стека, некоторые операции AWL изменяют сохраненные в стеке значения. На рис. 5–6 показаны три примера работы со стеком некоторых операций. В данном примере надписи "aw0" – "aw8" обозначают выходы логического стека, "nw" обозначает новое значение, подготавливаемое операцией, и S0 обозначает рассчитанное значение, которое сохраняется в логическом стеке. В примерах используются следующие булевы операторы логического сопряжения: UND (*) и ODER (+).

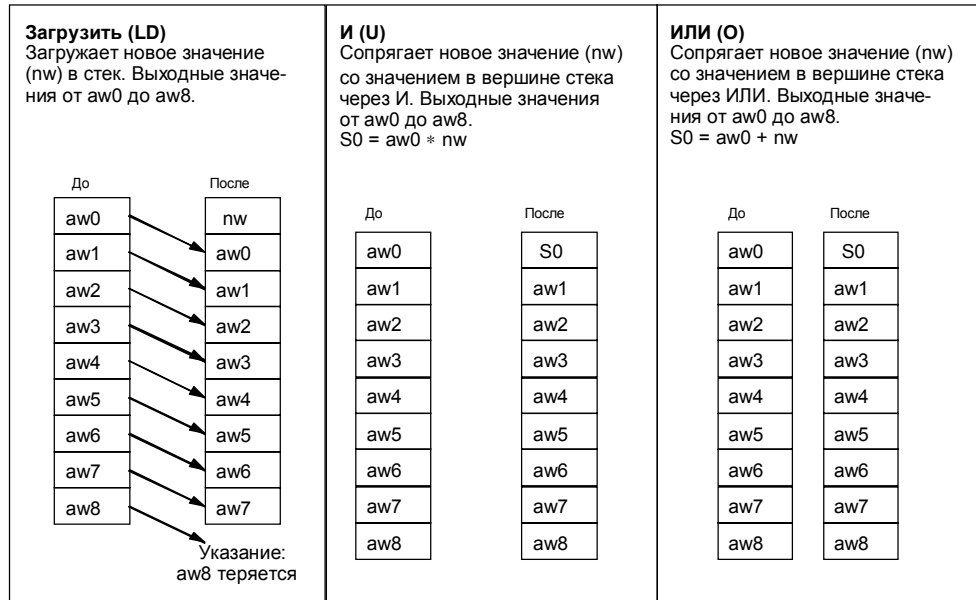


Рис. 5-6. Примеры воздействия операций AWL на логический стек

5.4 Основные элементы для разработки программы

CPU S7–200 непрерывно обрабатывает Вашу программу, чтобы управлять задачей или процессом. Вы разрабатываете программу с помощью STEP 7–Micro/WIN и затем загружаете ее в CPU. Вы можете вызывать из главной программы различные подпрограммы и программы обработки прерываний.

Структуризация программы

Программы для CPU S7–200 состояются из трех основных элементов: главная программа, подпрограммы (необязательные) и программы обработки прерываний (необязательные). Программа для CPU S7–200 подразделяется на следующие организационные единицы (см. рис. 5–7):

- Главная программа: В этой основной части программы располагаются операции, управляющие Вашим приложением. Операции главной программы в каждом цикле обрабатываются последовательно. Для окончания главной программы в форме KOP используется катушка абсолютного завершения программы, а в форме AWL - операция окончания главной программы (MEND) (см. (1) на рисунке 5–7).
- Подпрограммы: Эти необязательные компоненты программы обрабатываются только тогда, когда они вызываются из главной программы. Располагайте подпрограммы после главной программы (после катушки абсолютного завершения в KOP или после операции MEND в AWL). Завершайте каждую подпрограмму командой RET, "вернуться" (см. (2) на рисунке 5–7).
- Программы обработки прерываний: Эти необязательные компоненты программы обрабатываются только тогда, когда появляется событие прерывания. Располагайте программы обработки прерываний после главной программы (после катушки абсолютного завершения в KOP или после операции MEND в AWL). Завершайте каждую программу обработки прерываний операцией RETI, "вернуться из программы обработки прерываний" (см. (3) на рисунке 5–8).

Подпрограммы и программы обработки прерываний следуют за катушкой абсолютного завершения в KOP или за операцией MEND в AWL в главной программе. Других указаний, которые Вы должны соблюдать при размещении подпрограмм и программ обработки прерываний в Вашей программе, не существует. Вы можете располагать подпрограммы и программы обработки прерываний после главной программы в смешанной последовательности. Однако если Вы хотите строить Вашу программу в легко читаемой и понятной форме, то Вы должны присоединить все подпрограммы непосредственно к главной программе, а затем расположить все программы обработки прерываний вслед за подпрограммами.

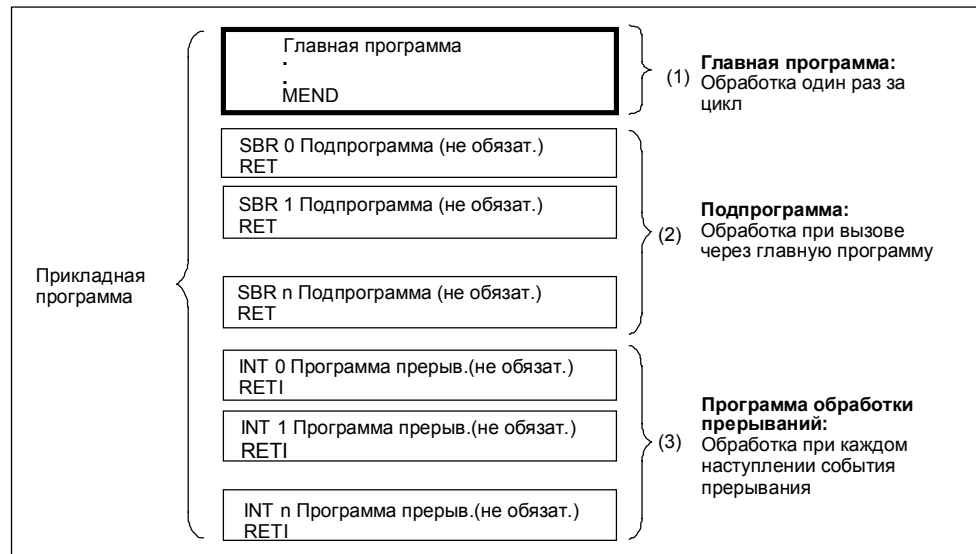


Рис. 5-7. Структура программы для CPU S7-200

Пример программы с подпрограммами и программами обработки прерываний

На рис. 5–8 показан пример программы управляемого временем прерывания, с помощью которого Вы можете считывать значение аналогового входа. В этом примере аналоговый вход опрашивается через каждые 100 мс.

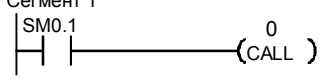
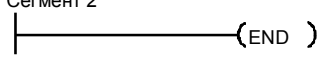
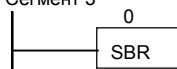
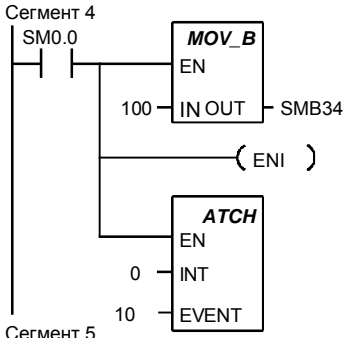
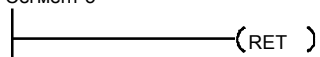

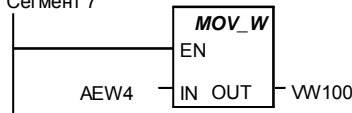
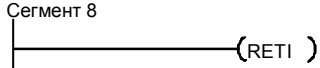
KOP	AWL
Главная программа	
Сегмент 1  Сегмент 2 	Сегмент 1 LD SM0.1 //Меркер первого цикла CALL 0 //Вызов подпрограммы 0 Сегмент 2 MEND
Подпрограмма	
Сегмент 3  Сегмент 4  Сегмент 5 	Сегмент 3 SBR 0 //Начало подпрограммы 0 Сегмент 4 LD SM0.0 //Меркер установлен постоянно MOVB 100, SMB34 //Установка интервала прерывания, управляемого временем, на 100 мс ENI //Разрешить все события прерывания ATCH 0, 10 //Управляемое временем прерывание 0 //поставить в соответствие программе прерываний 0 Сегмент 5 RET //Завершить подпрограмму
Программа обработки прерываний	
Сегмент 6  Сегмент 7  Сегмент 8 	Сегмент 6 INT 0 //Начало программы обработки прерываний 0 Сегмент 7 MOVW AEW4, VW100 //Опрос AEW4 Сегмент 8 RETI //Завершение программы обработки прерываний

Рис. 5-8. Пример программы с подпрограммами и программами обработки прерываний

5.5 Цикл CPU

CPU S7–200 обрабатывает программу циклически. Цикл состоит из нескольких шагов, которые выполняются регулярно и в строгой последовательности. Цикл CPU состоит из следующих задач (см. рис. 5–9):

- Считывание входов
- Обработка программы
- Обработка коммуникационных запросов
- Проведение самодиагностики в CPU
- Запись на выходы

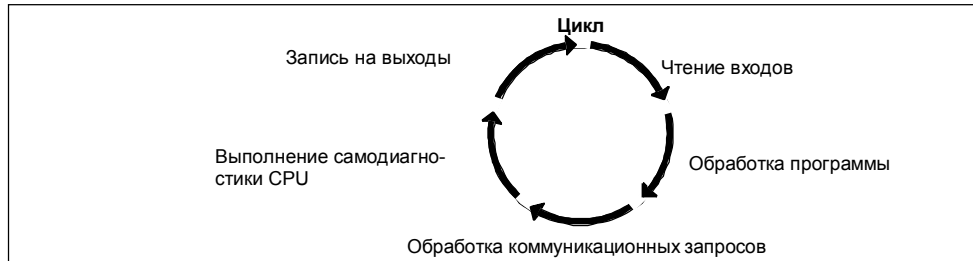


Рис. 5-9. Цикл CPU S7-200

Считывание цифровых входов

В начале цикла считываются текущие значения цифровых входов и затем записываются в область отображения процесса на входах.

В области отображения процесса на входах для CPU предусмотрены разделы по восемь битов (один байт). Если CPU или модуль расширения предоставляет в распоряжение физический вход не для каждого бита зарезервированного байта, то Вы не можете назначать такие биты последующим модулям в цепи ввода/вывода и использовать их в Вашей программе. Свободные входы в области отображения процесса обнуляются CPU в начале каждого цикла. Однако если Ваш CPU может дополняться несколькими модулями расширения и Вы не исчерпали эту возможность полностью (если Вы не установили максимальное количество модулей расширения), то Вы можете использовать свободные входные биты, предусмотренные для модулей расширения, в качестве дополнительных маркеров.

CPU автоматически не обновляет аналоговые входы в качестве части цикла, а также не записывает в память отображение для аналоговых входов. Доступ к аналоговым входам Вы должны производить непосредственно через Вашу программу.

Обработка программы

В этой фазе цикла CPU обрабатывает программу от первой до последней операции. Во время обработки главной программы или программы обработки прерываний Вы можете прямо обращаться к входам и выходам и таким образом управлять ими.

Если Вы в своей программе используете прерывания, то программы обработки прерываний, поставленные в соответствие событиям прерываний, сохраняются как часть главной программы (см. рис. 5.4). Однако программы обработки прерываний не обрабатываются как составная часть цикла, а обрабатываются только тогда, когда появляется событие прерывания (это возможно в любой точке цикла)

Обработка коммуникационных запросов

В этой фазе цикла CPU обрабатывает все сообщения, принятые через коммуникационный порт.

Проведение самодиагностики в CPU

В этой фазе цикла CPU проверяет программы в ПЗУ, память программ и состояние модулей расширения.

Запись на цифровые выходы

В конце цикла значения из области отображения процесса на выходах записываются на цифровые выходы.

В области отображения процесса на выходах для CPU предусмотрены разделы по восемь битов (один байт). Если CPU или модуль расширения предоставляет в распоряжение физический выход не для каждого бита зарезервированного байта, то Вы не можете назначать такие биты последующим модулям в цепи ввода/вывода. Однако Вы можете использовать свободные биты в области отображения процесса на выходах в качестве дополнительных внутренних меркеров.

CPU автоматически не актуализирует аналоговые выходы в качестве части цикла, а также не записывает в память отображение для аналоговых выходов. Доступ к аналоговым выходам Вы должны производить непосредственно через Вашу программу.

Прерывание цикла

Если Вы в своей программе используете прерывания, то программы обработки прерываний, поставленные в соответствие событиям прерываний, запоминаются как часть главной программы. Однако программы обработки прерываний не обрабатываются как составная часть нормального цикла, а обрабатываются только тогда, когда появляется событие прерывания (это возможно в любой точке цикла). CPU обрабатывает разблокированные прерывания асинхронно по отношению к циклу и выполняет программу обработки прерываний, когда появляется соответствующее событие прерывания. Обработка прерываний происходит в порядке их появления и в соответствии с их приоритетом.

Отображения процесса на входах и выходах

Во время обработки программы доступ к входам и выходам происходит не прямо, а через соответствующие области отображения процесса. Имеются три важных основания для существования областей отображения процесса:

- Система в начале цикла опрашивает входы. Благодаря этому, значения этих входов синхронизируются и “замораживаются” на период обработки программы. После обработки программы через область отображения процесса актуализируются выходы. Это оказывает стабилизирующее воздействие на систему.
- Программа может обращаться к области отображения процесса гораздо быстрее, чем непосредственно к входам и выходам. Это ускоряет обработку программы.
- Входы и выходы являются битовыми объектами, доступ к которым должен производиться в битовом формате. К областям же отображения процесса Вы можете обращаться в формате бита, байта, слова и двойного слова. Поэтому области отображения процесса обеспечивают дополнительную гибкость.

Дополнительным преимуществом является то, что области отображения процесса достаточно велики для того, чтобы обрабатывать максимальное количество входов и выходов. Так как реальная система состоит из входов и выходов, то в области отображения процесса всегда есть неиспользуемые адреса. Вы можете использовать эти свободные адреса как дополнительные внутренние меркеры.

Прямое управление входами и выходами

С помощью операций прямого управления входами и выходами Вы можете прямо обращаться к входу или выходу, хотя нормально в качестве источника и цели доступа к входам и выходам используются области отображения процесса. Если Вы обращаетесь прямо к входу, то соответствующий адрес в области отображения процесса на входах не изменяется. Если Вы обращаетесь прямо к выходу, то одновременно актуализируется соответствующий адрес в области отображения процесса на выходах.

5.6 Установка режима работы CPU

CPU S7-200 имеет в своем распоряжении два режима работы:

- STOP: CPU не обрабатывает программу. В режиме STOP Вы можете загружать в CPU программу и конфигурировать CPU.
- RUN: CPU обрабатывает программу. В режиме RUN Вы не можете загружать в CPU программу, а также не можете конфигурировать CPU.

Индикация состояния на лицевой панели CPU указывает текущий режим работы. Если Вы хотите загрузить программу в программную память, то Вы должны перевести CPU в состояние STOP.

Установка режима работы с помощью переключателя режимов работы

С помощью переключателя режимов работы (находится под защитной крышкой на CPU) Вы можете вручную установить режим работы CPU:

- Если переключатель режимов работы установлен в положение TERM, то программное обеспечение (STEP 7-Micro/WIN) может управлять режимами работы CPU.
- Если переключатель режимов работы устанавливается в положение STOP, обработка программы прекращается.
- Если переключатель режимов работы устанавливается в RUN, включается обработка программы.

Если переключатель режимов работы находится в положении STOP или TERM и прерывается подача напряжения питания, то при восстановлении напряжения питания CPU автоматически переходит в режим STOP. Если в тот момент, когда прерывается подача напряжения питания, переключатель режимов работы находится в положении RUN, то при восстановлении напряжения питания CPU автоматически переходит снова в режим работы RUN.

Установка режима работы с помощью STEP 7-Micro/WIN

Вы можете устанавливать режим работы CPU также с помощью STEP 7-Micro/WIN (см. рис. 5-10). Для того, чтобы программное обеспечение могло управлять режимом работы, Вы должны перевести переключатель режимов работы на CPU в положение TERM или RUN.

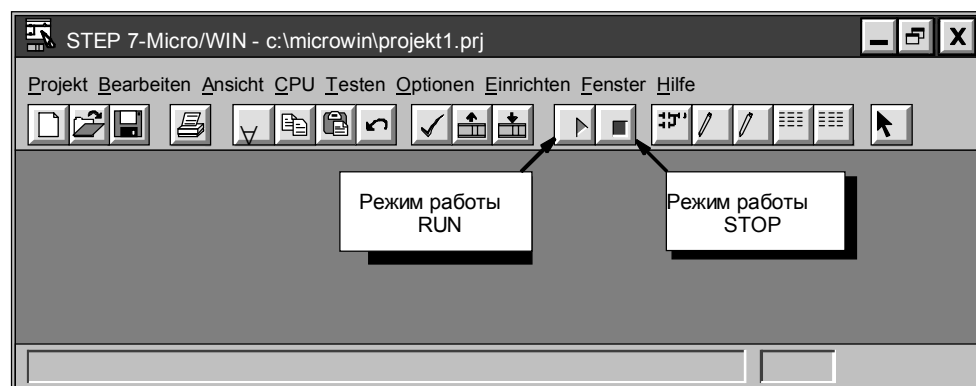


Рис. 5-10. Установка режима работы с помощью STEP 7-Micro/WIN

Установка режима работы с помощью программы

Вы можете в своей программе записать операцию, переводящую CPU в режим работы STOP. Таким способом можно прерывать обработку программы в зависимости от логики программы. Подробную информацию об этой операции Вы найдете в главе 9.

5.7 Установка пароля для CPU

Все варианты CPU S7-200 предоставляют парольную защиту для ограничения доступа к определенным функциям CPU. Благодаря установке пароля доступ к определенным функциям и памяти CPU имеют только уполномоченные лица. При отсутствии пароля CPU неограниченный доступ. Если парольная защита установлена, то CPU запрещает неуполномоченным лицам все операции, ограниченные в конфигурации пароля.

Уровни защиты CPU

CPU S7-200 предоставляют три разных уровня защиты с различными ограничениями доступа к функциям CPU (см. табл. 5-1). Каждый уровень защиты также допускает неограниченный доступ к определенным функциям без ввода пароля. При всех трех уровнях защиты Вы имеете доступ ко всем функциям CPU, если вводите правильный пароль. По умолчанию для CPU S7-200 установлен уровень защиты 1 (ограничений нет).

Если Вы вводите сетевой пароль, то этот пароль не оказывает влияния на парольную защиту CPU. Если некоторый пользователь наделен правом доступа к защищенным функциям CPU, то из этого не следует, что другие пользователи тоже наделены правом доступа к этим функциям. Всегда только один пользователь имеет неограниченный доступ к CPU.

Указание

После того, как Вы ввели пароль, уровень защиты пароля остается действительным после отсоединения устройства программирования от CPU максимум в течение одной минуты.

Таблица 5-1. Уровни защиты в CPU S7-200

Функция	Уровень защиты 1	Уровень защиты 2	Уровень защиты 3
Чтение и запись данных пользователя	Не ограничены	Не ограничены	Не ограничены
Запуск, останов и новый запуск CPU			
Чтение и установка часов реального времени			
Чтение принудительно установленных данных CPU		Нужен пароль	
Загрузка программы пользователя, данных и конфигурации из CPU			
Загрузка в CPU		Нужен пароль	
Стирание программы пользователя, данных и конфигурации ¹			
Принудительная установка данных и выполнение определенного количества циклов			
Копирование в модуль памяти			
¹ Защита от стирания может быть преодолена паролем "clearplc".			

Установка пароля

Пароль для CPU устанавливается с помощью STEP 7–Micro/WIN. Выберите команду меню **CPU** → **Konfigurieren** [**CPU** → **Конфигурирование**], а затем вкладку "Paßwort" ["Пароль"] (см. рис. 5–11). Задайте желаемый уровень защиты и подтвердите пароль.

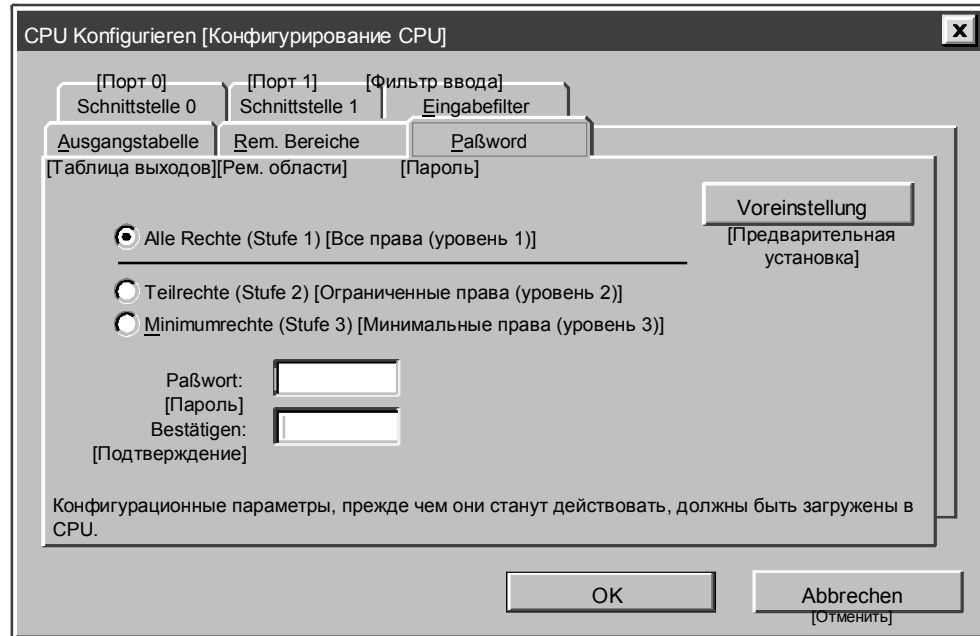


Рис. 5-11. Установка пароля для CPU

Мероприятия в случае забытого пароля

Если Вы забыли свой пароль, то Вы должны произвести общее стирание памяти CPU и снова загрузить в CPU свою программу. При общем стирании памяти CPU последний сначала переводится в состояние STOP, а затем переустанавливается на значения по умолчанию, предварительно установленные предприятием-изготовителем, за исключением адреса абонента и часов реального времени.

Для общего стирания CPU выберите команду меню **CPU** → **Urlöschen** [**CPU** → **Общее стирание**]. В ответ на это отображается диалоговое окно "Urlöschen" ["Общее стирание"]. Выберите опцию "Alles" ["Все"] и подтвердите ввод, щелкнув на "OK". При этом на экране отображается диалоговое окно, в котором запрашивается парольная авторизация. Если Вы введете пароль "clearplc" (общее стирание ПЛК), то Вы можете произвести общее стирание всей памяти CPU.

При общем стирании программа в модуле памяти не стирается. Так как в модуле памяти наряду с программой записан и соответствующий пароль, то Вы должны также перепрограммировать модуль памяти, чтобы стереть забытый пароль.



Предупреждение

При общем стирании CPU выходы выключаются (аналоговые выходы "замораживаются" с определенным значением).

Если при общем стирании CPU S7–200 подключен к оборудованию, то изменения в состоянии выходов могут быть передан на оборудование. Если Вы изменили "безопасные состояния" выходов, предварительно установленные предприятием, то изменения состояний выходов могут вызвать неожиданные реакции со стороны оборудования, что может привести к гибели или серьезным травмам персонала и/или к повреждению оборудования.

Поэтому примите все необходимые меры безопасности и обеспечьте, чтоб Ваш процесс находился в безопасном состоянии, прежде чем производить общее стирание CPU.

5.8 Тестирование и контроль программы

STEP 7–Micro/WIN предоставляет в распоряжение различные инструменты тестирования и контроля программы.

Контроль программы путем выполнения определенного количества циклов

Вы можете указать, что CPU должен обрабатывать Вашу программу в течение определенного количества циклов (от 1 до 65 535 циклов). Если Вы выберете, сколько циклов должен выполнить CPU, то Вы можете наблюдать обработку переменных процесса. Количество циклов, которое должен выполнить CPU, задается с помощью команды меню **Testen** → **Zyklus ausführen** [Тестирование → Выполнение циклов]. На рис. 5–12 показано диалоговое окно, в котором задается количество циклов.

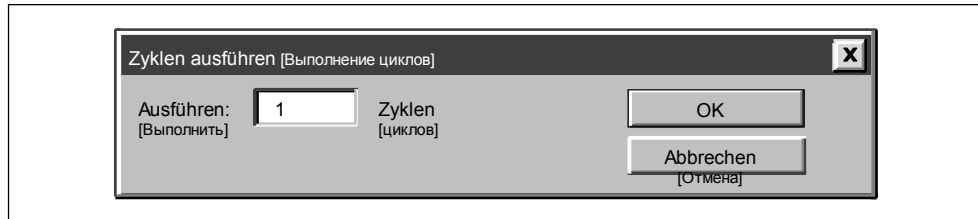


Рис. 5-12. Выполнение программы для определенного числа циклов

Управление и контроль Вашей программы с помощью таблицы состояний/ принудительного задания

С помощью таблицы состояний/принудительного задания можно считывать, записывать, устанавливать и наблюдать переменные в то время, когда обрабатывается программа (см. рис. 5–13). Подробную информацию о создании таблицы состояний Вы найдете в разделе 3.7.

Adresse [Адрес]	Format [Формат]	Aktueller Wert	Wert ändern in
"Start_1"	Binär [двоичный]	2#0	
"Start_2"	Binär	2#0	1
"Stopp_1"	Binär	2#0	
"Stopp_2"	Binär	2#0	
"Behälter_voll"	Binär	2#0	
"Behälter_leer"	Binär	2#0	
"Rücksetzen"	Binär	2#0	
"Pumpe_1"	Binär	2#0	
"Pumpe_2"	Binär	2#0	
"Rührmotor"	Binär	2#0	
"Dampfventil"	Binär	2#0	
"Abflußventil"	Binär	2#0	
"Abflußpumpe"	Binär	2#0	
"Max_Füllstand"	Binär	2#0	
"Mischzeit"	Mit Vorzeichen	+0	
"Zykluszähler"	Mit Vorzeichen	+0	

Рис. 5-13. Управление и визуализация переменных с помощью таблицы состояний/ принудительного задания

Отображение статуса в программе, представленной в форме KOP

С помощью редактора программ STEP 7–Micro/WIN Вы можете контролировать статус программы в режиме (online), причем программа должна отображаться в форме контактного плана (см. рис. 5–14). Так Вы можете наблюдать состояние операций в программе при ее исполнении в CPU.

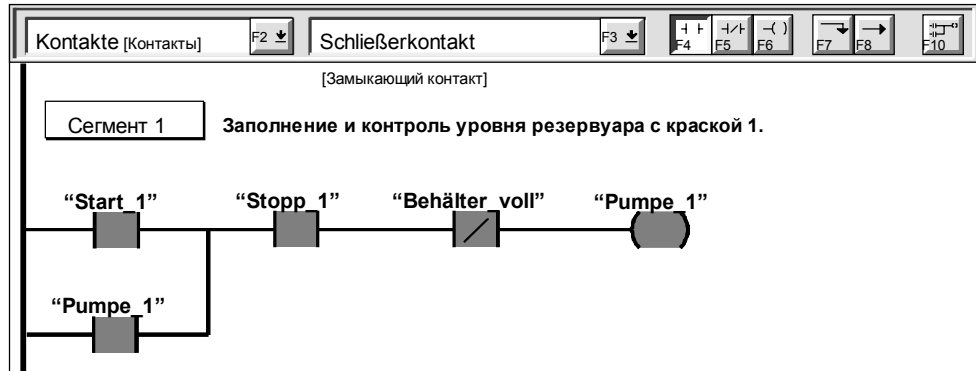


Рис. 5-14. Отображение статуса в программе, представленной в форме KOP

Установка значений с помощью таблицы состояний/принудительного задания

С помощью CPU S7–200 Вы можете некоторые или все входы и выходы биты E и A) принудительно устанавливать на определенные значения. Дополнительно Вы можете принудительно задавать до 16 внутренних меркеров (V или M) либо аналоговых входов или выходов (AE или AA). Значения в памяти переменных и значения меркеров можно задавать в форме байтов, слов и двойных слов. Аналоговые значения могут быть заданы только в форме слов, а именно, по четным байтам например, AEW6 или AAW 14). Все принудительно установленные значения записываются в EEPROM CPU, устойчивую к нулевому напряжению.

Во время выполнения цикла принудительно установленные значения данных могут изменяться (программой, при актуализации входов и выходов или вследствие обработки коммуникаций). Поэтому CPU снова и снова переписывает принудительно установленные значения в различные моменты времени цикла. На рис. 5–15 показано, в каких местах цикла CPU обновляет принудительно установленные переменные.

Функция принудительной установки (Force) подавляет операции прямого считывания или записи входов и выходов. Эта функция подавляет также выход, который был сконфигурирован на установку определенного значения после перехода в состояние STOP. Если CPU переходит в STOP, то выход сохраняет принудительно установленное, а не сконфигурированное значение.

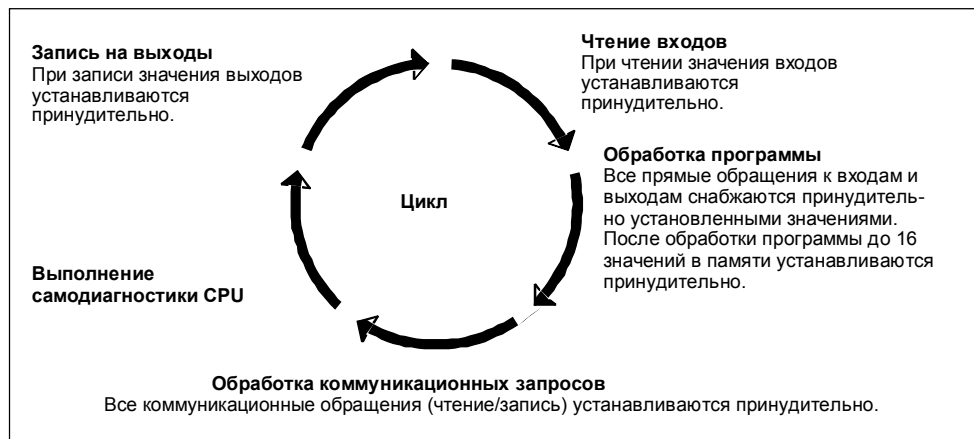


Рис. 5-15. Цикл CPU S7-200

На рис. 5–16 показан пример таблицы состояний/принудительного задания. Подробную информацию о работе с таблицей состояний/принудительного задания Вы найдете в разделе 3.7

Adresse [Адрес]	Format [Формат]	Aktueller Wert	Wert ändern in
"Start_1"	Binär [Двоичный]	2#0	
"Start_2"	Binär	2#0	1
"Stopp_1"	Binär	2#0	
"Stopp_2"	Binär	2#0	
"Behälter_voll"	Binär	2#0	
"Behälter_leer"	Binär	2#0	
"Rücksetzen"	Binär	2#0	
"Pumpe_1"	Binär	2#0	
"Pumpe_2"	Binär	2#0	
"Rührmotor"	Binär	2#0	
"Dampfventil"	Binär	2#0	
"Abflußventil"	Binär	2#0	
"Abflußpumpe"	Binär	2#0 [Со знаком]	
"Max_Füllstand"	Binär	2#0	
"Mischzeit"	Mit Vorzeichen	+0	
"Zykluszähler"	Mit Vorzeichen	+0	

Рис. 5-16. Принудительное задание значений переменным с помощью таблицы состояний/принудительного задания

5.9 Устранение ошибок в CPU S7–200

CPU S7–200 подразделяет встречающиеся ошибки на серьезные (неисправимые) и незначительные (исправимые) ошибки. С помощью STEP 7–Micro/WIN можно отобразить те коды ошибок, которые были порождены встретившимися ошибками. На рис. 5–17 показано диалоговое окно "CPU-Informationen" ["Информация CPU"], в котором отображаются код и описание ошибки. Полный список всех кодов ошибок Вы найдете в приложении С.

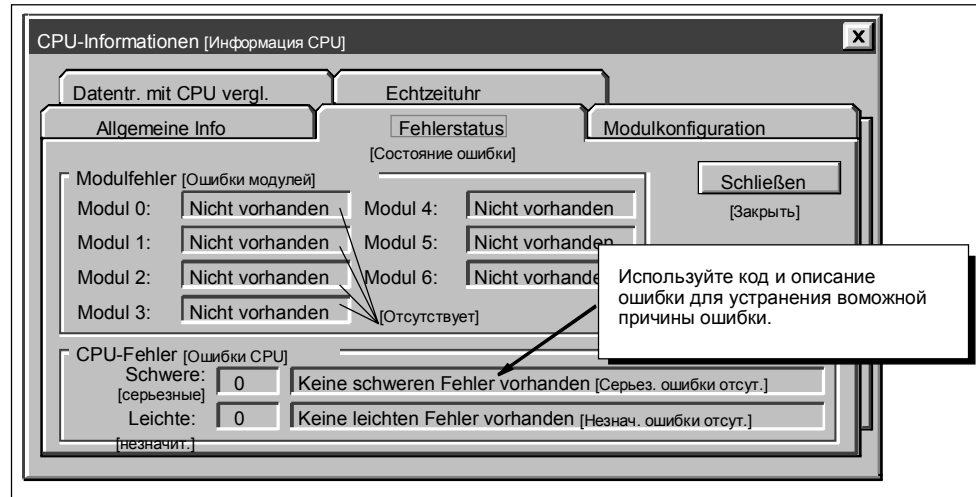


Рис. 5-17. Вкладка "Fehlerstatus" ("Состояние ошибки") в диалоговом окне "CPU-Informationen" ("Информация CPU")

Реакция на серьезные ошибки

Если встречается серьезная ошибка, то CPU прекращает обработку программы. В зависимости от степени серьезности ошибки CPU больше не может выполнять некоторые или даже все функции. Целью обработки серьезных ошибок является перевод CPU в безопасное состояние, так чтобы из CPU могла запрашиваться информация о сбойной ситуации. Если CPU обнаруживает серьезную ошибку, то он переходит в режим STOP, включает светодиодную индикацию серьезной ошибки (SF) и светодиодную индикацию режима STOP и выключает выходы. CPU остается в этом состоянии до тех пор, пока не будет устранена сбойная ситуация.

Если Вы устранили условие возникновения серьезной ошибки, то Вы должны снова запустить CPU. Для этого можно либо выключить и снова включить напряжение питания, либо перевести переключатель режимов работы из положения RUN или TERM в положение STOP. Новый пуск CPU стирает состояние ошибки и при запуске выполняется диагностика для того, чтобы проверить, устранена ли фактически серьезная ошибка. Если при этом обнаруживается еще одна серьезная ошибка, то снова загорается светодиодный индикатор CPU и тем самым указывается на то, что ошибка все еще имеет место. В противном случае CPU начинает свою нормальную работу.

Существует различные возможные сбойные ситуации, которые делают CPU неспособным к коммуникации. В таких случаях Вы не можете отобразить код ошибки CPU. Этот вид ошибки указывает чаще всего на неисправность аппаратуры, которая может быть устранена только путем ремонта CPU. Такие сбойные ситуации не могут быть устранены только путем изменения программы или общего стирания памяти CPU.

Устранение незначительных ошибок

Незначительные ошибки могут частично ограничивать работу CPU. Однако CPU и далее в состоянии обрабатывать программу и актуализировать входы и выходы. Вы можете с помощью STEP 7–Micro/WIN отобразить коды ошибок, которые были порождены незначительными ошибками (см. рис. 5–17). Есть три основные группы незначительных ошибок:

- Ошибки этапа выполнения: все незначительные ошибки, которые обнаруживаются в режиме RUN, отмечаются в специальных меркерах. Ваша программа может контролировать и анализировать эти специальные меркеры. Подробную информацию о специальных меркерах, отображающих незначительные ошибки этапа выполнения, Вы найдете в приложении D.
 При пуске CPU считывает конфигурацию входов и выходов и сохраняет эту информацию в памяти системных данных и в специальных меркерах. При нормальной работе состояние входов и выходов регулярно обновляется и запоминается в специальных меркерах. Если CPU обнаруживает различия в конфигурации входов и выходов, то он устанавливает в байте ошибок модуля бит индикации измененной конфигурации. Потом модуль расширения не актуализируется до тех пор, пока этот бит не будет сброшен. Чтобы CPU мог сбросить этот бит, Вы должны снова согласовать входы и выходы модуля с конфигурацией входов и выходов, записанной в памяти системных данных.
- Ошибки при компиляции программы: CPU компилирует программу при ее загрузке. Если CPU обнаруживает, что программа нарушает некоторое правило компиляции, то он прерывает процесс загрузки и генерирует код ошибки. (Если программа уже была загружена в CPU, то эта программа имеется еще в EEPROM и она не теряется.) После исправления программы Вы можете загружать ее снова.
- Ошибки программирования на этапе выполнения: Вы (или Ваша программа) можете создать сбойную ситуацию во время исполнения программы. Например, косвенный указатель адреса, который при компиляции программы был действительным, во время обработки программы может быть изменен таким образом, что будет указывать на адрес вне допустимой области. Это рассматривается как ошибка программирования на этапе выполнения. Используя диалоговое окно "CPU-Informationen" ["Информация CPU"] (см. рис. 5–17), Вы можете установить, какой вид ошибки встретился.

CPU не переходит в STOP, когда обнаруживается незначительная ошибка. Он отмечает эти события в специальных меркерах (SM) и продолжает обработку программы. Однако Вы можете написать Вашу программу таким образом, что при появлении незначительной ошибки происходит принудительный переход в режим работы STOP. На рис. 5–18 показан сегмент программы, которая контролирует специальный меркер. Операция переводит CPU в режим работы STOP, когда обнаруживается ошибка ввода/ вывода.



Рис. 5-18. Распознавание сбойных ситуаций, связанных с незначительными ошибками, в прикладной программе

