SIEMENS

SIMATIC S7

Программирование с помощью STEP 7 V5.0

Руководство

Это руководство является частью пакета документации с заказным номером:
6ES7 810-4CA04-8BA0

03/99

C79000-G7076-C562

Редакция 02

Указания по безопасности

Это руководство содержит указания, которые вы должны соблюдать для обеспечения собственной безопасности, а также защиты продукта и подключенного оборудования. Эти указания выделены в руководстве предупреждающим треугольником и помечены следующим образом в соответствии с уровнем опасности:

Опасность
Указывает, что несоблюдение надлежащих предосторожностей приведет к смерти, тяжким телесным повреждениям или существенному повреждению имущества.

Предупреждение
Указывает, что несоблюдение надлежащих предосторожностей может привести к смерти, тяжким телесным повреждениям или существенному повреждению имущества.

Предостережение
Указывает, что несоблюдение надлежащих предосторожностей может привести к небольшим телесным повреждениям или порче имущества.

Замечание
Привлекает ваше внимание к особенно важной информации о продукте, обращении с продуктом или к определенной части документации.

Квалифицированный персонал

К установке и работе на данном оборудовании должен допускаться только квалифицированный персонал. К квалифицированному персоналу относятся лица, имеющие право пускать в эксплуатацию, заземлять и маркировать электрические цепи, оборудование и системы в соответствии с установленным порядком и стандартами.

Правильное использование

Примите во внимание следующее:

Предупреждение
Это устройство и его компоненты могут быть использованы только для приложений, описанных в каталоге или технических описаниях, и только в соединении с устройствами или компонентами других производителей, которые были одобрены или рекомендованы фирмой Siemens.

Этот продукт может правильно и безопасно функционировать только при правильной транспортировке, хранении, установке и инсталляции, а также эксплуатации и обслуживании в соответствии с рекомендациями.

Торговые марки

SIMATIC®, SIMATIC HMI® и SIMATIC NET® являются зарегистрированными торговыми марками SIEMENS AG.

Некоторые из других обозначений, использованных в этих документах, также являются зарегистрированными торговыми марками; права собственности могут быть нарушены, если эти обозначения используются третьей стороной для своих собственных целей.

Siemens AG
Департамент техники автоматизации и приводов
Сфера деятельности: промышленные системы автоматизации
п/я 4848, D- 90327 Нюрнберг

Важные замечания

Назначение

Это руководство дает полный обзор программирования с помощью **STEP 7**. Оно разработано, чтобы оказать вам поддержку при установке и вводе в эксплуатацию программного обеспечения. Оно объясняет, как нужно действовать при создании программ, и описывает компоненты программ пользователя.

Руководство предназначено для людей, принимающих участие в выполнении задач управления с использованием STEP 7 и систем автоматизации SIMATIC S7.

Мы рекомендуем вам познакомиться с примерами, приведенными в руководстве "Working with STEP 7 V5.0, Getting Started [Работа со STEP 7 версии 5.0, Введение]". Эти примеры облегчат ваше знакомство с темой "Программирование с помощью STEP 7."

Требуемые основные знания

Для понимания этого руководства требуются общие знания в области технологии автоматизации.

Кроме того, вы должны быть знакомы с использованием компьютеров или PC-подобных инструментальных средств (напр., устройств программирования) с операционной системой Windows 95 / NT или Windows 98.

Область применения руководства

Это руководство действительно для версии 5.0 пакета программного обеспечения STEP 7.

Пакеты документации для STEP 7

Данное руководство является составной частью пакета документации „STEP 7 Basic Information [STEP 7-Основная информация]“. Следующая таблица дает обзор документации для STEP 7:

Документация	Назначение	Номер для заказа
STEP 7 Basic Information [STEP 7-Основная информация], включающее • Руководство Working with STEP 7 V5.0, Getting Started [Работа со STEP 7 версии 5.0, Введение] • Programming with STEP 7 V5.0 [Программирование с помощью STEP 7 V5.0] • Configuring Hardware and Communication Connections, STEP 7 V5.0 [Конфигурирование аппаратуры и проектирование соединений с помощью STEP 7 V5.0] • From S5 to S7, Converter Manual [От S5 к S7. Руководство по конвертированию]	Основная информация для технического персонала, описывающая методы реализации задач управления с помощью STEP 7 и программируемых контроллеров S7-300/400.	6ES7810-4CA04-8BA0
STEP 7 Reference [Справочник по STEP 7], включающий • Руководства по KOP/FUP/AWL для S7-300/400 • Стандартные и системные функции для S7-300/400	Предоставляет справочную информацию и описывает языки программирования KOP, FUP и AWL, а также стандартные и системные функции в дополнение к базовой информации по STEP 7.	6ES7810-4CA04-8BR0

Оперативная помощь в режиме online	Назначение	Номер для заказа
Помощь для STEP 7	Основные сведения по программированию и конфигурированию аппаратуры с использованием STEP 7 как оперативная помощь в режиме online.	Составная часть стандартного программного обеспечения STEP 7.
Справки по AWL/KOP/FUP Справки по SFB/SFC Справки по организационным блокам	Контекстная справка.	Составная часть стандартного программного обеспечения STEP 7.

Помощь в режиме online

Это руководство дополнено оперативной помощью, встроенной в программное обеспечение. Эта оперативная помощь направлена на то, чтобы обеспечить вас детальной поддержкой при использовании данного программного обеспечения.

Система помощи встроена в программное обеспечение через несколько интерфейсов:

- Имеется несколько команд, которые вы можете выбрать в меню **Help [Помощь]**: Команда **Contents [Содержание]** открывает указатель помощи для STEP 7.
- **Using Help [Использование помощи]** дает подробные указания по обращению с оперативной помощью.
- Контекстно-чувствительная помощь предоставляет информацию к текущему контексту, напр., к открытому диалоговому окну или к активному окну. Ее можно вызвать через экранную кнопку ”Help [Помощь]” или с помощью клавиши F1.
- Еще одну форму контекстно-чувствительной помощи предоставляет строка состояния. Для каждой команды меню здесь отображается краткое объяснение, как только указатель мыши оказывается на этой команде.
- Для символов на панели инструментов также высвечивается краткое объяснение, когда указатель мыши находится кратковременно на символе.

Если вы предпочитаете читать информацию из оперативной помощи в напечатанном виде, то вы можете распечатать отдельные темы помощи, книги или же всю помощь

Данное руководство является фрагментом помощи для STEP 7, основанной на HTML. Благодаря почти идентичной структуре деления руководства и оперативной помощи вы можете легко переходить от руководства к оперативной помощи и обратно.

Учебные центры SIMATIC

Фирма Siemens предлагает вам ряд учебных курсов для ознакомления с системой автоматизации SIMATIC S7. Для получения подробной информации обращайтесь, пожалуйста, в свой региональный учебный центр или в центральный учебный центр в Нюрнберге (D 90327 Nuernberg).
Тел.:+49 (911) 895-3154

Горячая линия поддержки пользователей SIMATIC

Доступна по всему миру в любое время суток:

Основная горячая линия SIMATIC

Нюрнберг

Основная горячая линия SIMATIC

Местное время: Пн.-Пт. с 7:00 до 17:00

Телефон: +49 (911) 895–7000

Факс: +49 (911) 895–7002

E–Mail: simatic.support@ nbgm.siemens.de

Время по Гринвичу: +1:00

Платная горячая линия SIMATIC Premium Hotline

(с возмещением расходов, только через SIMATIC Card)

Время: Пн.-Пт. с 0:00 до 24:00

Телефон: +49 (911) 895–7777

Факс: +49 (911) 895–7001

Время по Гринвичу: +01:00

Джонсон-Сити

Основная горячая линия SIMATIC

Местное время: Пн.-Пт. с 8:00 до 17:00

Телефон: +1 423 461–2522

Факс: +1 423 461–2231

E–Mail: simatic.hotline@ sea.siemens.com

Время по Гринвичу: -5:00

Сингапур

Основная горячая линия SIMATIC

Местное время: Пн.-Пт. с 8:30 до 17:30

Телефон: +65 740–7000

Факс: +65 740–7001

E–Mail: simatic@ singnet.com.sg

Время по Гринвичу: +8:00

Службы поддержки пользователей SIMATIC (SIMATIC Customer Support) в режиме online

SIMATIC Customer Support предлагает вам через службы, работающие в режиме online, обширную дополнительную информацию о продуктах SIMATIC:

- Общая текущая информация может быть получена:
  - в **Internet** под http://www.ad.siemens.de/simatic
  - через **Fax-Polling** № 08765-93 02 77 95 00
- Текущие данные о продукте и загрузки, которые вы, возможно, найдете полезными, доступны:
  - в **Internet** под http://www.ad.siemens.de/support/html_00/
  - через **Bulletin Board System** (BBS) в Нюрнберге (*SIMATIC Customer Support Mailbox)* под номером +49 (911) 895-7100.

  Для набора почтового ящика используйте модем с протоколом до V.34 (28,8 кБод), параметры которого установите следующим образом: 8, N, 1, ANSI; или через ISDN (x.75, 64 кБод).

1 Знакомство с продуктом

1.1 Обзор STEP 7

Что такое STEP 7?

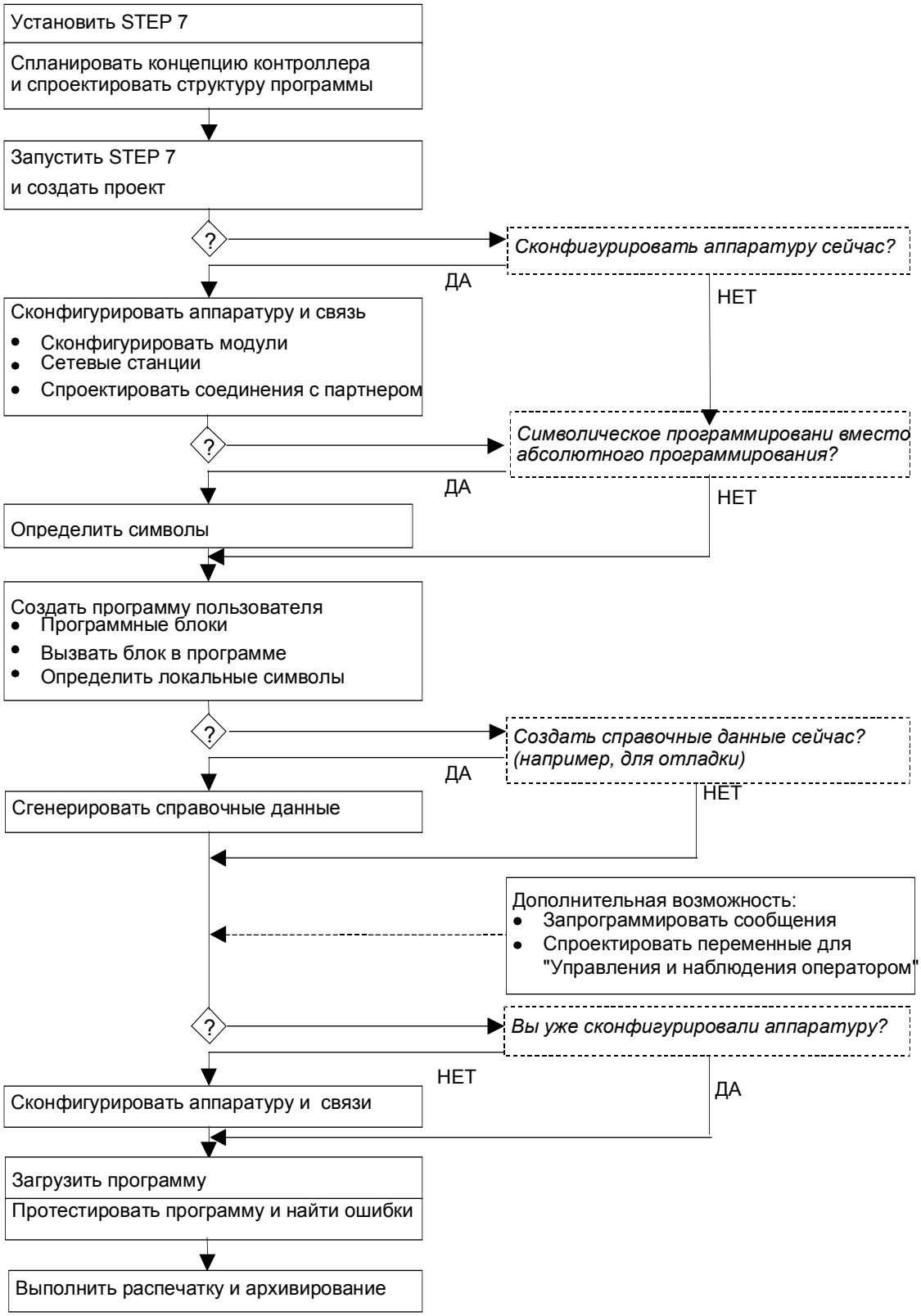
STEP 7 – это пакет стандартного программного обеспечения, используемый для конфигурирования и программирования программируемых логических контроллеров SIMATIC. Он является частью промышленного программного обеспечения SIMATIC. Имеются следующие версии стандартного пакета STEP 7:

- STEP 7 Micro/DOS и STEP 7 Micro/Win для относительно простых автономных приложений на SIMATIC S7-200.
- STEP 7 Mini для относительно простых автономных приложений на SIMATIC S7-300 и SIMATIC C7-620 (см. также Специальные указания для STEP 7 Mini).
- STEP 7 для приложений на SIMATIC S7-300/S7-400, SIMATIC M7-300/M7-400 и SIMATIC C7 с более широким набором функций:
  - Может быть расширен по выбору программными продуктами, имеющимися в промышленном программном обеспечении SIMATIC (см. также Расширенное использование стандартного пакета STEP 7)
  - Возможность назначения параметров функциональным модулям и коммуникационным процессорам
  - Принудительный и многопроцессорный режим
  - Связь через глобальные данные
  - Управляемая событиями передача данных с использованием коммуникационных функциональных блоков
  - Проектирование соединений

STEP 7 и STEP 7 Mini являются предметом обсуждения в данном руководстве, STEP 7 Micro описан в руководстве пользователя "STEP 7 Micro/DOS".

Основные задачи

При решении задачи автоматизации с помощью STEP 7 появляется ряд основных задач. Следующий рисунок показывает задачи, которые должны быть решены для большинства проектов, и ставит им в соответствие базовые процедуры. Он отсылает вас к соответствующей главе, давая вам, таким образом, возможность перемещения по руководству в поисках информации, относящейся к конкретной задаче.


Установить STEP 7
Спланировать концепцию контроллера и спроектировать структуру программы
Запустить STEP 7 и создать проект
?
Сконфигурировать аппаратуру сейчас?
ДА
НЕТ
Сконфигурировать аппаратуру и связь
Сконфигурировать модули
Сетевые станции
Спроектировать соединения с партнером
?
Символическое программировани вместо абсолютного программирования?
ДА
НЕТ
Определить символы
Создать программу пользователя
Программные блоки
Вызвать блок в программе
Определить локальные символы
?
Создать справочные данные сейчас? (например, для отладки)
ДА
НЕТ
Сгенерировать справочные данные
Дополнительная возможность:
Запрограммировать сообщения
Спроектировать переменные для "Управления и наблюдения оператором"
?
Вы уже сконфигурировали аппаратуру?
НЕТ
ДА
Сконфигурировать аппаратуру и связи
Загрузить программу
Протестировать программу и найти ошибки
Выполнить распечатку и архивирование

Альтернативные процедуры

Как показано на предыдущем рисунке, в вашем распоряжении есть две альтернативы:

- Вы можете сначала сконфигурировать аппаратуру, а затем программировать блоки.
- Вы можете, однако, сначала запрограммировать блоки, не конфигурируя аппаратуру. Это рекомендуется для работ, связанных с эксплуатацией и обслуживанием, например, для встраивания программных блоков в существующий проект.

Краткое описание отдельных шагов

- Установка и авторизация
  При первом использовании STEP 7, установите его и перенесите авторизацию с дискеты на жесткий диск (см. также Установка STEP 7 и авторизация).
- Спланируйте концепцию использования вашего контроллера
  Перед началом работы со STEP 7 спланируйте решение задачи автоматизации от деления процесса на отдельные задачи до создания диаграммы конфигурации (см. также Основную последовательность действий при планировании проекта автоматизации).
- Спроектируйте структуру программы
  Преобразуйте задачи, описанные в эскизном проекте вашего контроллера, в структуру программы, используя блоки, имеющиеся в STEP 7 (см. также Блоки в программе пользователя).
- Запустите STEP 7
  STEP 7 запускается из пользовательского интерфейса Windows 95/98/NT (см. также Запуск STEP 7).
- Создайте структуру проекта
  Проект похож на папку, в которой все данные хранятся в виде иерархической структуры и доступны вам в любое время. После создания проекта все остальные задачи выполняются в этом проекте (см. также Структуру проекта).
- Сконфигурируйте станцию
  При конфигурировании станции вы указываете, какой программируемый контроллер вы хотите использовать; например, SIMATIC 300, SIMATIC 400, SIMATIC S5 (см. также Вставка станций).
- Сконфигурируйте аппаратуру
  При конфигурировании аппаратуры вы указываете в конфигурационной таблице, какие модули вы хотите использовать для решения своей задачи автоматизации и какие адреса должны быть использованы для доступа к модулям из программы пользователя. Модулям также могут быть назначены свойства с помощью параметров (см. также Основная последовательность действий при конфигурировании аппаратуры) .
- Спроектируйте сети и коммуникационные связи
  Основой для коммуникаций является предварительно спроектированная сеть. Для этого вам нужно будет создать подсети, необходимые для ваших задач автоматизации, установить свойства подсетей и установить свойства сетевых подключений и всех коммуникационных связей, требуемых для сетевых станций (см. также Последовательность действий при конфигурировании подсети).

- Определите символы
  Вы можете определить в таблице символов локальные или совместно используемые символы, имеющие более наглядные имена, для использования вместо абсолютных адресов в своей пользовательской программе (см. также Создание таблицы символов).
- Создайте программу
  Используя один из доступных языков программирования, создайте программу, связанную с модулем или независимую от модуля, и сохраните ее в виде блоков, исходных файлов или схем (см. также Основную последовательность действий при создании логических блоков и Основную информацию по программированию в исходных файлах на языке AWL).
- Только для S7: сгенерируйте и проанализируйте справочные данные
  Вы можете использовать эти справочные данные для облегчения отладки и модификации программы пользователя (см. также Обзор доступных справочных данных).
- Спроектируйте сообщения
  Сообщения, относящиеся к блокам, создаются, например, с помощью их текстов и атрибутов. Используя передающую программу, вы переносите созданные данные о конфигурации сообщений в базу данных системы взаимодействия с оператором (например, SIMATIC WinCC, SIMATIC ProTool), см. также Проектирование сообщений.
- Спроектируйте переменные для управления и наблюдения оператором
  Вы создаете переменные для управления и наблюдения оператором один раз в STEP 7 и назначаете им требуемые атрибуты. Используя передающую программу, вы переносите созданные переменные для управления и наблюдения оператором в базу данных системы взаимодействия с оператором WinCC (см. также Проектирование переменных для управления и наблюдения оператором).
- Загрузите программы в программируемый контроллер
  Только для S7: после завершения конфигурирования, назначения параметров и программирования задач вы можете загрузить всю свою пользовательскую программу или отдельные боки из нее в программируемый контроллер (программируемый модуль для вашего аппаратного решения). CPU уже содержит операционную систему.
  Только для M7: выберите подходящую операционную систему для решения своей задачи автоматизации из ряда различных операционных систем и перенесите ее отдельно или вместе с программой пользователя на требуемый носитель данных системы программного управления M7.
- Протестируйте программу
  Только для S7: для тестирования или отобразить значения переменных из своей пользовательской программы или CPU, или присвоить значения переменным и создать таблицу для переменных, которые вы хотите отображать или изменять (см. также Введение в тестирование с помощью таблицы переменных).
  Только для M7: протестируйте программу пользователя с помощью средств отладки языка высокого уровня.
- Наблюдайте за работой, диагностируйте аппаратуру
  Причина неисправности модуля определяется отображением информации о модуле в режиме online. Причины ошибок в обработке программы пользователя определяются с помощью диагностического буфера и содержимого стеков. Вы можете также проверить, может ли

программа пользователя исполняться на конкретном CPU (см. также Диагностику аппаратуры и Отображение информации о модуле).

- Задокументируйте установку
  После создания проекта/установки имеет смысл выполнить четкое документирование данных проекта, чтобы облегчить дальнейшее редактирование проекта и любую деятельность по обслуживанию (см. также Печать проектной документации). DOCPRO, дополнительное инструментальное средство для создания и управления документацией на установку, позволяет структурировать данные проекта, представить их в форме руководства по монтажу и распечатать их в обычном формате.

Специализированные темы

При решении задач автоматизации имеется ряд специальных тем, которые могут представлять для вас интерес:

- Мультипроцессорный режим – синхронная работа нескольких CPU (см. также Мультипроцессорный режим – синхронная работа нескольких CPU)
- Работа с проектом нескольких пользователей (см. также Редактирование проектов более чем одним пользователем)
- Работа с системами M7 (см. также Последовательность действий для систем M7)

1.2 Стандартный пакет STEP 7

Используемые стандарты

Языки программирования SIMATIC и встроенные в STEP 7 представления языков соответствуют требованиям стандарта EN 61131-3 или IEC 1131-3. Стандартный пакет работает в операционной системе Windows 95/98/NT и соответствует графической и объектно-ориентированной философии работы Windows.

Функции стандартного пакета

Стандартное программное обеспечение оказывает вам поддержку на всех стадиях процесса решения задачи автоматизации, таких как:

- Создание и управление проектами
- Конфигурирование и назначение параметров аппаратуре и связям
- Управление символами
- Создание программ, например, для программируемых контроллеров S7
- Загрузка программ в программируемые контроллеры
- Тестирование системы автоматизации
- Диагностика неисправностей установки

Пользовательский интерфейс программного пакета STEP 7 спроектирован так, чтобы удовлетворить самым последним достижениям эргономики, и облегчает вам начало работы.

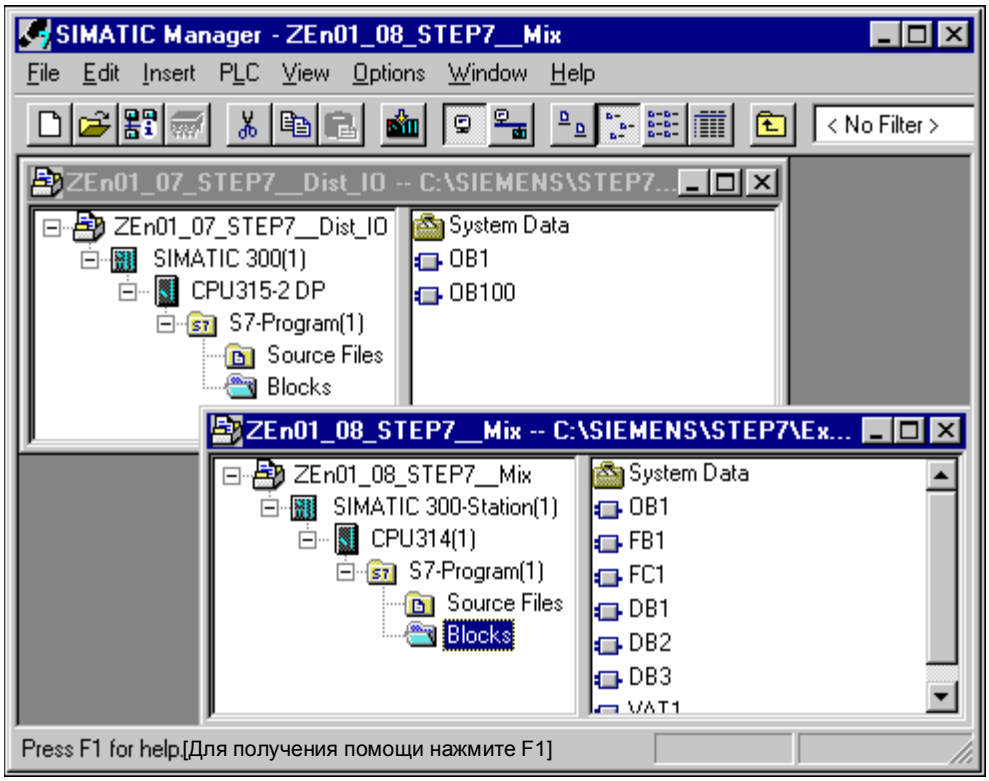
Приложения в STEP 7

Стандартный пакет STEP 7 предоставляет ряд приложений (инструментальных средств) внутри программного пакета:

Вам не нужно открывать инструментальные средства отдельно; они запускаются автоматически при выборе соответствующей функции или открытии объекта.

SIMATIC Manager

SIMATIC Manager управляет всеми данными, относящимися к проекту автоматизации – независимо от того, для какой системы программного управления (S7/M7/C7) они спроектированы. Инструментальные средства, необходимые для редактирования выбранных данных, запускаются автоматически SIMATIC Manager'ом.



Пояснения к рисунку: File – файл, Edit – редактирование, Insert – вставка, PLC – ПЛК (программируемый логический контроллер), View – вид, Options – параметры, Window – окно, Help – помощь, No Filter – фильтр отсутствует, Source Files – исходные файлы, Blocks – блоки, System Data – системные данные

Редактор символов

С помощью редактора символов вы управляете всеми совместно используемыми символами. Доступны следующие функции:

- Установка символических имен и комментариев для сигналов процесса (входов/выходов), меркеров и блоков
- Функции сортировки
- Импорт/экспорт в другие программы/из других программ Windows

Таблица символов, созданная этим инструментальным средством, доступна и всем другим инструментам. Следовательно, любые изменения свойств символа автоматически распознаются всеми инструментальными средствами.

Диагностика аппаратуры

Эти функции предоставляют вам обзор состояния программируемого контроллера. Обзор может отображать символы, чтобы показать, неисправен какой-либо модуль или нет. Двойной щелчок на неисправном модуле отображает подробную информацию о неисправности. Объем этой информации зависит от конкретного модуля:

- Отображение общей информации о модуле (например, номер для заказа, версия, имя) и состояния модуля (например, неисправен)
- Отображение неисправностей модуля (например, неисправность канала) для центрального устройства и slave-устройств DP
- Отображение сообщений из диагностического буфера

Для CPU отображается следующая дополнительная информация:

- Причины неисправностей при обработке программы пользователя
- Отображение длительности цикла (самого длинного, самого короткого и последнего)
- Возможности и загрузка связей через MPI
- Отображение функциональных характеристик (числа возможных входов/выходов, меркеров, счетчиков, таймеров и блоков)

Языки программирования

Языки программирования контактный план, список операторов и функциональный план для S7-300 и S7-400 являются составной частью стандартного пакета.

- Контактный план (нем. KOP, англ. LAD) – это графическое представление языка программирования STEP 7. Его синтаксис для команд похож на релейно-контактные схемы: такая схема дает возможность проследить поток энергии между шинами при его прохождении через различные контакты, составные элементы и выходные катушки.
- Список команд (нем. AWL, англ. STL) – это текстовое представление языка программирования STEP 7, подобное машинному коду. Если программа написана в виде списка команд, то отдельные команды соответствуют шагам, с помощью которых CPU исполняет программу. Для облегчения программирования список команд расширен путем включения в него некоторых конструкций языков высокого уровня (таких как доступ к структурированным данным и параметры блоков).
- Функциональный план (нем. FUP, англ. FBD) – это графическое представление языка программирования STEP 7, использующее для представления логики логические блоки подобно булевой алгебре. Сложные функции (например, математические функции) могут быть представлены непосредственно в соединении с логическими блоками.

Другие языки программирования доступны в виде дополнительных пакетов.

Конфигурирование аппаратуры

Это инструментальное средство используется для конфигурирования и назначения параметров аппаратуре, используемой в проекте автоматизации. Имеются в распоряжении следующие функции:

- Для конфигурирования программируемого контроллера вы выбираете стойки из электронного каталога и размещаете выбранные модули в необходимых слотах на этих стойках.
- Конфигурирование децентрализованной периферии идентично конфигурированию центрального устройства. Поддерживаются также входы/выходы на уровне каналов.
- В процессе назначения параметров CPU вы можете установить такие свойства как поведение при запуске и контроль времени цикла под управлением меню. Поддерживается многопроцессорный режим. Введенные данные хранятся в системных блоках данных.
- В процессе назначения параметров модулям все устанавливаемые параметры назначаются вами через диалоговые окна. Настройки, устанавливаемые с помощью двухпозиционных переключателей, отсутствуют. Присвоение параметров модулям производится автоматически при запуске CPU. Это значит, например, что модуль может быть заменен без назначения новых параметров.
- Назначение параметров функциональным модулям (FM) и коммуникационным процессорам (CP) производится с помощью инструментального средства Hardware Configuration [Конфигурирование аппаратуры] точно таким же образом, как и для других модулей. Для каждого FM и CP (включенного в сферу действия функционального пакета FM/CP) существуют специфические для модулей диалоговые окна и правила. Система препятствует неправильным вводам, предлагая только допустимые варианты в диалоговых окнах.

NetPro (конфигурирование сетей)

Использование управляемой временем циклической передачи данных NetPro через MPI возможно, когда вы:

- выбираете коммуникационные узлы
- вводите в таблице источник и приемник данных; все подлежащие загрузке блоки (SDB) генерируются автоматически и полностью автоматически загружаются во все CPU

Возможна также передача данных, управляемая событиями, когда вы:

- устанавливаете коммуникационные соединения
- выбираете коммуникационные или функциональные блоки из встроенной библиотеки блоков
- назначаете параметры выбранным коммуникационным или функциональным блокам на выбранном вами языке программирования

1.3 Что нового содержится в STEP 7 версии 5.0?

SIMATIC Manager

- Фильтрация отображаемых объектов в правой половине окна проекта.
- Параметры запуска могут быть переданы в SIMATIC Manager.
- Для определенных системных сообщений в будущем станет возможной деактивация их отображения. Используя кнопку "Activate [Активизировать]" в закладке "General [Общие свойства]" (**Options > Customize [Параметры > Настроить]**) вы можете снова активизировать любые неактивные сообщения.
- Могут создаваться и редактироваться файлы WinLC (для эмуляции программируемых контроллеров / SoftPLC).
- Команда меню для вычисления контрольной суммы содержимого платы памяти.
- Если вы выделяете блок в подробном изображении ("Details"), то размер блока отображается в строке состояния.
- Колонки для подробного отображения правого окна (окно содержимого) также могут конфигурироваться с помощью команды меню **Options > Customize [Параметры > Настроить]**.
- Блоки и системные данные могут сравниваться друг с другом в режиме offline.
- Левое окно (окно с древовидной структурой) может быть распечатано.
- Могут быть вставлены новые объекты:
- H-станции (требуется дополнительный пакет S7 H для редактирования отказоустойчивых систем)
- Станции SIMATIC PC (для проектирования соединений с PC).

LAD/STL/FBD - Programming Blocks [Программирование блоков – KOP/AWL/FUP]

- После определенных изменений интерфейса могут автоматически сравниваться любые вызовы блоков, ставшие недопустимыми:
  - **Edit > Block Call > Update Block Interface [Редактирование > Вызов блока > Обновить интерфейс блока],**
  - **Edit > Block Call > Change to Multiple Instance Call [Редактирование > Вызов блока > Перейти к мультиэкземплярному вызову],**
  - **Edit > Block Call > Change to FB/DB Call [Редактирование > Вызов блока > Перейти к вызову FB/DB]**.

  Вы должны проверить, чтобы все блоки были скомпилированы с помощью STEP 7 версии 5. Сгенерируйте исходный файл для всех блоков и скомпилируйте его или откройте блоки и сохраните их.
- У блоков, открытых online, операнды могут быть изменены из окна редактора:
  - Modify Address [Изменить операнд] (меню Debug [Отладка]),
  - Modify Address to 1 [Изменить операнд в 1] (меню Debug [Отладка]),
  - Modify Address to 0 [Изменить операнд в 0] (меню Debug [Отладка]).

- В свойствах папки блоков (offline) вы можете установить, хотите ли вы, чтобы при редактировании таблицы символов для блоков, которую вы уже сохранили, имели приоритет абсолютные адреса или символы (приоритет адреса для символа или для абсолютного значения).

  Требование состоит в том, чтобы блоки были скомпилированы с использованием STEP 7 версии 5 (как и выше, с помощью **Edit > Call > Update [Редактирование > Вызов > Обновить]**).
- Строки заголовков окон online выделены цветом, чтобы лучше отличать их от окон offline.
- Используя команду меню **View > Display > Symbol Selection [Вид > Отобразить > Выбранные символы]**, вы можете включать и выключать отображение списка текущих символов, когда вы вводите адреса в KOP и FUP. Первое символическое имя, совпадающее с вводимыми символами, выделяется в списке.
- Используя команду меню **View > Update Symbol Selection [Вид > Обновить выбранные символы]**, вы можете обновить отображенные на экране символы, чтобы учесть изменения, выполненные вами в таблице символов.
- Расширено диалоговое окно, появляющееся, когда вы выбираете команду меню **Options > Customize [Параметры > Настроить]**, путем включения в него закладки "Symbol Selection [Выбранные символы]". Здесь вы можете определить настройки для способа появления символов.
- У вас есть возможность создания шаблонов сетей, составленных из одной или нескольких сетей, и сохранения их в библиотеках. Шаблон сети может содержать символы маркеров, например, для адресов (команда меню **Edit > Create Network Template [Редактирование > Создать шаблон сети]**).
- При сохранении блока может быть автоматически сгенерирован исходный файл (команда меню **Options > Customize [Параметры > Настроить]**, закладка "Source Files [Исходные файлы]").

Monitoring and Modifying Variables [Наблюдение и управление переменными]

- При наблюдении за переменными вы можете их редактировать, например, добавлять новые переменные, модифицировать или удалять существующие переменные.
- Строки заголовков окон online выделены цветом, чтобы лучше отличать их от окон offline.
- Список выбранных символов может быть отображен также с использованием команды меню **Insert > Symbol [Вставка > Символ]**. Эта помощь для ввода символов доступна также тогда, когда отображение выбранных символов деактивировано.
- Используя команду меню **View > Display Symbol Selection [Вид > Отобразить выбранные символы]**, вы можете включать и выключать отображение списка текущих символов, когда вы вводите адреса. Первое символическое имя, совпадающее с вводимыми символами, выделяется в списке.
- Используя команду меню **View > Update Symbol Selection [Вид > Обновить выбранные символы]**, вы можете обновить отображенные на экране символы, чтобы учесть изменения, выполненные вами в таблице символов.

- Командой меню **Options > Customize [Параметры > Настроить]** отображается диалоговое окно, в котором вы можете выполнить настройку для выбора символов.
- Команда меню **Variable > Modify/Force Value Valid [Переменная > Изменить/Принудительно присвоить допустимое значение]** переименована в **Variable > Modify/Force Value as Comment [Переменная > Изменить/Принудительно присвоить значение как комментарий]**
- Команда меню **Variable > Trigger [Переменная > Запуск]** переименована в **Variable > Set Trigger [Переменная > Установить запуск]**.
- Новые или измененные кнопки на панели инструментов в командах меню:
  Variable > Monitor [Переменная > Наблюдать],
  Variable > Modify [Переменная > Изменить],
  Variable > Update Monitor Values [Переменная > Обновить наблюдаемые значения],
  Variable > Activate Modify Values [Переменная > Активизировать измененные значения],
  Variable > Modify/Force Value as Comment [Переменная > Изменить/Принудительно присвоить значение как комментарий].
- Новые или измененные комбинации клавиш:

CTRL+J	**Insert > Symbol [Вставка > Символ]**
CTRL+L	**PLC > Connect To > Configured CPU [ПЛК > Подключить к > Сконфигурированный CPU]**
CTRL+7	**View > Display > Symbol Selection [Вид > Отобразить > Выбранные символы]**
CTRL+ALT+E	**Options > Customize [Параметры > Настроить]**

- У блоков, открытых online, операнды могут быть изменены из окна "LAD/STL/FBD: Programming Blocks [KOP/AWL/FUP: Программирование блоков]" (команды меню **Debug > Modify Address [Отладка > Изменить операнд]**, **Debug > Modify Address to 0 [Отладка > Изменить операнд на 0]** и **Debug > Modify Address to 1 [Отладка > Изменить операнд на 1]**).

Конфигурирование аппаратуры

- Символы для каналов модулей ввода/вывода могут быть назначены во время конфигурирования (команда меню **Edit > Symbols [Редактирование > Символы]**).
- Аппаратные конфигурации могут быть экспортированы и импортированы (команда меню **Station > Export [Станция > Экспорт]** или **Station > Import [Станция > Импорт]**).
- Пробелы в адресной области также могут быть отображены в диалоговом окне Address Overview [Обзор адресов] (команда меню **View > Address Overview [Вид > Обзор адресов]**).
- На одну станцию может быть сконфигурировано больше master-систем DP (максимум до 10).
- Для абонентов PROFIBUS-DP могут быть спроектированы межузловые коммуникации.
- Могут быть сконфигурированы новые CPU с новыми характеристиками; например:
- CPU S7 с интерфейсом, выбираемым с помощью переключателя (MPI/PROFIBUS-DP)
- Команды управления SYNC и FREEZE для встроенного интерфейса DP
- CPU S7-400 с назначением адресов на уровне байтов

- Конфигурируемый размер памяти (для диагностического буфера, области локальных данных, коммуникационных заданий, образа процесса ...)
- Новый тип запуска "cold restart [холодны рестарт]" может быть сконфигурирован после включения питания.

Конфигурирование сетей и соединений

- В сеть может быть включен новый объект "SIMATIC PC station [Станция SIMATIC PC]"; пригодна для проектирования соединений S7 и отказоустойчивых соединений S7 с приложениями PC.
- Резервные соединения могут быть спроектированы для H-станций (требуется дополнительный пакет S7 H).
- Могут быть установлены циклы шины PROFIBUS-DP одинаковой длины (эквидистантное поведение).
- Назначенное устройство программирования/PC выделяется в отображении сети и в SIMATIC Manager отдельным символом или окраской.
- Дополнительные функции загрузки: для загрузки станций в подсети (команда меню **PLC > Download > Stations on Subnet [ПЛК > Загрузить > Станции в подсети]**) и для загрузки соединений и информации о маршрутизаторах отдельно (**PLC > Download > Connections and Routers [ПЛК > Загрузить > Соединения и маршрутизаторы]**).
- Имеется возможность считать конфигурацию сети (с соединениями) станция за станцией в проект (команда меню **PLC > Upload [ПЛК > Считать]**).
- Станции, подключенные к подсети, отличной от устройства программирования, доступны через маршрутизаторы (информация о маршрутизаторах должна быть загружена).

Редактор символов

- Стандартизовано диалоговое окно "Find and Replace [Найти и заменить]".
- Имеется возможность назначить абсолютный адрес колонке "Symbol" автоматически (команда меню **Edit > Add Default Symbolic Name [Редактировать > Добавить символическое имя по умолчанию]**).

Проектирование сообщений

- Теперь имеется диалоговое окно проектирования сообщений PCS7 для редактирования типов сообщений и сообщений, подлежащих выводу на устройствах отображения WinCC. Этот диалог упрощает процедуру конфигурирования устройств отображения и ввод атрибутов и текстов для сообщений, и помогает обеспечить непротиворечивость сообщений.
- Пользовательские тексты, созданные в STEP 7, могут редактироваться вне STEP 7 с использованием редактора ASCII или редактора таблиц, если вы выбираете команды меню **Options > Translate Texts [Параметры > Транслировать тексты]** и **Texts > Export [Тексты > Экспорт]**. Когда вы закончили редактирование пользовательских текстов, они могут быть импортированы обратно в STEP 7 с помощью команды мен **Texts > Import [Тексты > Импорт]**.

Документация для пакета программного обеспечения STEP 7

Была пересмотрена структура документации для пакета STEP 7, и вся информация была встроена в оперативную помощь в режиме online.

Это значит, что теперь у вас есть значительно более полный обзор имеющихся в наличии тем помощи и более быстрый доступ к искомой информации. Основные изменения состоят в следующем:

- Все основные знания, необходимые для работы с программным обеспечением STEP 7, включены в **центральную помощь по STEP 7**. Независимо от того, какое приложение используется в данный момент, вы можете обратиться ко всем тематическим областям от введения и программирования вплоть до конфигурирования аппаратуры и проектирования коммуникационных соединений с помощью STEP 7. Порядок тем соответствует процедуре работы со STEP 7.
- Впервые **помощь по STEP 7** предоставляется в **формате HTML**. Это значит, что предметный указатель и содержание выбранной темы помощи теперь могут быть отображены в одном окне помощи. Теперь вы можете получить быстрый обзор дополнительной информации, доступной на открытой книге, путем просмотра предметного указателя в левой секции окна помощи (подобно проводнику в Windows). Одновременно вы можете отобразить выбранную информацию в правой секции окна помощи. Если вам нужно перейти к другим темам через звенья "See also [Смотри также]", то предметный указатель изменяется автоматически.
- Вся основная информация предоставляется в виде оперативной online--помощи STEP 7 и в виде электронного руководства в формате PDF. Кроме того, вы можете получить эту информацию в бумажной форме, как обычно, от вашего местного представителя фирмы Siemens.

Дополнительная информация об использовании функций помощи может быть найдена в разделе "Notes on the Documentation [Замечания по документации]" в файле Readme.wri на вашем CD STEP 7. Этот файл содержит также подробности любых изменений в оперативной online-помощи и в электронных руководствах, которые не были известны во время издания.

1.4 Расширенное использование стандартного пакета STEP 7

1.4.1 Расширенное использование стандартного пакета STEP 7

Стандартный пакет может быть расширен с помощью дополнительных программных пакетов, которые сгруппированы в следующие три класса программного обеспечения:

- Инструментальные средства для проектирования;
  это языки программирования высокого уровня и программное обеспечение, ориентированное на технологии.
- Рабочее (Run-Time) программное обеспечение;
  оно содержит готовые к использованию рабочие программы для производственного процесса.
- Human Machine Interfaces [человеко-машинные интерфейсы] (HMI);
  это программное обеспечение специально для управления и наблюдения оператором.

Следующая таблица показывает дополнительное программное обеспечение, которое вы можете использовать в зависимости от вашей системы программного управления и стандартного пакета:

	STEP 7 - Mini		**STEP 7**		
	S7-300	**C7-620**	**S7-300 S7-400**	**M7-300 M7-400**	**C7-620**
Инструменты для проектирования					
Borland C/C++				o	
CFC			+[1)]	+	+[2)]
DOCPRO			+	+[3)]	+
HARDPRO			+		
M7 ProC/C++				o	
S7 GRAPH			+[1)]		+[2)]
S7 HiGraph			+		+
S7 PDIAG			+		
S7 PLCSIM			+		+
S7 SCL			+		+
Teleservice			+	+	+
Рабочее программное обеспечение					
Fuzzy Control [Нечеткий регулятор]			+		+
Сервер M7-DDE				+	
M7-SYS RT				o	
Modular PID Control [Модульный PID-регулятор]			+		+
Сервер PC-DDE			+		
PRODAVE MPI			+		
Standard PID Control [Стандартный PID-регулятор]			+		+
Человеко-машинный интерфейс					
ProAgent					
SIMATIC ProTool					
SIMATIC ProTool/Lite		o			o
SIMATIC WinCC					

o = обязательный
\+ = необязательный
[1)] = рекомендуется от S7-400 и выше
[2)] = не рекомендуется для C7-620
[3)] = не для программ на языке Си

1.4.2 Инструментальные средства для проектирования

Инструментальные средства для проектирования – это инструментальные средства, ориентированные на задачи, которые могут быть использованы для расширения стандартного пакета. Инструментальные средства для проектирования включают в себя:

- Языки высокого уровня для программистов
- Графические языки для технического персонала
- Дополнительное программное обеспечение для диагностики, имитации, дистанционного обслуживания, документирования установки и т. д.

Языки высокого уровня

Следующие языки доступны как дополнительные пакеты для использования при программировании программируемых логических контроллеров SIMATIC S7-300/S7-400:

- S7 GRAPH – это язык программирования, используемый для программирования последовательного управления (состоящего из шагов и переходов). В этом языке ход процесса делится на шаги. Эти шаги содержат действия по управлению выходами. Переход от одного шага к другому управляется условиями переключения.
- S7 HiGraph – это язык программирования, используемый для описания асинхронных, непоследовательных процессов в виде графов состояний. Чтобы сделать это, установка разбивается на отдельные функциональные единицы, каждая из которых может принимать различные состояния. Эти функциональные единицы могут быть синхронизированы путем обмена сообщениями между графами.
- S7 SCL – это текстовый язык высокого уровня, удовлетворяющий требованиям стандарта EN 61131-3 (IEC 1131-3). Он содержит языковые конструкции, подобные имеющимся в языках программирования Pascal и C. Поэтому S7 SCL особенно пригоден для пользователей, привыкших работать с языками высокого уровня. Язык S7 SCL может быть

использован, например, для программирования сложных и часто встречающихся функций.

Графический язык

CFC для S7 и M7 – это язык программирования для графического связывания существующих функций. Эти функции покрывают широкий диапазон от простых логических операций до сложных систем управления, работающих по замкнутому и разомкнутому циклу. Большое количество функций этого типа доступно в виде блоков в библиотеке. Вы программируете, копируя эти блоки в схему и соединяя их с помощью линий.

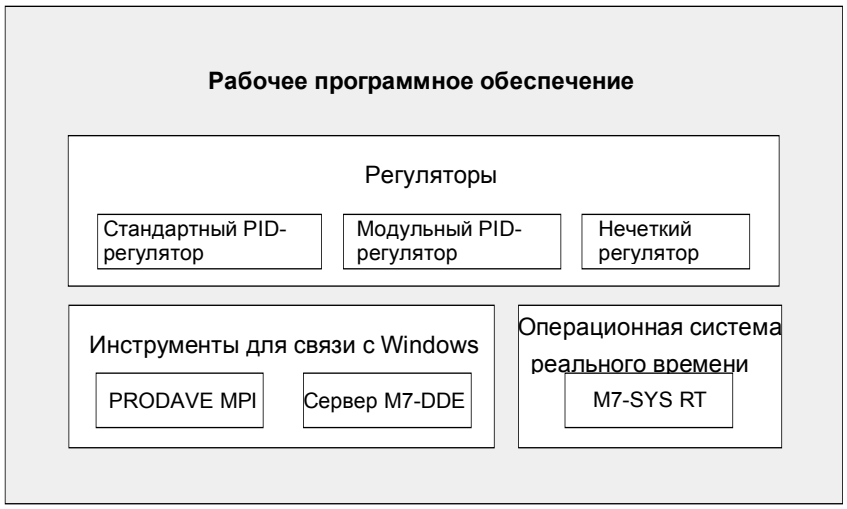
Дополнительное программное обеспечение

- Borland C++ (только для M7) содержит среду проектирования фирмы Borland.
- С помощью DOCPRO вы можете организовать все конфигурационные данные, которые вы создаете с помощью STEP 7, в руководства по монтажу. Это облегчает управление конфигурационными данными и позволяет подготовить информацию к распечатке в соответствии с указанными стандартами.
- HARDPRO – это система конфигурирования аппаратуры для S7-300, предназначенная для поддержки пользователя при крупномасштабном конфигурировании сложных задач автоматизации.
- M7-ProC/C++ (только для M7) позволяет встроить среду проектирования Borland для языков программирования C и C++ в среду проектирования STEP 7.
- Вы можете использовать S7 PLCSIM (только для S7) для имитации программируемых контроллеров S7, соединенных с устройством программирования или PC, в целях тестирования.
- S7 PDIAG (только для S7) предоставляет стандартизованную конфигурацию диагностики процесса для SIMATIC S7-300/S7-400. Используя диагностику процесса, вы можете обнаруживать дефекты и неисправные состояния вне программируемого контроллера (например, не достигнут конечный выключатель).
- TeleService позволяет вам программировать и обслуживать удаленные программируемые контроллеры S7 и M7 через телефонную сеть, используя ваше устройство программирования или PC.

1.4.3 Рабочее (Run-Time) программное обеспечение

Рабочее (Run-Time) программное обеспечение охватывает заранее запрограммированные решения, которые могут быть вызваны программой пользователя. Рабочее программное обеспечение непосредственно встроено в решение задачи автоматизации. Оно включает в себя:

- Регуляторы для SIMATIC S7, например, стандартный, модульный и нечеткий регулятор
- Инструментальные средства для связи программируемых контроллеров с приложениями Windows
- Операционную систему реального времени для SIMATIC M7



Регуляторы для SIMATIC S7

- Стандартный PID-регулятор дает возможность встраивать в программу пользователя непрерывные регуляторы, импульсные регуляторы и ступенчатые регуляторы. Инструментальное средство назначения параметров со встроенной настройкой регулятора дает возможность настраивать регулятор на оптимальный режим за очень короткое время.
- Модульный PID-регулятор вступает в действие, если простой PID-регулятор недостаточен для решения вашей задачи автоматизации. Путем включения поставляемых стандартных функциональных блоков может быть спроектирована и настроена почти любая структура регулятора.
- С помощью нечеткого регулятора (Fuzzy Control) могут создаваться системы с нечеткой логикой. Эти системы вступают в действие, когда процессы очень сложны или не могут быть описаны математически, поведение процессов непредсказуемо или появляются нелинейности, но доступно экспериментальное исследование действующего процесса.

Инструментальные средства для связи с Windows

- PRODAVE MPI – это набор инструментов для управления потоком данных между SIMATIC S7, SIMATIC M7 и SIMATIC C7. Он управляет потоком данных автономно через многоточечный интерфейс (MPI).
- С помощью сервера M7-DDE (**D**ynamic **D**ata **E**xchange – Динамический обмен данными) приложения Windows могут быть связаны с переменными процесса в SIMATIC M7 без дополнительных затрат на программирование.

Операционная система реального времени

- M7-SYS RT содержит операционную систему M7 RMOS 32 и системные программы. Это предпосылка для использования M7-ProC/C++ и CFC для пакетов SIMATIC M7.

1.4.4 Человеко-машинный интерфейс

Человеко-машинный интерфейс (Human Machine Interface, HMI) – это программное обеспечение специально для управления и наблюдения за процессом со стороны оператора в SIMATIC.

- Открытая система визуализации процесса SIMATIC WinCC – это базовая система взаимодействия с оператором со всеми важными функциями управления и наблюдения оператора, которые могут быть использованы в любой отрасли промышленности и с любой технологией.
- SIMATIC ProTool и SIMATIC ProTool/Lite – это современные инструментальные средства для конфигурирования панелей оператора SIMATIC (OP) и компактных устройств SIMATIC C7.
- ProAgent дает возможность целенаправленной и быстрой диагностики в установках и машинах путем установления информации о месте и причине неисправности.

2 Установка и авторизация

2.1 Авторизация

2.1.1 Авторизация

Для использования программного обеспечения STEP 7 за исключением STEP 7 Mini вам потребуется специфическая для продукта авторизация (права пользователя). То есть программное обеспечение защищено от копирования и может быть использовано только в том случае, если на жестком диске соответствующего устройства программирования или PC найдена соответствующая авторизация для программы или программного пакета.

Например, различные авторизации требуются для STEP 7 и для дополнительных пакетов программного обеспечения.

2.1.2 Установка и удаление авторизации

Авторизационная дискета

Предназначенная только для чтения авторизационная дискета включается в объем поставки программного обеспечения. Она содержит текущую авторизацию. Программа "AuthorsW", требуемая для отображения, установки и удаления авторизации, находится на CD-ROM, который содержит также STEP 7 V5.0.

Количество возможных авторизаций определяется счетчиком авторизаций на авторизационной дискете. Каждый раз, как вы устанавливаете авторизацию, этот счетчик уменьшается на 1. Когда значение счетчика достигнет нуля, вы более не сможете установить ни одной авторизации с помощью этой дискеты.

Замечание

Для стандартного программного пакета STEP 7 вы получили желтую авторизационную дискету с соответствующей авторизацией. Для STEP 7 Mini авторизация не нужна.

Для дополнительных программных пакетов вы получите красную авторизационную дискету с одной авторизацией для каждого.

Предостережение
Прочитайте замечания в файле README.TXT на авторизационной дискете и указания в "Guidelines for Handling Authorizations [Руководящие указания по обращению с авторизациями]". Если вы не будете придерживаться этих указаний, то авторизация может быть безвозвратно утеряна.

Вы можете использовать стандартное программное обеспечение для ознакомления с пользовательским интерфейсом без авторизации. Однако эффективная работа со STEP 7 допускается и возможна только с установленной авторизацией. Если вы не установили авторизацию, то через регулярные интервалы времени вы будете получать напоминание о необходимости сделать это.

Что делать при потере авторизации...

Авторизация может быть, например, потеряна, если возникла неисправность на жестком диске, и у вас нет возможности снять авторизацию с неисправного жесткого диска.

Если вы потеряли авторизацию, вы можете вернуться к использованию аварийной авторизации. Она тоже находится на авторизационной дискете. Эта аварийная лицензия дает вам возможность продолжить работу с программным обеспечением в течение ограниченного периода времени. В этом случае время, остающееся до истечения интервала действия авторизации, отображается на экране в начале работы. В течение этого периода вы должны получить замену потерянной авторизации. Для этой цели обратитесь к своему местному представителю фирмы Siemens.

Замечание

Время действия аварийной лицензии начинается с момента установки авторизации, даже если вы не запустили STEP 7. Даже если вы запишете авторизацию обратно на дискету, отсчет времени действия аварийной лицензии не останавливается.

Установка AuthorsW

Версия 2.0 программы "AuthorsW", требуемой для отображения, установки и удаления авторизаций, включена в CD-ROM, который также содержит STEP 7 версии 5.0. Вы устанавливаете эту программу на своем жестком диске с помощью программы установки, откуда вы можете использовать "AuthorsW" для операций по авторизации.

Замечание

По умолчанию программа AuthorsW размещается в START --> SIMATIC --> AuthorsW → AuthorsW

Установка авторизации при первой установке пакета

Вы должны установить авторизацию, когда появляется подсказка о необходимости сделать это при установке программного обеспечения STEP 7. Действуйте следующим образом:

1. При появлении подсказки вставьте авторизационную дискету в дисковод.
2. Подтвердите подсказку.
3. Авторизация переносится на физический дисковод.

Позднейшее добавление авторизации

Если вы пытаетесь запустить пакет STEP 7, авторизация для которого отсутствует, то на экране появляется сообщение об этом. Для добавления авторизации на последующем этапе действуйте следующим образом:

1. Вставьте авторизационную дискету в дисковод для гибких дисков, например, A:.
2. Запустите с жесткого диска программу "AUTHORSW.EXE".
3. Выберите дисковод A:\. Отображаются авторизационная дискета и существующие авторизации.
4. Выберите полную авторизацию STEP 7 (неограниченный срок действия).
5. При нажатой левой клавише мыши перетащите выбранную авторизацию на целевой диск. Авторизация переносится на целевой диск.

Замечание

Авторизация действует под Windows NT только в том случае, если имеется полный доступ к дисководу жесткого диска "C:" и к целевому диску.

Обновление авторизации

Для обновления авторизации используйте команду меню "Upgrade [Модернизация]". Для выполнения этой функции вам потребуются:

- авторизационная дискета для авторизации, которую вы хотите обновить,
- программа авторизации "AuthorsW, version 2.0" на жестком диске,
- новый STEP 7 Upgrade на дискете,
- старая авторизация на дискете или жестком диске.

Во время процедуры обновления старые авторизации удаляются и заменяются новыми. Поэтому в любой момент времени авторизационная дискета не должна быть защищена от записи (read-only).

1. Вставьте новую авторизационную дискету.
2. Запустите программу "AUTHORSW.EXE" с жесткого диска.
3. Выберите команду меню **Authorization > Upgrade [Авторизация > Обновление]**. Появляется диалоговое окно, где вы можете выбрать программу обновления. Затем вам будет сделано приглашение вставить авторизационную дискету со старыми авторизациями.
4. Вставьте требуемую авторизационную дискету. Затем вы получите запрос, действительно ли вы хотите выполнить обновление. Это последняя возможность отказаться от процедуры. Как только вы дали подтверждение, процедура не должна прерываться ни при каких обстоятельствах, иначе авторизация будет потеряна.
5. Сделайте подтверждение с помощью кнопки "OK". Затем вы получите приглашение вставить авторизационную дискету со старой авторизацией.

Затем проверяется выполнение всех необходимых условий. Если контроль завершен успешно, новая авторизация активизируется, завершая обновление.

Восстановление авторизации

Если ваша авторизация имеет дефект, обратитесь к линии оперативной поддержки Hotline. Тогда вы сможете восстановить авторизацию с помощью команды меню **Authorization > Restore [Авторизация > Восстановить]**.

Удаление авторизации

Если вам необходимо повторить авторизацию, например, если вы хотите переформатировать диск, на котором она размещена, вы сначала должны создать резервную копию авторизации (удалить ее). Для этого вам потребуется оригинальная желтая авторизационная дискета для стандартного пакета STEP 7. На этой дискете вы сможете затем создать резервные копии авторизаций также и для используемых дополнительных пакетов.

Для переноса авторизации обратно на авторизационную дискету действуйте следующим образом:

1. Вставьте оригинальную желтую авторизационную дискету в дисковод жесткого диска, например, A:.
2. Запустите с жесткого диска программу "AUTHORSW.EXE".
3. Выберите диск, на котором размещена авторизация. Отображаются все авторизации, имеющиеся на этом диске.
4. Выберите требуемую авторизацию.
5. При нажатой левой клавише мыши перетащите выбранную авторизацию на диск A:\. Авторизация переносится на авторизационную дискету.
6. Закройте диалоговое окно, если вы не хотите больше удалять другие авторизации. Затем вы сможете использовать эту дискету снова для установки авторизации.

2.1.3 Руководящие указания по обращению с авторизациями

Предостережение
Прочитайте указания в этой главе и в файле README.TXT на авторизационной дискете. Если вы не будете придерживаться этих указаний, авторизация может быть безвозвратно утеряна.

Когда требуется снятие авторизации?

Все существующие авторизации должны быть удалены перед тем, как вы будете форматировать, сжимать или восстанавливать свой жесткий диск или перед установкой новой операционной системы.

Резервирование

Если резервная копия вашего жесткого диска содержит копии авторизаций, имеется опасность, что эти копии могут затереть правильно установленные авторизации при восстановлении резервных данных на жестком диске, тем самым разрушив их.

Для предотвращения затирания действующих авторизаций резервной копией вы должны выполнить одно из следующих действий:

- Удалить все авторизации перед созданием резервной копии.
- Исключить авторизации из резервной копии.

Оптимизация вашего жесткого диска

Если вы используете программу оптимизации, которая предлагает перенести фиксированные блоки данных, используйте эту возможность только в том случае, если вы перенесли все авторизации с жесткого диска обратно на авторизационную дискету.

Дефектные секторы

При установке авторизации на целевом диске появляются специальные кластеры, которые иногда помечаются как "дефектные". Не пытайтесь восстановить эти кластеры. Вы можете разрушить авторизацию.

Защита от записи и защита от копирования

Авторизационная дискета не должна быть защищена от записи.

Файлы на авторизационной дискете могут быть скопированы на другой диск (например, на жесткий диск) и использованы оттуда. Однако авторизация с использованием этих скопированных файлов невозможна; для этого вам потребуется оригинальная авторизационная дискета.

Разрешенные дисководы

Авторизация может быть установлена только на жестком диске. Для сжатых дисков (например, DBLSPACE) вы можете установить авторизацию на соответствующем основном диске.

Инструментальные средства авторизации препятствуют установке авторизации на неразрешенных дисках.

Место хранения

Когда вы устанавливаете авторизацию, авторизационные файлы сохраняются в защищенном каталоге "AX NF ZZ" с атрибутами "System" и "Hidden [Скрытый]".

- Эти атрибуты не должны изменяться.
- Файлы не должны изменяться или удаляться.
- Каталог не может быть перемещен. Копии файлов из этого каталога (авторизации) распознаются как дефектные и поэтому не являются действительными авторизациями.

В противном случае авторизация будет безвозвратно потеряна.

Защищенный каталог 'AX NF ZZ' создается один на дисковод и содержит все авторизации, установленные на этом дисководе. Он создается при установке первой авторизации и удаляется при снятии последней авторизации.

Для каждой авторизации в защищенной директории создаются два файла с одинаковыми именами и разными расширениями. Этим файлам дается то же имя, что и авторизации.

Количество авторизаций

Вы можете устанавливать на диске столько авторизаций, сколько желаете, при условии, что имеется достаточное количество свободной памяти; но только по одной для каждой версии (например, только одна для STEP 7 V4.x и одна для STEP 7 V5.x). Эти авторизации не влияют друг на друга.

Дефектные авторизации

Дефектные авторизации на жестком диске не могут быть удалены программой AuthorsW. Они могут даже воспрепятствовать установке новой, действительной авторизации. В этом случае обратитесь к вашему местному представителю фирмы Siemens.

Инструментальное средство для авторизации

Использование текущей версии V2.0 инструментального средства для авторизации AuthorsW предпочтительно по отношению к любой более старой версии.

Замечание

Так как не все более старые авторизации могут быть распознаны версией V2.0, то вам в этом случае придется работать с более старой версией AUTHORS (DOS-версия) < V3.x.

2.2 Установка STEP 7

2.2.1 Установка STEP 7

STEP 7 содержит программу Setup, которая выполняет установку автоматически. Подсказки на экране ведут вас шаг за шагом через всю процедуру установки. Программа Setup вызывается с помощью стандартной процедуры инсталляции программного обеспечения Windows 95/98 или Windows NT.

Основные этапы инсталляции:

- копирование данных на ваше устройство программирования
- установка драйверов для СППЗУ и связи
- ввод идентификационного номера (ID)
- авторизация (если необходима)

Замечание

Устройства программирования фирмы Siemens (такие как PG 740) поставляются с программным обеспечением STEP 7 на жестком диске, уже готовым для инсталляции.

Требования для инсталляции

- Операционная система
  Microsoft Windows 95, Windows 98 или Windows NT.
- Основная аппаратура:
  устройство программирования или PC с:
  - процессором 80486 или выше (процессор Pentium для Windows NT) и
  - RAM: не менее 32 Мбайт, рекомендуется 64 Мбайта
  - Цветной монитор, клавиатура и мышь, поддерживаемая Microsoft Windows 95/98/NT

  Устройство программирования (PG) – это персональный компьютер в специальном компактном исполнении, пригодный для промышленного использования. Он полностью оборудован для программирования программируемых логических контроллеров SIMATIC.
- Объем памяти:
  Обратитесь к readme-файлу за сведениями о требуемом свободном пространстве на жестком диске.

- Многоточечный интерфейс (не обязателен)
  Многоточечный интерфейс (MPI) между устройством программирования или PC и программируемым логическим контроллером требуется только в том случае, если вы хотите обмениваться информацией через MPI с программируемым логическим контроллером в STEP 7.
  Поэтому вам необходим:
  - или кабель PC/MPI, подключенный к коммуникационному порту вашего устройства, или
  - плата MPI, установленная в вашем устройстве.

  Определенные устройства программирования имеют уже встроенный многоточечный интерфейс.
- Внешний программатор ППЗУ (не обязателен)
  Внешний программатор ППЗУ необходим только в том случае, если вы желаете программировать СППЗУ с помощью PC.

2.2.2 Процедура инсталляции

Подготовка к инсталляции

Перед началом инсталляции программного обеспечения должна быть запущена операционная система Windows 95/98/NT.

- Вам не нужен внешний носитель данных, если программное обеспечение STEP 7 было поставлено на жестком диске вашего устройства программирования.
- Для установки STEP 7 с дискет вставьте дискету 1 в дисковод гибких дисков вашего устройства программирования или PC.
- Для установки с CD-ROM вставьте CD-ROM в дисковод CD-ROM вашего PC.

Запуск программы инсталляции

Для инсталляции программного обеспечения действуйте следующим образом:

1. Запустите диалоговое окно для инсталляции программного обеспечения под Windows 95/98/NT, дважды щелкнув на пиктограмме ”Add/Remove Programs [Установка и удаление программ]" на "Панели управления" (”Control Panel").
2. Щелкните на ”Install [Установить]".
3. Вставьте дискету (disk 1) или CD-ROM и щелкните на ”Continue [Продолжить]". Windows 95/98/NT автоматически ищет программу установки SETUP.EXE.
4. Шаг за шагом следуйте инструкциям, выводимым на экран программой инсталляции.

Программа ведет вас шаг за шагом через весь процесс инсталляции. Вы можете перейти к следующему шагу или вернуться к предыдущему шагу из любой позиции.

Во время установки в диалоговом окне появляются вопросы, на которые вы должны ответить, и отображаются варианты для выбора. Прочтите следующие указания, чтобы вы могли легче и быстрее ответить на эти вопросы.

Если уже установлена какая-либо версия STEP 7...

Если программа установки находит на устройстве программирования другую версию STEP 7, она сообщает об этом и предлагает принять решение, как действовать дальше:

- Отменить инсталляцию, чтобы вы могли удалить старую версию STEP 7 под Windows 95/98/NT, а затем запустить инсталляцию снова, или
- Продолжить инсталляцию и переписать старую версию новой.

Ваше программное обеспечение будет лучше организовано, если вы удалите все старые версии перед установкой новой версии. Переписывание старой версии новой версией имеет тот недостаток, что если вы затем ее будете удалять, то некоторые оставшиеся компоненты старой версии удалены не будут.

Выбор возможностей при установке

Для вас открываются три возможности выбора объема инсталляции:

- Стандартная конфигурация: все языки для пользовательского интерфейса, все приложения и все примеры. Обратитесь к текущей информации о продукте для получения данных о том, какой объем памяти необходим для этой конфигурации.
- Минимальная конфигурация: только один язык, нет примеров. Обратитесь к текущей информации о продукте для получения данных о том, какой объем памяти необходим для этой конфигурации.
- Конфигурация, определяемая пользователем: вы можете определить объем инсталляции, выбирая, какие программы, базы данных, примеры и коммуникационные функции вы хотите установить.

Идентификационный номер (ID)

Во время инсталляции вам будет предложено ввести идентификационный номер (ID). Введите этот номер, который вы найдете в сертификате на программный продукт (Software Product Certificate). В противном случае ваше стандартное программное обеспечение STEP 7 будет работать только как демонстрационная версия.

Использование авторизации

Во время инсталляции программа проверяет, установлена ли авторизация на жестком диске. Если авторизация не найдена, то появляется сообщение, что программное обеспечение может быть использовано только вместе с авторизацией. По вашему желанию вы можете запустить программу авторизации немедленно или продолжить установку и выполнить авторизацию позднее. В первом случае вставьте авторизационную дискету, когда вы получите приглашение сделать это.

Настройки интерфейса PG/PC

Во время инсталляции появляется диалоговое окно, в котором вы можете назначить параметры интерфейсу устройства программирования/PC. Дополнительную информацию об этом вы найдете в разделе "Настройка интерфейса PG/PC".

Назначение параметров платам памяти

Во время инсталляции появляется диалоговое окно, в котором вы можете назначить параметры платам памяти.

- Если вы не пользуетесь платами памяти, то вам не нужен драйвер СППЗУ (EPROM). Выберите опцию ”No EPROM Driver [Нет драйвера СППЗУ]".
- В противном случае выберите пункт, который относится к вашему устройству программирования.
- Если вы используете PC, вы можете выбрать драйвер для внешнего программатора СППЗУ. Здесь вы должны указать порт, к которому этот программатор подключен (например, LPT1).

Вы можете изменить установленные параметры после инсталляции, вызвав программу ”Memory Card Parameter Assignment [Назначение параметров платам памяти]" в программной группе STEP 7.

Системы флэш-файлов

В диалоговом окне для назначения параметров платам памяти вы можете указать, должна ли быть установлена система флэш-файлов.

Система флэш-файлов требуется, например, если вы записываете отдельные файлы на плату памяти СППЗУ или удаляете отдельные файлы с этой платы памяти в SIMATIC M7 без изменения остального содержимого платы памяти.

Если вы используете подходящее устройство программирования (PG 720/PG 740/PG 760) или внешний программатор СППЗУ и хотите использовать эту функцию, выберите инсталляцию системы флэш-файлов.

Если во время инсталляции возникает ошибка

Следующие ошибки могут вызвать неудачу инсталляции:

- Если непосредственно после запуска программы Setup возникает ошибка инициализации (initialization error), то программа, то программа, возможно, не была запущена из-под Windows.
- Не хватает памяти (Not enough memory): вам нужно не менее 100 Мбайт свободного пространства на вашем жестком диске для стандартного программного обеспечения, независимо от объема вашей инсталляции.
- Дефектный диск (Bad disk): проверьте, не дефектен ли диск. Затем обратитесь к своему местному представителю фирмы Siemens.
- Ошибка оператора (Operator error): запустите инсталляцию снова и внимательно читайте инструкции.

Завершение инсталляции

Если инсталляция выполнена успешно, то на экране появляется сообщение об этом.

Если во время инсталляции были изменены DOS-файлы, то вам будет сделано напоминание о необходимости перезапустить Windows. Если вы сделали это, вы можете запустить базовое приложение STEP 7 - SIMATIC Manager.

Вы можете также выбрать запуск SIMATIC Manager'a непосредственно из завершающего диалога инсталляции.

Как только инсталляция успешно завершена, создается программная группа для STEP 7.

2.2.3 Настройка интерфейса PG/PC

С помощью выполняемых здесь настроек вы устанавливаете коммуникационную связь между устройством программирования/PC и программируемым логическим контроллером. Во время инсталляции появляется диалоговое окно, в котором вы можете назначить параметры интерфейсу устройства программирования/PC. Вы можете вывести это диалоговое окно после инсталляции, вызвав программу ”Setting PG/PC Interface [Настройка интерфейса PG/PC]" в программной группе STEP 7. Это дает вам возможность изменить параметры интерфейса независимо от инсталляции.

Основная последовательность действий

Для работы с интерфейсом вам потребуется следующее:

- настройки в операционной системе
- надлежащие параметры интерфейса

Если вы используете устройство программирования через многоточечный интерфейс (MPI), никакой дополнительной адаптации, относящейся к операционной системе, не требуется.

Если вы используете PC с платой MPI или коммуникационные процессоры (CP), вы должны проверить назначения прерываний и адресов в панели управления ("Control Panel") Windows 95/98/NT, чтобы убедиться, что здесь нет конфликта прерываний и адресные области не перекрываются.

Чтобы облегчить назначение параметров интерфейсу устройства программирования/PC, в диалоговом окне для вашего выбора выводится набор предварительно определенных базовых параметров (параметров интерфейса).

Назначение параметров интерфейсу PG/PC

Чтобы установить параметры модуля, выполните описанные ниже шаги (более подробное описание можно найти в оперативной online-помощи):

1. Дважды щелкните в "Панели управления" ("Control Panel") на пиктограмме ”Setting PG/PC Interface [Настройка интерфейса PG/PC]".
2. Установите ”Access Point of Application [Точка доступа приложения]" на ”S7ONLINE."
3. В списке "Interface parameter set used [Используемый набор параметров интерфейса]" выберите требуемые значения параметров интерфейса. Если параметры интерфейса, которые вам нужны, не отображаются, вы должны сначала установить модуль или протокол, используя кнопку ”Install [Установить]". После этого параметры интерфейса создаются автоматически.
   - Если вы выбираете интерфейс, который **автоматически распознает параметры шины** (например, MPI-ISA Card (Auto)), вы можете подключить устройство программирования к MPI или PROFIBUS без установки параметров шины. При скорости передачи, меньшей, чем 187,5 Кбит/с, возможна задержка до одной минуты, пока параметры шины будут считаны.
     Требование для автоматического распознавания: мастера, которые циклически распределяют параметры шины, должны быть подключены к шине. Это делают все новые компоненты MPI; у подсетей PROFIBUS циклическое распределение параметров шины должно быть разрешено (настройка сети PROFIBUS по умолчанию).
   - Если вы выбираете интерфейс, который **не распознает автоматически параметры шины**, вы можете отобразить эти свойства и адаптировать их к подсети.

Изменения могут потребоваться, если возникают конфликты с другими настройками (например, с назначением прерываний или адресов). В этом случае выполните надлежащие изменения с помощью распознавания аппаратуры и панели управления в Windows 95/98/NT.

Предостережение
Не удаляйте назначение параметров модуля ”TCP/IP", если оно отображается. Это может помешать правильной работе других приложений.

Проверка назначений прерываний и адресов

Если вы используете PC с платой MPI, вы всегда должны проверить, свободны ли установленные по умолчанию прерывание и адресная область, и, если необходимо, выберите свободное прерывание и/или адресную область.

Windows 95/98

Вы можете отобразить текущие назначения под Windows 95 следующим образом:

1. Откройте диалоговое окно ”System [Система]" в панели управления (”Control Panel") и выберите закладку ”Device Manager [Администратор устройств]".
2. Выберите в отображаемом списке пункт ”Computer" и щелкните на кнопке ”Properties [Свойства]".
3. В еще одном диалоговом окне вы можете отобразить список занятых прерываний (IRQ) или список занятых адресных областей (I/O), выбрав кнопку.

Windows NT

Под Windows NT вы можете:

- отобразить настройки ресурсов через **Start > Programs > Administrative Tools (Common) > Windows NT Diagnostics > Resources [Пуск > Программы > Инструменты администрирования (общие) > Диагностика Windows NT > Ресурсы]**.
- изменить ресурсы через **PG/PC Interface > Install > Resources [Интерфейс PG/PC > Установить > Ресурсы]**.

Разница между Windows 95 и Windows NT

Вы должны назначать прерывания, адресные области и другие ресурсы в Windows NT в специальном диалоговом окне (за подробным описанием обратитесь к оперативной online-помощи).

2.3 Удаление STEP 7

2.3.1 Удаление STEP 7

Для удаления STEP 7 используйте обычную процедуру Windows:

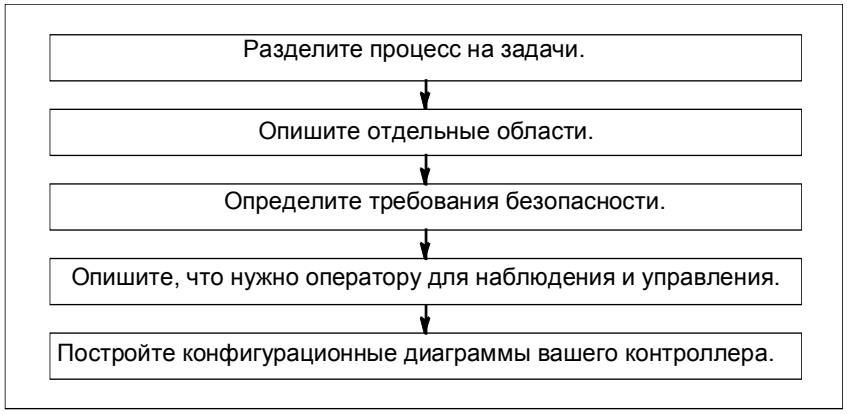
1. Запустите диалоговое окно для установки программного обеспечения под Windows двойным щелчком на пиктограмме ”Add/Remove Programs [Установка и удаление программ]" на "Панели управления" (”Control Panel").
2. Выберите пункт STEP 7 в отображенном списке установленного программного обеспечения. Щелкните на кнопке ”Add/Remove [Добавить/ Удалить]" программное обеспечение.
3. Если появляется диалоговое окно ”Remove Enabled File [Удалить разрешенный файл]", щелкните на кнопке ”No [Нет]", если сомневаетесь, как ответить.

3 Проектирование решения задачи автоматизации

3.1 Основная последовательность действий при планировании проекта автоматизации

В этой главе в общих чертах намечены основные задачи, включаемые в планирование проекта автоматизации для программируемого контроллера (ПЛК). На примере автоматизации промышленного процесса смешивания мы проведем вас шаг за шагом через всю процедуру.

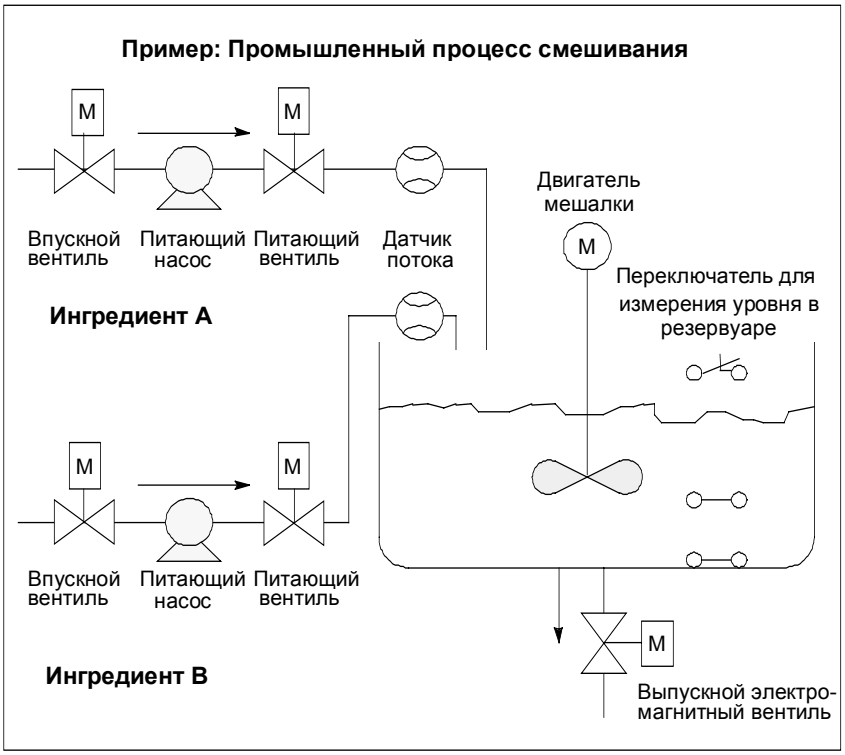
Существует много способов планирования проекта автоматизации. Основная последовательность действий, которую вы можете использовать для любого проекта, проиллюстрирована на следующем рисунке.



3.2 Деление процесса на задачи и области

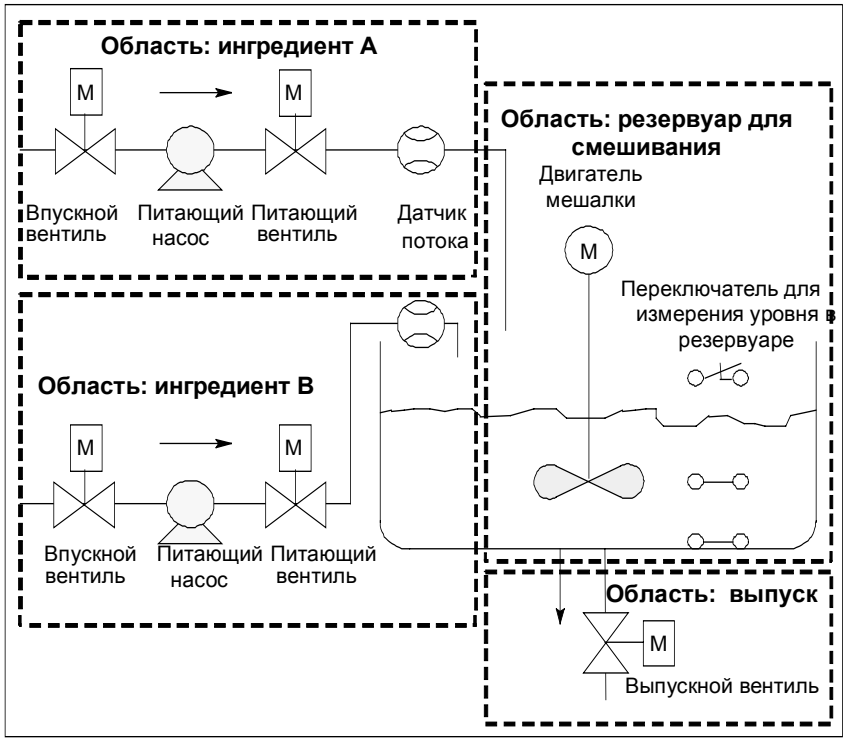
Процесс автоматизации состоит из ряда отдельных задач. Путем выделения групп связанных задач внутри процесса и последующего разбиения этих групп на более мелкие задачи может быть определен даже самый сложный процесс.

Следующий пример промышленного процесса смешивания может быть использован для иллюстрации того, как представить процесс в виде функциональных областей и отдельных задач:



Определение областей процесса

После определения процесса, подлежащего управлению, разделите процесс на связанные группы областей:



Так как каждая группа разделена на более мелкие задачи, то задачи, необходимые для управления этой частью процесса, становятся менее сложными.

В нашем примере промышленного процесса смешивания можно выделить четыре отдельные области (см. следующую таблицу). В этом примере область для ингредиента A содержит такое же оборудование, как и область для ингредиента B.

Функциональная область	Используемое оборудование
Ингредиент A	Питающий насос для ингредиента A Впускной вентиль для ингредиента A Питающий вентиль для ингредиента A Датчик потока для ингредиента A
Ингредиент B	Питающий насос для ингредиента B Впускной вентиль для ингредиента B Питающий вентиль для ингредиента B Датчик потока для ингредиента B
Резервуар для смешивания	Двигатель мешалки Переключатель для измерения уровня в резервуаре
Выпуск	Выпускной вентиль

3.3 Описание отдельных функциональных областей

Описывая каждую область и задачу внутри вашего процесса, вы не только определяете функционирование каждой области, но и различные элементы, управляющие этой областью. Они включают в себя:

- электрические, механические и логические входы и выходы для каждой задачи
- блокировки и зависимости между отдельными задачами

Промышленный процесс смешивания в нашем примере использует насосы, двигатели и вентили. Они должны быть точно описаны для определения рабочих характеристик и типа блокировок, необходимых во время работы. В следующих таблицах приведены примеры описания оборудования, используемого в промышленном процессе смешивания. Завершив описание, вы можете его также использовать для заказа необходимого оборудования.

Ингредиенты A/B: двигатели питающих насосов
1. Двигатели питающих насосов подают ингредиенты A и B в резервуар для смешивания. • Скорость потока: 400 л в минуту • Номинальная мощность: 100 кВт при 1200 об/мин.
2. Насосы управляются (пуск/остановка) со станции оператора, расположенной рядом с резервуаром для смешивания. Количество пусков подсчитывается в целях обслуживания. Как счетчики, так и индикаторы могут быть сброшены одной кнопкой.
3. Для работы насосов должны быть выполнены следующие условия: • Резервуар для смешивания не полон. • Выпускной вентиль резервуара для смешивания закрыт. • Аварийное отключение не активизировано.

Ингредиенты A/B: двигатели питающих насосов
4. Двигатели выключаются, если выполнены следующие условия: • Датчик потока извещает об отсутствии потока через 7 секунд после запуска двигателя насоса. • Датчик потока извещает, что поток прекратился.

Ингредиенты A/B: впускной и питающий вентили
1. Впускной и питающий вентили для ингредиентов A и B разрешают или запрещают подачу ингредиентов в резервуар для смешивания. Вентили имеют электромагнит с пружинным возвратом. • Когда электромагнит активизирован, вентиль открыт. • Когда электромагнит не активизирован, вентиль закрыт.
2. Впускной и питающий вентили управляются программой пользователя.
3. Для активизации вентилей должны быть соблюдены следующие условия: • Двигатель насоса должен уже работать в течение не менее 1 секунды.
4. Насосы выключаются, если выполнены следующие условия: • Датчик потока извещает об отсутствии потока.

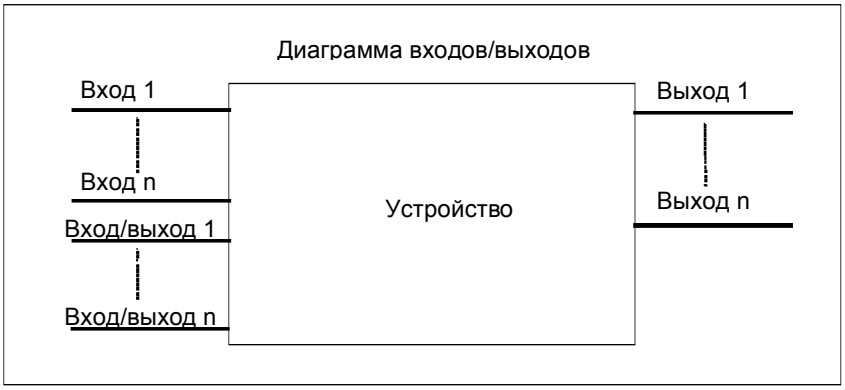
Двигатель мешалки
1. Двигатель мешалки смешивает ингредиент A с ингредиентом B в резервуаре для смешивания. • Номинальная мощность: 100 кВт при 1200 об/мин.
2. Двигатель мешалки управляется (пуск/останов) со станции оператора, расположенной рядом с резервуаром для смешивания. Количество пусков подсчитывается в целях обслуживания. Как счетчики, так и индикаторы могут быть сброшены одной кнопкой.
3. Для работы двигателя мешалки должны быть выполнены следующие условия: • Датчик уровня жидкости в резервуаре не выдает сигнала "Уровень в резервуаре ниже минимального" • Выпускной вентиль резервуара для смешивания закрыт. • Аварийный выключатель не активизирован.
4. Двигатель мешалки выключается, если выполнено следующее условие: • Тахометр не показывает достижения номинальной скорости в течение 10 секунд после запуска двигателя.

Выпускной вентиль
1. Выпускной вентиль разрешает выпуск смеси (самотеком) на следующей стадии процесса. Вентиль имеет электромагнит с возвратной пружиной. • Если электромагнит активизирован, выпускной вентиль открыт. • Если электромагнит не активизирован, выпускной вентиль закрыт.
2. Выпускной вентиль управляется (открытие/закрытие) со станции оператора.
3. Выпускной вентиль может быть открыт при следующих условиях: • Двигатель мешалки выключен. • Датчик уровня жидкости в резервуаре не выдает сигнала "Резервуар пуст". • Аварийный выключатель не активизирован.
4. Выпускной вентиль закрывается, если выполнено следующее условие: • Датчик уровня жидкости в резервуаре выдает сигнал " Резервуар пуст".

Переключатели для измерения уровня в резервуаре
1. Переключатели в резервуаре для смешивания измеряют уровень в резервуаре и используются для блокировки питающих насосов и двигателя мешалки.

3.4 Список входов, выходов и входов/выходов

Сделав физическое описание каждого устройства, подлежащего управлению, нарисуйте диаграммы входов и выходов для каждого устройства или группы задач.



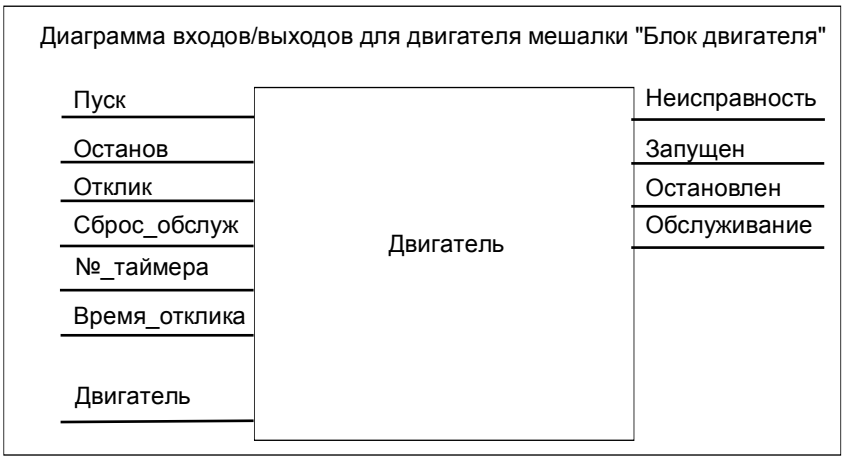
Эти диаграммы соответствуют логическим блокам, подлежащим программированию.

3.5 Создание диаграмм входов/выходов для двигателей

В нашем примере промышленного процесса смешивания используются два питающих насоса и одна мешалка. Каждый двигатель управляется своим собственным "блоком двигателя", одинаковым для всех трех устройств. Этот блок требует шести входов: два для запуска и остановки двигателя, один для сброса обслуживающего дисплея, один для ответного сигнала о работе двигателя (двигатель работает/не работает), один для времени, в течение которого должен быть получен ответный сигнал, и один для номера таймера, используемого для измерения времени.

Логический блок требует также четырех выходов: два для индикации рабочего состояния двигателя, один для индикации неисправностей и один для индикации того, что двигатель подлежит обслуживанию.

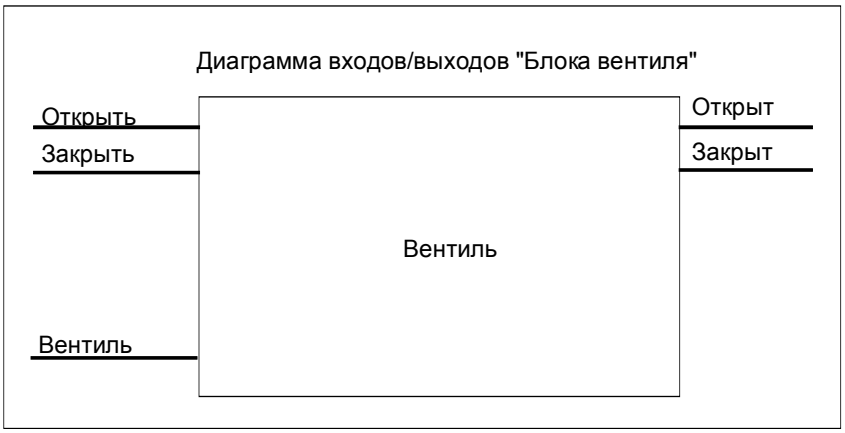
Для активизации двигателя необходим также вход/выход. Он используется для управления двигателем, но в то же время редактируется и изменяется в программе для "блока двигателя".



3.6 Создание диаграммы входов/выходов для вентилей

Каждый вентиль управляется собственным "блоком вентиля", одинаковым для всех используемых вентилей. Логический блок имеет два входа: один для открытия вентиля и один для его закрытия. У него имеется также два выхода: один для индикации того, что вентиль открыт, а другой для индикации того, что он закрыт.

Блок имеет вход/выход для активизации вентиля. Он используется для управления вентилем, но в то же самое время редактируется и изменяется в программе для "блока вентиля".



3.7 Определение требований безопасности

Определите, какие дополнительные элементы необходимы для обеспечения безопасности процесса – на основе юридических требований и корпоративной политики в области охраны здоровья и безопасности. В свое описание вам следует также включить все воздействия, которые элементы безопасности оказывают на области вашего процесса.

Определение требований безопасности

Выясните, какие устройства требуют аппаратно реализованных схем для удовлетворения требований безопасности. По определению, эти защитные схемы функционируют независимо от программируемого контроллера (хотя защитная схема в общем случае предоставляет интерфейс ввода/вывода для обеспечения координации с программой пользователя). Обычно проектируется матрица подключения каждого исполнительного устройства со своей собственной областью аварийного отключения. Эта матрица является основой для построения принципиальной схемы защитного устройства.

Для проектирования механизмов защиты действуйте следующим образом:

- Определите логические и механические/электрические блокировки между отдельными задачами автоматизации.
- Спроектируйте схемы, разрешающие ручное управление устройствами, относящимися к процессу, в случае аварии.
- Определите все остальные требования к защите для безопасного функционирования процесса.

Создание схемы защиты

Промышленный процесс смешивания в нашем примере использует следующую логику для своей схемы защиты:

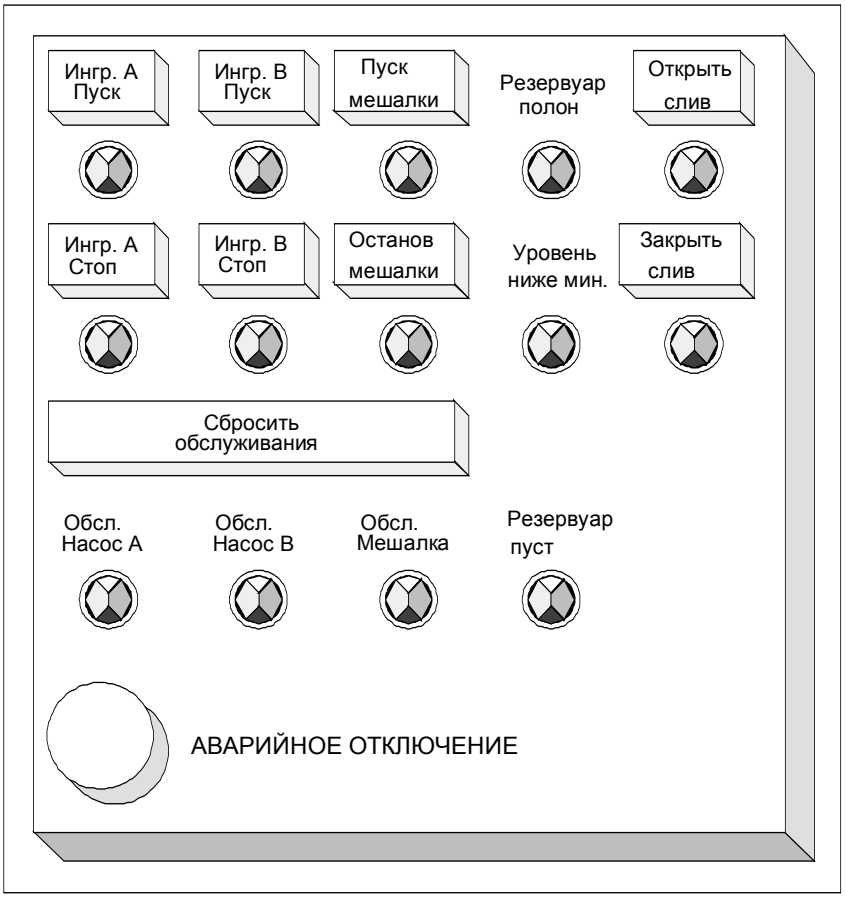
- Один аварийный выключатель отключает следующие устройства независимо от программируемого контроллера (ПЛК):
- питающий насос для ингредиента A
- питающий насос для ингредиента B
- двигатель мешалки
- вентили
- Аварийный выключатель находится на станции оператора.
- Один вход в контроллер индицирует состояние аварийного выключателя.

3.8 Описание требуемых для оператора устройств отображения и управления

Каждый процесс требует интерфейса с оператором, который обеспечивает вмешательство человека в процесс. Часть спецификации проекта включает в себя проект пульта оператора.

Описание пульта оператора

В промышленном процессе смешивания, описанном в нашем примере, каждое устройство может быть запущено или остановлено нажатием кнопки, расположенной на пульте оператора. Этот пульт оператора содержит индикаторы для отображения состояния функционирования (см. следующий рисунок).



Пульт содержит также индикаторные лампы для устройств, требующих обслуживания после определенного числа пусков, аварийный выключатель, с помощью которого процесс может быть остановлен немедленно. На пульте имеется также кнопка сброса для индикаторов обслуживания трех двигателей. С ее помощью вы можете отключить индикаторные лампы для двигателей, подлежащих обслуживанию, и сбросить соответствующие счетчики на 0.

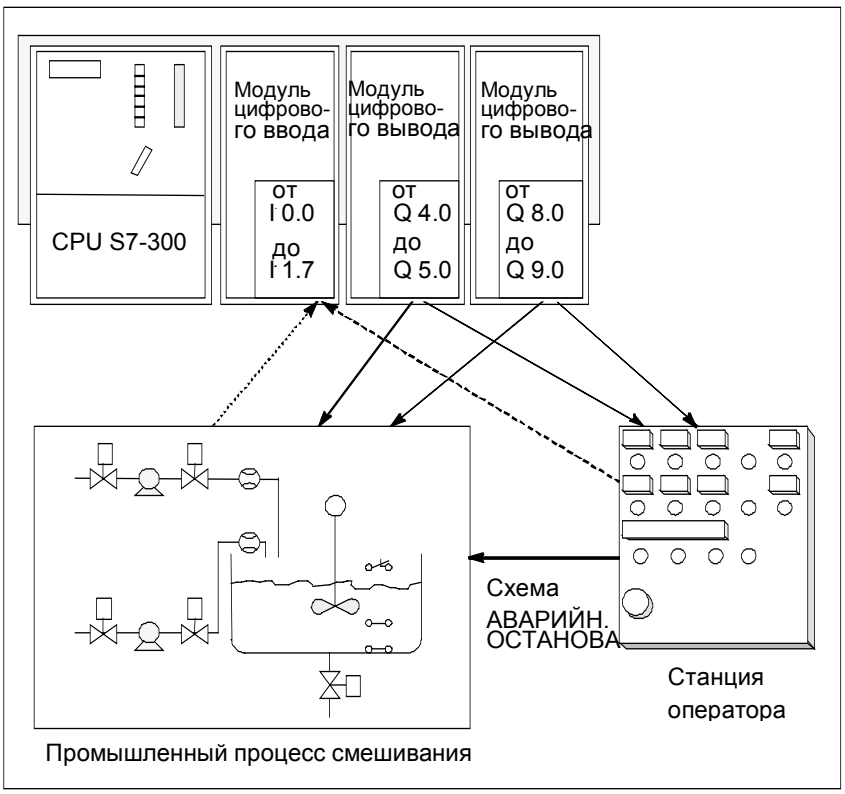
3.9 Составление конфигурационной диаграммы

После того как вы задокументировали требования к проекту, вы должны принять решение относительно типа управляющего оборудования, требующегося для проекта.

Принимая решение о том, какие модули вы хотите использовать, вы также определяете структуру программируемого контроллера. Составьте конфигурационную диаграмму, определяющую следующие аспекты:

- тип CPU
- количество и тип модулей ввода/вывода
- конфигурация физических входов и выходов

Следующий рисунок иллюстрирует пример конфигурации S7 для промышленного процесса смешивания.



4 Основы проектирования структуры программы

4.1 Программы в CPU

В CPU всегда исполняются две программы:

- операционная система
- программа пользователя

Операционная система

Каждый CPU содержит операционную систему, которая организует все функции и последовательности в CPU, не связанные с конкретной задачей управления. Задачи операционной системы состоят в следующем:

- обработка "теплого" и "горячего" перезапуска
- обновление таблицы образа процесса для входов и вывод таблицы образа процесса для выходов
- вызов программы пользователя
- обнаружение прерываний и вызов OB прерываний
- обнаружение и обработка ошибок
- управление областями памяти
- обмен информацией с устройствами программирования и другими коммуникационными партнерами

Если вы измените параметры операционной системы (настройку операционной системы по умолчанию), вы можете повлиять на работу CPU в определенных областях.

Программа пользователя

Вы должны сами создать программу пользователя и загрузить ее в CPU. Она содержит все функции, необходимые для обработки вашей конкретной задачи автоматизации. Задачи программы пользователя состоят в следующем:

- определение условий для "теплого" и "горячего" перезапуска в CPU (например, инициализация сигналов с определенным значением)
- обработка данных процесса (например, логическая комбинация двоичных сигналов, считывание и анализ аналоговых сигналов, задание двоичных сигналов для вывода, вывод аналоговых значений)
- определение реакции на прерывания
- обработка нарушений в нормальном исполнении программы

4.2 Блоки в программе пользователя

4.2.1 Блоки в программе пользователя

Программное обеспечение STEP 7 дает вам возможность структурировать свою пользовательскую программу, иными словами, разбивать программу на отдельные автономные программные секции. Это дает следующие преимущества:

- Такие программы проще для понимания.
- Отдельные программные секции могут быть стандартизованы.
- Упрощается организация программы.
- Легче производить модификацию программы.
- Отладка упрощается, так как можно тестировать отдельные секции.
- Значительно упрощается прием системы в эксплуатацию.

Пример промышленного процесса смешивания иллюстрировал преимущества разбиения процесса автоматизации на отдельные задачи. Программные секции структурированной программы пользователя соответствуют этим отдельным задачам и известны как блоки программы.

Типы блоков

Имеется несколько различных типов боков, которые вы можете использовать внутри пользовательской программы S7:

Блок	Краткое описание функции	См. также
Организационные блоки (OB)	OB определяют структуру программы пользователя.	Организационные блоки и структура программы
Системные функциональные блоки (SFB) и системные функции (SFC)	SFB и SFC встроены в CPU S7 и обеспечивают вам доступ ко всем важным системным функциям.	Системные функциональные блоки (SFB) и системные функции (SFC)
Функциональные блоки (FB)	FB – это блоки с "памятью", которые вы можете программировать сами.	Функциональные блоки (FB)
Функции (FC)	FC содержат программы для часто встречающихся функций.	Функции (FC)
Экземплярные блоки данных (экземплярные DB)	Экземплярные DB связываются с блоком, когда вызывается FB/SFB. Они создаются автоматически при компиляции.	Экземплярные блоки данных
Блоки данных (DB)	DB – это области данных для хранения данных пользователя. Кроме данных, соответствующих функциональному блоку, могут быть определены также данные, совместно используемые любыми блоками.	Совместно используемые блоки данных (DB)

OB, FB, SFB, FC и SFC содержат секции программы и поэтому известны также как логические блоки. Допустимое количество блоков каждого типа и допустимая длина блоков зависят от CPU.

4.2.2 Организационные блоки и структура программы

Организационные блоки (OB) образуют интерфейс между операционной системой и программой пользователя. Они вызываются операционной системой и управляют циклическим и по прерываниям исполнением программы, а также запуском программируемого логического контроллера. Они также обрабатывают реакцию на ошибки. Программируя организационные блоки, вы определяете реакцию CPU.

Приоритет организационного блока

Организационные блоки определяют порядок, в котором исполняются отдельные программные секции. Исполнение OB может быть прервано вызовом другого OB. Какому OB разрешается прервать другой OB, зависит от его приоритета. OB с более высоким приоритетом могут прерывать OB с более низким приоритетом. Фоновый OB имеет самый низкий приоритет.

Типы прерываний и классы приоритета

События, которые приводят к вызову OB, известны как прерывания. Следующая таблица показывает типы прерываний в STEP 7 и приоритеты соответствующих им организационных блоков. Не все перечисленные организационные блоки и их классы приоритета доступны во всех CPU S7 (см. " S7-300 Programmable Controller, Hardware and Installation Manual [Программируемый контроллер S7-300. Руководство по аппаратному обеспечению и монтажу]" и "S7-400, M7-400 Programmable Controllers Module

Specifications Reference Manual [Программируемые контроллеры S7-400, M7-400. Спецификации модулей. Справочное руководство]").

Тип прерывания	Организационный блок	Класс приоритета (по умолчанию)	См. также
Главный программный цикл	OB1	1	Организационный блок для циклической обработки программы (OB1)
Прерывания по времени суток	OB10 – OB17	2	Организационные блоки прерываний по времени (OB10 – OB17)
Прерывания с задержкой	OB20 OB21 OB22 OB23	3 4 5 6	Организационные блоки прерываний с задержкой (OB20 – OB23)
Циклические прерывания	OB30 OB31 OB32 OB33 OB34 OB35 OB36 OB37 OB38	7 8 9 10 11 12 13 14 15	Организационные блоки циклических прерываний (OB30 – OB38)
Аппаратные прерывания	OB40 OB41 OB42 OB43 OB44 OB45 OB46 OB47	16 17 18 19 20 21 22 23	Организационные блоки аппаратных прерываний (OB40 – OB47)
Мультипроцессорное прерывание	OB60 Многопроцессорный режим	25	Многопроцессорный режим – Синхронная работа нескольких CPU
Ошибки резервирования	OB70 Ошибка резервирования ввода/вывода (только в H-системах) OB72 Ошибка резервирования CPU (только в H-системах)	25 28	Организационные блоки обработки ошибок (OB70 – OB87 / OB121 – OB122)

Тип прерывания	Организационный блок	Класс приоритета (по умолчанию)	См. также
Асинхронные ошибки	OB80 Временная ошибка OB81 Ошибка по питанию OB82 Диагностическое прерывание OB83 Ошибка установки/удаления модуля OB84 Аппаратная неисправность CPU OB85 Ошибка класса приоритета OB86 Неисправность стойки OB87 Ошибка связи	26 (или 28, если OB асинхронных ошибок существует в программе запуска)	Организационные блоки обработки ошибок (OB70 – OB87 / OB121 – OB122)
Фоновый цикл	OB90	29 [1)]	Фоновый организационный блок (OB90)
Запуск	OB100 Теплый рестарт OB101 Горячий рестарт OB102 Холодный рестарт	27 27 27	Организационные блоки запуска (OB100/OB101/OB102)
Синхронные ошибки	OB121 Ошибка программирования OB122 Ошибка доступа	Приоритет OB, вызвавшего ошибку	Организационные блоки обработки ошибок (OB70 – OB87 / OB121 – OB122)

1) Класс приоритета 29 соответствует приоритету 0,29. Фоновый цикл имеет более низкий приоритет, чем свободный цикл.

Изменение приоритета

Прерываниям с помощью STEP 7 могут быть назначены параметры. Назначая параметры, вы можете, например, отменить выбор OB прерываний или классы приоритета в блоках параметров: прерывания по времени суток, прерывания с задержкой, циклические прерывания и аппаратные прерывания.

Приоритет организационных блоков в CPU S7-300 фиксирован.

У CPU S7-400 (и CPU 318) вы можете изменять приоритет следующих организационных блоков с помощью STEP 7:

- OB10–OB47
- OB70 – OB72 (только H-CPU) и OB81 – OB87 в режиме RUN.

Разрешены следующие классы приоритета:

- классы приоритета от 2 до 23 для OB10 – OB47
- классы приоритета от 2 до 28 для OB70 – OB72
- классы приоритета от 24 до 26 для OB81 – OB87

Вы можете назначить нескольким OB одинаковые приоритеты. OB с одинаковым приоритетом обрабатываются в порядке наступления событий, вызвавших их запуск.

OB ошибок, запущенные синхронными ошибками, исполняются в том же классе приоритета, что и блок, исполняющийся в тот момент, когда ошибка произошла.

Локальные данные

При создании логических блоков (OB, FC, FB) вы можете описать временные локальные данные. Область локальных данных в CPU делится между классами приоритета.

В S7-400 вы можете изменить количество локальных данных на класс приоритета в блоке параметров ”priority classes [классы приоритета]" с помощью STEP 7.

Стартовая информация OB

Каждый организационный блок имеет стартовую информацию в 20 байтов локальных данных, предоставляемых операционной системой при запуске OB. Стартовая информация указывает на событие, вызвавшее запуск OB, дату и время запуска OB, возникшие ошибки и диагностические события.

Например, OB40, OB аппаратных прерываний, содержит в своей стартовой информации, адрес модуля, который сгенерировал прерывание.

Отмененные OB прерываний

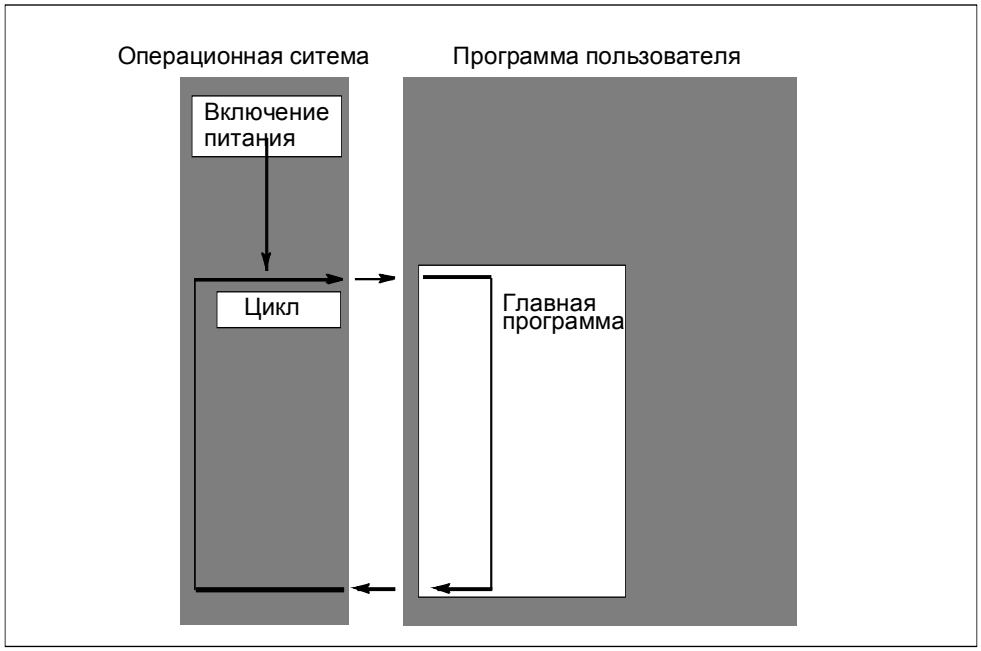
Если вы назначаете класс приоритета 0 или назначаете менее 20 байтов локальных данных классу приоритета, то выбор соответствующего OB прерываний отменяется. Обработка отмененных OB прерываний ограничена следующим образом:

- В режиме RUN они не могут быть скопированы в вашу пользовательскую программу или связаны с ней.
- В режиме STOP они могут быть скопированы в вашу пользовательскую программу или связаны с ней, но когда CPU проходит через "теплый" рестарт, они останавливают запуск, и делается запись в диагностический буфер.

Отменяя выбор OB прерываний, которые вам не нужны, вы увеличиваете размер имеющейся в распоряжении области локальных данных, а это может быть использовано для сохранения временных данных в других классах приоритета.

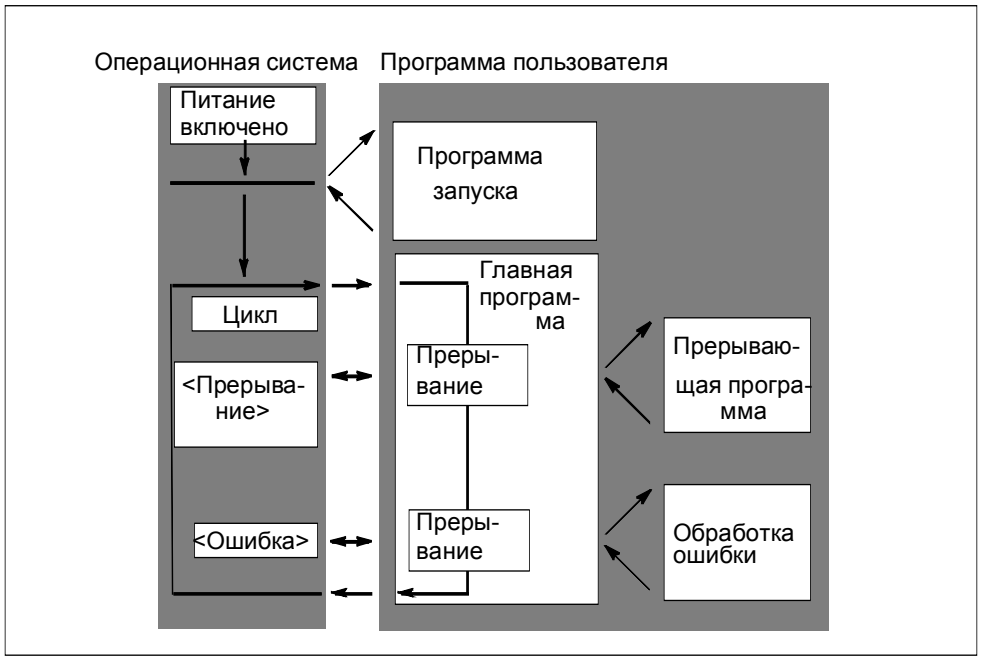
Циклическая обработка программы

Циклическая обработка программы – это "стандартный" способ исполнения программы в программируемых логических контроллерах, означающий, что операционная система работает в программном цикле и вызывает организационный блок OB1 один раз в каждом цикле главной программы. Поэтому программа пользователя в OB1 исполняется циклически.



Обработка программы, управляемая событиями

Циклическая обработка программы может быть прервана определенными событиями (прерываниями). Если такое событие происходит, то блок, обрабатываемый в это момент времени, прерывается на границе команды, и вызывается другой организационный блок, назначенный соответствующему событию. Как только этот организационный блок завершает свою работу, циклическая программа возобновляется с точки, на которой она была прервана.

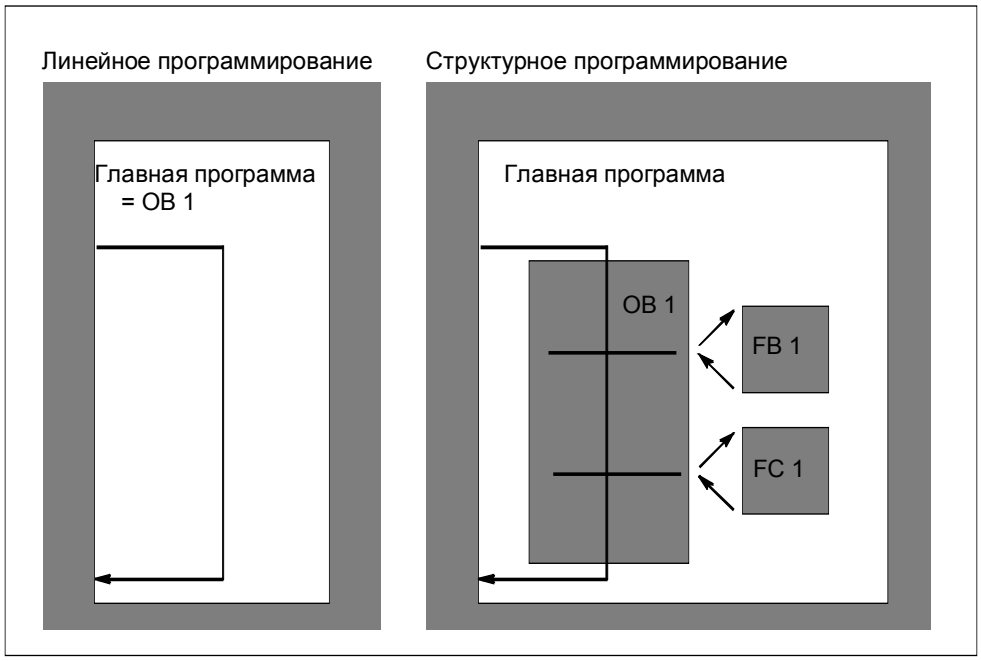


Это значит, что имеется возможность обрабатывать части программы пользователя, которые не должны обрабатываться циклически, а только тогда, когда они необходимы. Программа пользователя может быть разделена на "подпрограммы" и распределена между различными организационными блоками. Если программа пользователя должна реагировать на важный сигнал, который появляется относительно редко (например, датчик граничного значения для измерения уровня в резервуаре сообщает, что достигнут максимальный уровень), то подпрограмма, которая должна обрабатываться, когда выдается этот сигнал, может быть помещена в ОВ, обработка которого управляется событиями.

Линейное и структурное программирование

Вы можете записать всю свою программу в OB1 (линейное программирование). Это целесообразно только в случае простых программ, написанных для CPU S7-300 и требующих мало памяти.

Сложными задачами автоматизации проще управлять, если они разделены на более мелкие задачи, которые отражают технологические функции процесса и могут быть использованы неоднократно. Эти задачи представляются соответствующими программными секциями, известными как блоки (структурное программирование).



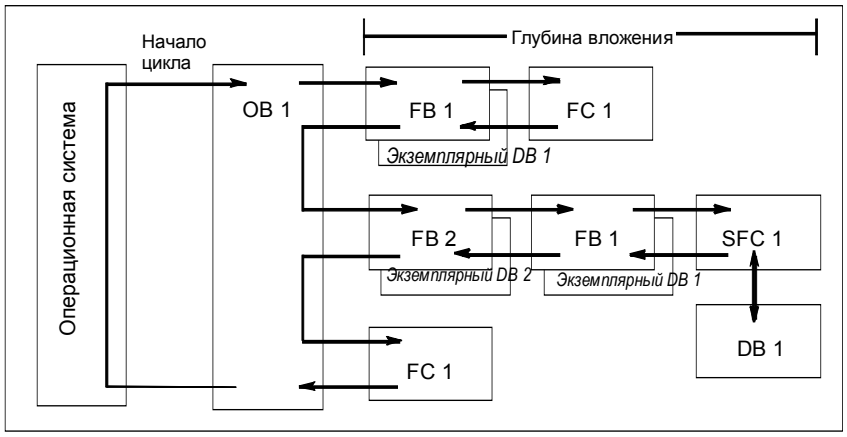
Иерархия вызовов в программе пользователя

Для функционирования программы пользователя составляющие ее блоки должны вызываться. Это делается с помощью специальных команд STEP 7, вызовов блоков, которые могут быть запрограммированы и запущены только в логических блоках.

Порядок и глубина вложения

Порядок и вложение вызовов блоков называется иерархией вызовов. Количество блоков, которые могут быть вложены друг в друга (глубина вложения), зависит от конкретного CPU.

Следующий рисунок иллюстрирует порядок и глубину вложения вызовов блоков внутри цикла обработки программы.



Существует установленный порядок создания блоков:

- Блоки создаются сверху вниз, то есть вы начинаете с верхнего ряда блоков.
- Каждый вызываемый блок уже должен существовать, т. е. внутри ряда блока они должны создаваться справа налево.
- Последним создается блок OB1.

Применение этих правил на практике для приведенного на рисунке примера дает следующую последовательность создания блоков:

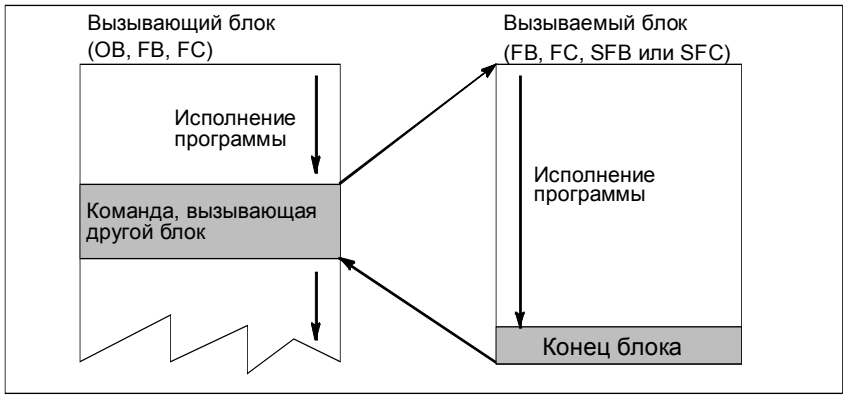
FC1 > FB1 + экземплярный DB1 > DB1 > SFC1 > FB2 + экземплярный DB2 > OB1

Замечание

Если глубина вложения слишком велика (слишком много уровней), то стек локальных данных может переполниться (см. также Стек локальных данных).

Вызовы блоков

На следующем рисунке показана последовательность вызова блоков внутри программы пользователя. Программа вызывает второй блок, команды которого затем полностью выполняются. Как только второй, или вызываемый, блок выполнен, исполнение прерванного блока, который произвел вызов, возобновляется с команды, следующей за вызовом блока.



Перед программированием блока вы должны указать, какие данные будут использоваться вашей программой, иными словами, вы должны описать переменные блока.

Замечание

Выходные параметры должны быть описаны для каждого вызова блока.

Замечание

Операционная система сбрасывает экземпляры SFB3 "TP", когда выполнен холодный рестарт. Если вы хотите инициализировать экземпляры этого SFB после холодного рестарта, вы должны вызвать соответствующие экземпляры этого SFB с PT=0 мс через OB100. Вы можете это сделать, например, выполнив программу инициализации в блоках, содержащих экземпляры этого SFB.

4.2.4 Типы блоков и циклическая обработка программы

4.2.4.1 Организационный блок для циклической обработки программы (OB1)

Циклическая обработка программы – это "стандартный" тип исполнения программы в программируемых логических контроллерах. Операционная система вызывает OB1 циклически, и этим вызовом она начинает циклическое исполнение программы пользователя.

Последовательность циклической обработки программы

В следующей таблице показаны фазы циклической обработки программы:

Шаг	Последовательность в существующих CPU	Последовательность в новых CPU (с 10/98)
1.	Операционная система запускает время контроля цикла.	Операционная система запускает время контроля цикла.
2.	CPU считывает состояния входов модулей ввода и обновляет таблицу образа процесса на входах.	CPU записывает значения из таблицы образа процесса на выходах в модули вывода.
3.	CPU обрабатывает программу пользователя и исполняет содержащиеся в ней команды.	CPU считывает состояния входов модулей ввода и обновляет таблицу образа процесса на входах.
4.	CPU записывает значения из таблицы образа процесса на выходах в модули вывода.	CPU обрабатывает программу пользователя и исполняет содержащиеся в ней команды.
5.	В конце цикла операционная системы выполняет все ждущие своей очереди задачи, например, загрузка и удаление блоков, прием и передача глобальных данных.	В конце цикла операционная системы выполняет все ждущие своей очереди задачи, например, загрузка и удаление блоков, прием и передача глобальных данных.
6.	Наконец, CPU возвращается к началу цикла и перезапускает время контроля цикла.	Наконец, CPU возвращается к началу цикла и перезапускает время контроля цикла.

Образ процесса

Чтобы в CPU во время циклической обработки программы находился непротиворечивый образ сигналов процесса, CPU обращается не непосредственно к адресным областям входов (I) и выходов (Q) на модулях ввода/вывода, а к области внутренней памяти CPU, содержащей образ входов и выходов.

Программирование циклической обработки программы

Циклическая обработка программы программируется записью программы пользователя в OB1 и в блоки, вызываемые внутри OB1 с помощью STEP 7.

Циклическая обработка программы начинается, как только завершается без ошибок программа запуска.

Прерывания

Циклическая обработка программы может быть прервана в результате:

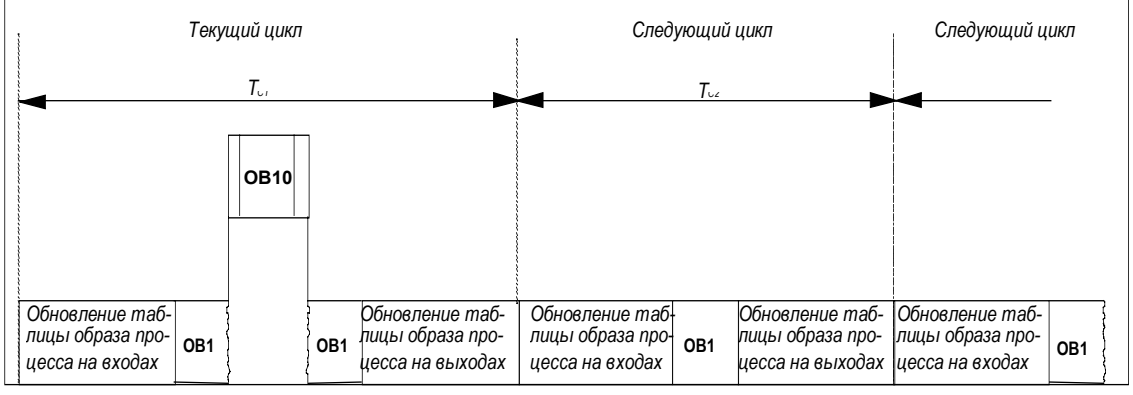
- прерывания
- команды STOP (переключатель выбора режимов, опция меню на устройстве программирования, SFC46 STP, SFB20 STOP)
- выхода из строя питания
- появления неисправности или ошибки в программе

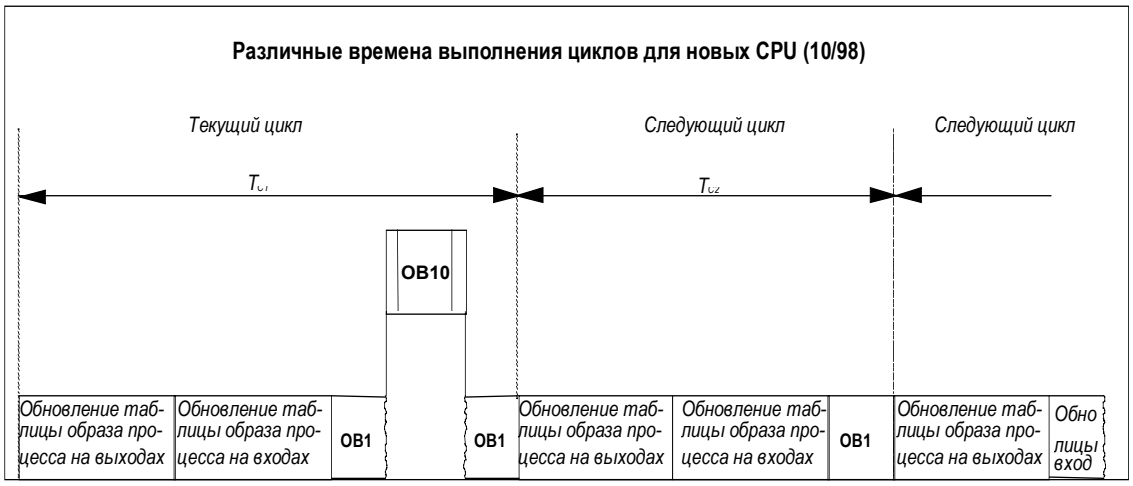
Время выполнения цикла

Время выполнения цикла – это время, необходимое операционной системе для выполнения циклической программы и всех программных секций, прерывающих цикл (например, выполнение других организационных блоков), и системных операций (например, обновления образа процесса). Это время контролируется.

Время выполнения цикла (TC) не одинаково в каждом цикле. На следующих рисунках показаны различные времена выполнения циклов (TC1 ≠ TC2) для существующих и новых CPU:

Различные времена выполнения цикла для существующих CPU





В текущем цикле выполняется прерывание OB1 по времени суток.

Максимальное время цикла

С помощью STEP 7 вы можете изменить максимальное время цикла, установленное по умолчанию. Когда это время истекает, или CPU переходит в режим STOP, или вызывается OB80, в котором вы можете определить, как CPU должен реагировать на эту ошибку.

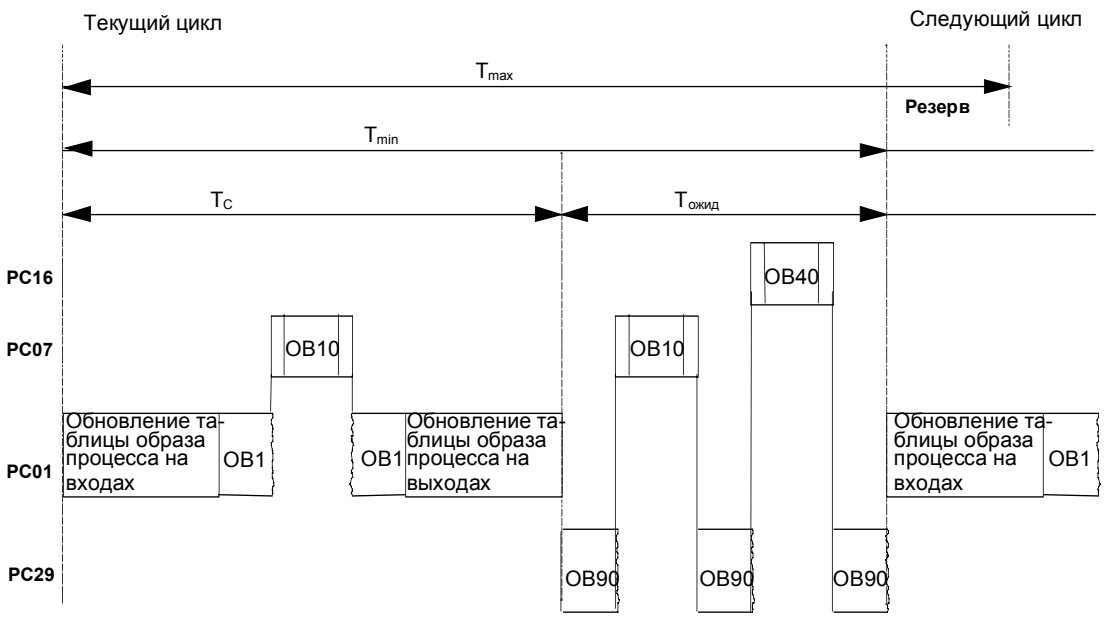
Минимальное время цикла

С помощью STEP 7 вы можете установить минимальное время цикла для S7-400 и CPU 318. Это полезно в следующих ситуациях:

- если интервал, с которым запускается исполнение программы в OB1 (проход главной программы), всегда должен быть одним и тем же или
- если таблицу образа процесса не нужно обновлять излишне часто без необходимости, когда время цикла слишком коротко

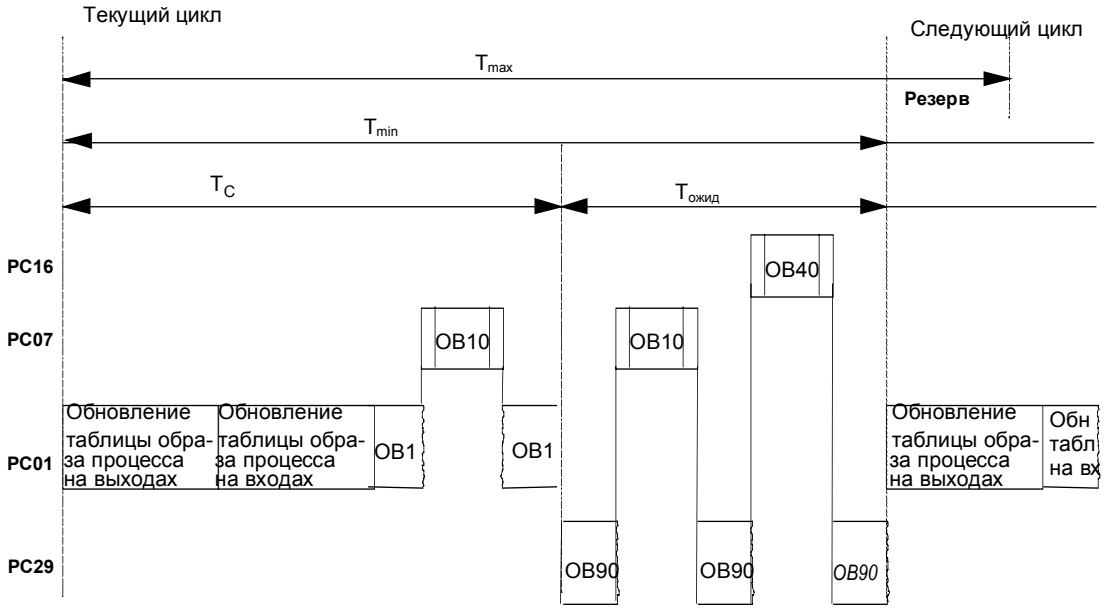
На следующих рисунках показано функционирование времени контроля цикла обработки программы в существующих и новых CPU.

Время контроля цикла для существующих CPU



Tmax = максимальное время цикла, которое может быть установлено
Tmin = минимальное время цикла, которое может быть установлено
Tc = время исполнения текущего цикла
Тожид = разность между T_{min} и временем исполнения текущего цикла. В это время могут обрабатываться произошедшие прерывания и фоновый OB
PC = Класс приоритета

Время контроля цикла для новых CPU (10/98)



Tmax = максимальное время цикла, которое может быть установлено
Tmin = минимальное время цикла, которое может быть установлено
Tc = время выполнения текущего цикла
Тожид = разность между T_{min} и временем выполнения текущего цикла. В это время могут обрабатываться произошедшие прерывания и фоновый OB
PC = класс приоритета

Обновление образа процесса

Когда CPU обрабатывает циклическую программу, образ процесса обновляется автоматически. В случае CPU S7-400 и CPU 318 вы можете отменить обновление образа процесса, если вы хотите:

- вместо этого непосредственно обращаться к входам/выходам или
- обновлять одну или более секций входов или выходов образа процесса в другой точке программы, используя системные функции SFC26 UPDAT_PI и SFC27 UPDAT_PO.

Коммуникационная нагрузка

Чтобы коммуникационные функции не слишком увеличивали время, необходимое для выполнения программы, вы можете указать максимальную величину, на которую цикл может быть расширен за счет коммуникаций.

Когда вы принимаете решение о нагрузке, добавляемой к циклу коммуникациями, помните, что время работы операционной системы дополнительно увеличивает время исполнения. Если вы устанавливаете коммуникационную нагрузку в 50 %, то это не удваивает, а более чем удваивает первоначальное время исполнения, дополнительное увеличение времени зависит от используемого CPU. Это иллюстрируется примером, основанным на наихудшей ситуации.

Случай 1:

- Время работы операционной системы составляет 250 мс на 1 секунду времени цикла.
- Программа пользователя имеет время исполнения 750 мс.
- Нагрузка на цикл, вызванная коммуникациями, тестовыми функциями и подготовительными функциями, составляет 0 %.

Цикл может быть представлен в упрощенной форме следующим образом:

Случай 2:

- Время работы операционной системы продолжает составлять 250 мс на 1 секунду времени цикла.
- Программа пользователя продолжает работать 750 мс.
- Нагрузка на цикл, вызванная коммуникациями, тестовыми функциями и подготовительными функциями, теперь установлена на 50 %.

Временная последовательность будет теперь следующей:

Таким образом, загрузка этапа выполнения коммуникациями составляет 1500 мс на цикл.

В этом примере установка коммуникационной нагрузки на 50 % увеличивает время цикла с 1 секунды до 3 секунд, иными словами, общее время цикла утраивается.

Коммуникационная нагрузка должна приниматься в расчет при установке минимального времени цикла, иначе возникнут ошибки, связанные со временем.

4.2.4.2 Функции (FC)

Функции (FC) относятся к блокам, которые вы программируете сами. Функция – это логический блок "с памятью". Временные переменные, принадлежащие FC, хранятся в стеках локальных данных. После того как FC выполнена, эти данные теряются. Чтобы хранить эти данные постоянно, функции могут также использовать разделяемые (глобальные) блоки данных.

Так как FC не имеет собственной памяти, то вы всегда должны определять для нее фактические параметры. Для локальных данных FC нельзя назначать начальные значения.

Применение

FC содержит программную секцию, которая выполняется всегда, когда FC вызывается другим логическим блоком. Функции можно использовать для следующих целей:

- для возврата значения функции в вызывающий блок (пример: математические функции)
- для выполнения технологической функции (пример: отдельная функция управления с битовой логической операцией).

Назначение фактических параметров формальным параметрам

Формальный параметр – это макет для "фактического" параметра. Фактические параметры заменяют формальные параметры при вызове функции. Вы всегда должны ставить в соответствие фактические параметры формальным параметрам FC (например, фактический параметр "E3.6" формальному параметру "Start"). Входные, выходные параметры и параметры типа вход/выход, используемые FC, хранятся как указатели на фактические параметры логического блока, который вызвал FC.

4.2.4.3 Функциональные блоки (FB)

Функциональные блоки (FB) относятся к блокам, которые вы программирует сами. Функциональный блок – это логический блок "с памятью". В качестве памяти ему назначается блок данных (экземплярный блок данных). В экземплярном DB сохраняются параметры, передаваемые FB, и статические переменные. Временные переменные хранятся в стеке локальных данных.

Данные, сохраняемые в экземплярном DB, не теряются, когда исполнение FB завершено. Однако, данные, сохраняемые в стеке локальных данных, теряются, теряются, когда исполнение FB завершено.

Замечание

Во избежание ошибок при работе с FB прочтите раздел Допустимые типы данных при передаче параметров в Приложении.

Применение

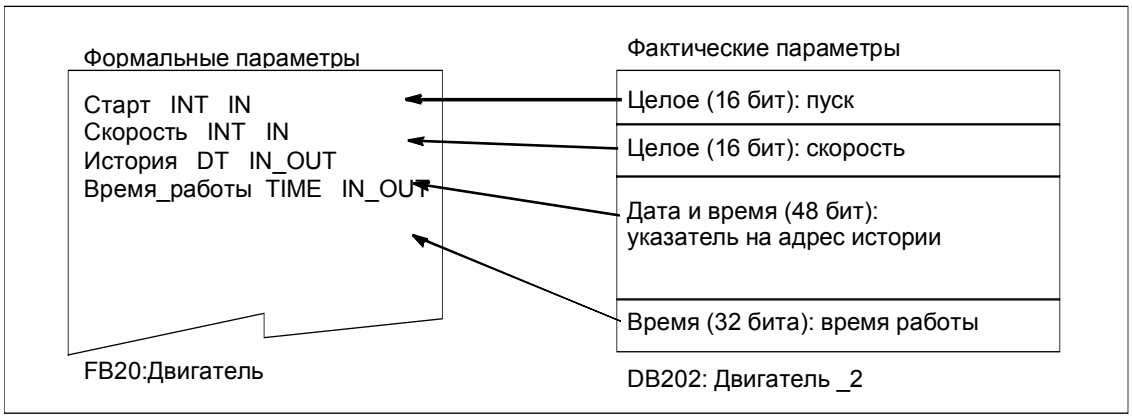
FB содержит программу, которая исполняется всегда, когда FB вызывается другим логическим блоком. Функциональные блоки значительно облегчают программирование часто встречающихся сложных функций.

Функциональные блоки и экземплярные блоки данных

Экземплярный блок данных назначается каждому вызову функционального блока, который передает параметры.

Вызывая более одного экземпляра FB, можно с помощью одного FB управлять более чем одним устройством. Например, FB для некоторого класса двигателей может управлять различными двигателями, используя различные наборы экземпляров данных для разных двигателей. Данные для каждого двигателя (например, скорость, накопленное время работы и т. д.) могут быть сохранены в одном или нескольких экземплярных DB.

На следующем рисунке показаны формальные параметры FB, который использует фактические параметры, сохраненные в экземплярном DB.



Переменные, имеющие тип данных FB

Если ваша пользовательская программа структурирована таким образом, что некоторый FB содержит вызовы других, уже существующих функциональных блоков, вы можете включить FB, подлежащие вызову, в таблицу писания переменных вызывающего блока в качестве статических переменных, имеющих тип данных FB. Этот способ позволяет вам "вкладывать" переменные и концентрировать экземпляры данных в одном блоке данных (мультиэкземплярном).

Назначение фактических параметров формальным параметрам

В STEP 7 в общем случае нет необходимости назначать фактические параметры формальным параметрам FB. Однако имеется исключение из этого правила. Фактические параметры должны быть назначены в следующих ситуациях:

- для параметра вход/выход (in/out) сложного типа данных (например, STRING, ARRAY или DATE_AND_TIME)
- для всех параметрируемых типов (например, TIMER, COUNTER или POINTER)

STEP 7 назначает фактические параметры формальным параметрам FB следующим образом:

- *Если вы указываете фактические параметры в операторе вызова:* команды FB используют предоставленные фактические параметры.
- *Если вы не указываете фактические параметры в операторе вызова:* команды FB используют значение, сохраненное в экземплярном DB.

В следующей таблице показано, каким переменным FB должны быть назначены фактические параметры.

	Тип данных		
Переменная	**Элементарный тип данных**	**Составной тип данных**	**Параметрируемый тип**
Вход	Параметр не нужен	Параметр не нужен	Нужен фактический параметр
Выход	Параметр не нужен	Параметр не нужен	Нужен фактический параметр
Вход/выход	Параметр не нужен	Нужен фактический параметр	–

Присвоение начальных значений формальным параметрам

Вы можете присвоить начальные значения формальным параметрам в разделе описаний FB. Эти значения записываются в экземплярный DB, связанный с FB.

Вам нет необходимости назначать фактические параметры формальным параметрам в операторе вызова, STEP 7 использует значения, сохраненные в экземплярном DB. Этими значениями также могут быть начальные значения, которые были введены в таблицу описания переменных FB.

В следующей таблице показано, каким переменным может быть присвоено начальное значение. Так как временные данные теряются после выполнения блока, то им нельзя назначить никаких значений.

Переменная	**Тип данных**		
	Элементарный тип данных	**Составной тип данных**	**Параметрируемый тип**
Вход	Начальное значение разрешено	Начальное значение разрешено	–
Выход	Начальное значение разрешено	Начальное значение разрешено	–
Вход/выход	Начальное значение разрешено	–	–
Статическая	Начальное значение разрешено	Начальное значение разрешено	–
Временная	–	–	–

4.2.4.4 Экземплярные блоки данных

Экземплярный блок данных назначается каждому вызову функционального блока, который передает параметры. Фактические параметры и статические данные FB хранятся в экземплярном DB. Переменные, описанные в FB, определяют структуру экземплярного блока данных. Экземпляр означает вызов функционального блока. Если, например, функциональный блок вызывается в программе пользователя S7 пять раз, то имеется пять экземпляров этого блока.

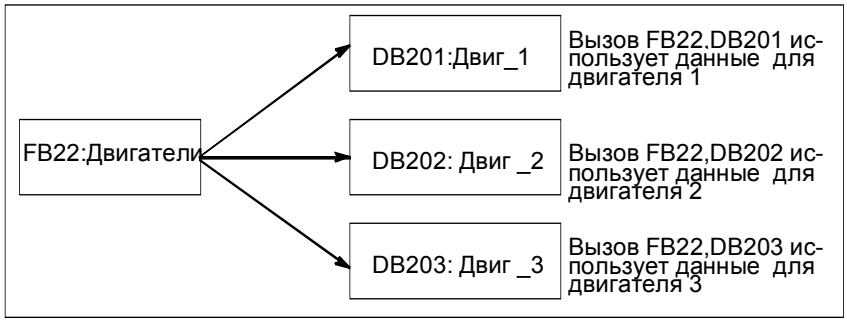
Создание экземплярного DB

Перед созданием экземплярного блока данных соответствующий FB уже должен существовать. При создании экземплярного блока данных указывается номер FB.

Один экземплярный DB для каждого отдельного экземпляра

При назначении функциональному блоку (FB), который управляет двигателем, нескольких экземплярных блоков данных этот FB можно использовать для управления разными двигателями.

Данные для каждого конкретного двигателя (например, скорость, время пуска, общее время работы) сохраняются в разных экземплярных блоках данных. DB, связываемый с FB при вызове последнего, определяет, какой из двигателей управляется. При использовании этого метода для нескольких двигателей нужен только один функциональный блок (см. следующий рисунок).

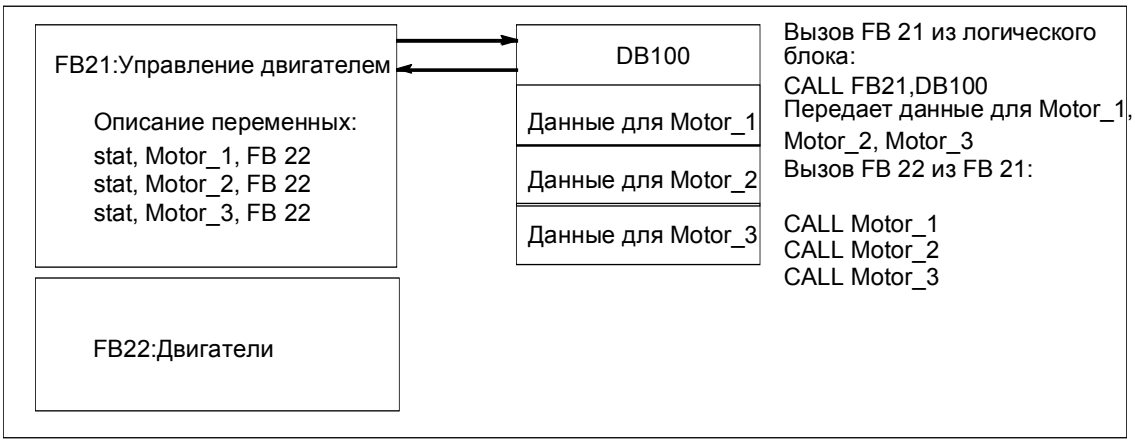


Один экземплярный DB для нескольких экземпляров FB (мультиэкземпляры)

Вы можете также передать экземпляры данных для нескольких двигателей одновременно в один экземплярный DB. Чтобы сделать это, вы должны запрограммировать вызовы для устройств управления двигателями еще в одном FB и описать статические переменные, имеющие тип данных FB, для отдельных экземпляров в разделе описаний вызывающего FB.

Если вы используете один экземплярный DB для нескольких экземпляров FB, то вы экономите память и оптимизируете использование блоков данных.

На следующем рисунке вызывающим FB является FB21 "Управление двигателем", переменные имеют тип данных FB22, а экземпляры определяются посредством Motor_1, Motor_2 и Motor_3.



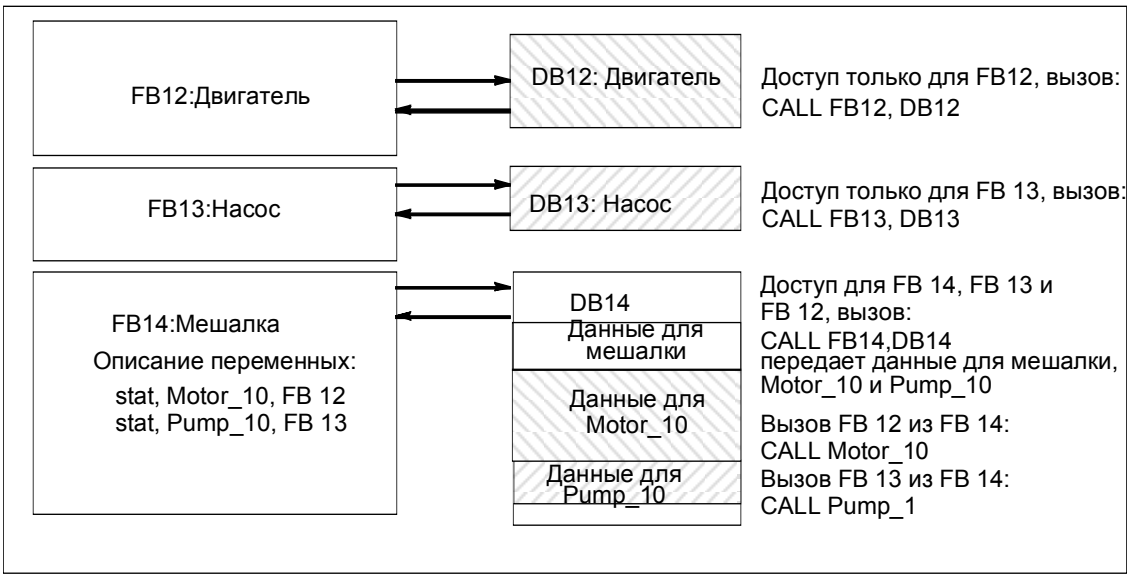
В этом примере FB22 не нуждается в собственном экземплярном блоке данных, так как данные его экземпляров сохраняются в экземплярном блоке данных вызывающего FB.

Один экземплярный DB для нескольких экземпляров различных FB (мультиэкземпляры)

В функциональном блоке вы можете вызывать экземпляры других существующих FB. Экземплярные данные, необходимые для этого, вы можете назначить экземплярному блоку данных вызывающего FB, то есть в этом случае у вас нет необходимости ни в каких дополнительных блоках данных для вызываемых FB.

Для этих мультиэкземпляров в одном экземплярном блоке данных вы должны описать статические переменные, имеющие тип данных вызываемого функционального блока, для каждого отдельного экземпляра в разделе описаний вызывающего функционального блока. Тогда вызов внутри функционального блока требует не экземплярного блока данных, а только символического имени переменной.

В примере на этом рисунке назначенные экземпляры данных хранятся в общем экземплярном DB.



4.2.4.5 Совместно используемые блоки данных (DB)

В отличие от логических блоков, блоки данных не содержат команд STEP 7. Они используются для хранения данных пользователя, иными словами, боки данных содержат переменные данные, с которыми работает программа пользователя. Совместно используемые блоки данных применяются для хранения пользовательских данных, к которым могут обратиться все остальные блоки.

Размер DB может изменяться. За информацией о максимально возможном размере обратитесь к описанию вашего CPU.

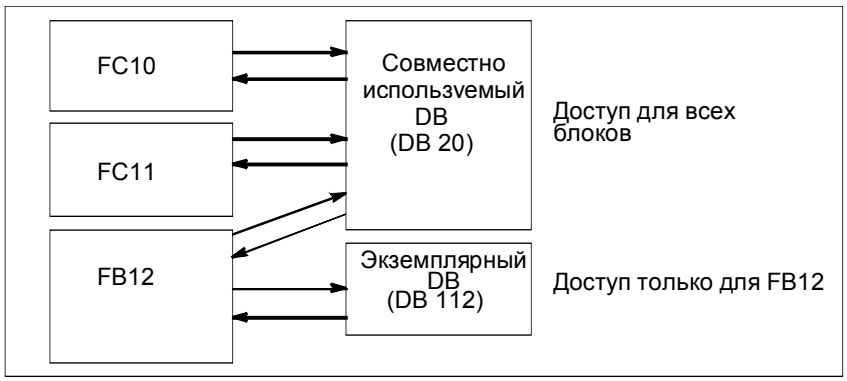
Вы можете структурировать совместно используемые блоки данных любым способом для удовлетворения своих конкретных потребностей.

Совместно используемые блоки данных в программе пользователя

Если логический блок (FC, FB или OB) вызывается, то он временно занимает место в области локальных данных (L-стек). В дополнение к этой области локальных данных логический блок может открыть область памяти в виде DB. В отличие от данных, находящихся в области локальных данных, данные в DB не удаляются, когда DB закрывается, иначе говоря, после исполнения соответствующего логического блока.

Каждый FB, FC или OB может читать данные из совместно используемого DB или записывать данные в этот DB. Эти данные сохраняются в DB после выхода из него.

Совместно используемый и экземплярный DB могут быть открыты одновременно. На следующем рисунке показаны различные методы доступа к блокам данных.



4.2.4.6 Системные функциональные блоки (SFB) и системные функции (SFC)

Предварительно запрограммированные блоки

У вас нет необходимости программировать каждую функцию самостоятельно. CPU S7 предоставляют в ваше распоряжение заранее запрограммированные блоки, которые вы можете вызывать в своей пользовательской программе.

Дополнительную информацию можно найти в справочнике по системным блокам и системным функциям (перейдите к Описаниям языков и к Помощи по блокам и системным атрибутам).

Системные функциональные блоки

Системный функциональный блок (SFB) – это функциональный блок, встроенный в CPU S7. SFB являются частью операционной системы и не загружаются как часть программы пользователя. Как и FB, SFB – это блоки "с памятью". Для SFB тоже нужно создавать экземплярные блоки данных и загружать их в CPU как часть программы.

CPU S7 предоставляют в распоряжение следующие SFB:

- для связи через проектируемые соединения
- для встроенных специальных функций (например, SFB29 ”HS_COUNT" на CPU 312 IFM и CPU 314 IFM).

Системные функции

Системная функция – это заранее запрограммированная, оттестированная функция, встроенная в CPU S7. Вы можете вызвать SFC в своей программе. SFC являются частью операционной системы и не загружаются как часть программы. Как и FC, SFC являются блоками "без памяти".

CPU S7 предоставляют SFC для следующих функций:

- копирование и блочные функции
- контроль программы
- работа с часами и счетчиками рабочего времени
- передача наборов данных
- передача событий из CPU всем остальным CPU в мультипроцессорном режиме
- обработка прерываний по времени и с задержкой
- обработка синхронных и асинхронных ошибок
- информация о статических и динамических системных данных, например, диагностика
- обновление образа процесса и обработка битовых массивов
- адресация модулей
- децентрализованная периферия
- связь с помощью глобальных данных
- связь через неспроектированные соединения
- генерирование сообщений, относящихся к блокам

Дополнительная информация

За более подробной информацией о SFB и SFC обращайтесь к справочному руководству "Системное программное обеспечение S7-300/400. Системные и стандартные функции". В книгах "Программируемые контроллеры S7-300. Аппаратное обеспечение и руководство по монтажу" и "Система автоматизации S7-400, M7-400. Данные модулей. Справочное руководство" объясняется, какие SFB и SFC доступны.

4.2.5 Организационные блоки для обработки программ, управляемой прерываниями

4.2.5.1 Организационные блоки для обработки программ, управляемой прерываниями

Предоставляя OB прерываний, CPU S7 обеспечивают следующие возможности:

- секции программы могут исполняться в определенные моменты времени или через определенные интервалы (управление по времени)
- ваша программа может реагировать на внешние сигналы от процесса

Циклическая программа пользователя не должна запрашивать, произошли или нет события прерывания. Если прерывание происходит, то операционная система обеспечивает, чтобы программа исполнялась пользователя в OB прерываний, так что имеет место запрограммированная реакция на прерывание со стороны программируемого логического контроллера.

Типы и применения прерываний

Следующая таблица показывает, как могут быть использованы различные типы прерываний.

Тип прерывания	OB прерываний	Примеры применения
Прерывание по времени	OB10 – OB17	Расчет общего расхода жидкости в процессе смешивания к концу смены
Прерывание с задержкой	OB20 – OB23	Управление вентилятором, который должен продолжать работать в течение 20 с после выключения двигателя
Циклическое прерывание	OB30 – OB38	Опрос уровня сигнала для системы управления по замкнутому контуру
Аппаратное прерывание	OB40 – OB47	Сообщение о достижении максимального уровня в резервуаре

4.2.5.2 Организационные блоки прерываний по времени (OB10 – OB17)

CPU S7 предоставляют в распоряжение OB прерываний по времени суток, которые могут исполняться в указанный день или через определенные интервалы времени.

Прерывания по времени суток могут запускаться следующим образом:

- однократно в определенное время (указанное в абсолютной форме с датой)
- периодически при указании стартового времени и интервала, с которым прерывание должно повторяться (например, ежеминутно, ежечасно, ежедневно).

Правила для прерываний по времени

Прерывания по времени могут исполняться только тогда, когда прерыванию были назначены параметры и в программе пользователя существует соответствующий организационный блок. Если это не так, в диагностический буфер вносится сообщение об ошибке и выполняется обработка асинхронной ошибки (OB80, см. Организационные блоки обработки ошибок (OB70 – OB87 / OB121 – OB122)).

Периодические прерывания по времени должны соответствовать реальной дате. Повторение OB10 ежемесячно, начиная с 31 января, невозможно. В этом случае OB будет запускаться только в месяцы, имеющие 31 день.

Прерывание по времени, активизированное во время запуска (теплого или горячего рестарта) исполняется только после завершения запуска.

OB прерываний по времени, отмененные при назначении параметров, не могут быть запущены. CPU распознает ошибку программирования и переходит в STOP.

После теплого рестарта прерывания по времени должны быть установлены заново (например, с помощью SFC30 ACT_TINT в программе запуска).

Запуск прерывания по времени

Чтобы дать CPU возможность запустить прерывание по времени, вы должны сначала установить, а затем активизировать это прерывание. Существуют три способа запуска этого прерывания:

- автоматический запуск прерывания по времени путем назначения подходящих параметров с помощью STEP 7 (блок параметров ”time-of-day interrupts [прерывания по времени]")
- установка и активизация прерывания по времени с помощью SFC28 SET_TINT и SFC30 ACT_TINT из программы пользователя
- установка прерывания по времени interrupt путем назначения параметров с помощью STEP 7 и его активизация с помощью SFC30 ACT_TINT в программе пользователя.

Опрос прерываний по времени

Чтобы запросить, какие прерывания по времени установлены и на какое время они настроены, вы можете выполнить одно из следующих действий:

- вызвать SFC31 QRY_TINT
- запросить список ”interrupt status [состояние прерываний]" из списка состояний системы.

Деактивизация прерываний по времени

Вы можете деактивировать еще не исполненные прерывания по времени с помощью SFC29 CAN_TINT. Деактивированные прерывания по времени могут быть снова установлены с помощью SFC28 SET_TINT и активизированы с помощью SFC30 ACT_TINT.

Приоритет OB прерываний по времени

Все восемь OB прерываний по времени по умолчанию имеют один и тот же класс приоритета (2) и поэтому обрабатываются в порядке возникновения событий запуска. Однако вы можете изменить этот класс приоритета выбором подходящих параметров.

Изменение установленного времени

Вы можете изменить время, установленное для прерывания следующим образом:

- мастер часов синхронизирует время для master- и slave-устройств
- для установки нового времени в программе пользователя может быть вызван SFC0 SET_CLK.

Реакция на изменение времени

Следующая таблица показывает, как прерывания по времени реагируют на изменение времени.

Если...	То...
время было передвинуто вперед и одно или более прерываний по времени были пропущены,	запускается OB80, и пропущенные прерывания по времени вводятся в стартовую информацию OB80.
вы деактивировали пропущенные прерывания по времени в OB80,	пропущенные прерывания по времени более не исполняются.
вы не деактивировали пропущенные прерывания по времени в OB80,	первое пропущенное прерывание по времени выполняется, а остальные игнорируются.
в результате сдвига времени назад какое-либо из событий для прерываний по времени происходит снова,	выполнение прерывания по времени не повторяется.

4.2.5.3 Организационные блоки прерываний с задержкой (OB20 – OB23)

CPU S7 предоставляют в распоряжение OB прерываний с задержкой, с помощью которых вы можете программировать отложенное исполнение частей своей пользовательской программы.

Правила для прерываний с задержкой

Прерывания с задержкой могут исполняться только при наличии соответствующего организационного блока в программе CPU. Если это не так, то в диагностический буфер вносится сообщение об ошибке и производится обработка асинхронной ошибки (OB80, Организационные блоки обработки ошибок (OB70 – OB87 / OB121 – OB122)).

OB прерываний с задержкой, отмененные при назначении параметров, не могут быть запущены. CPU распознает ошибку программирования и переходит в STOP.

Прерывания с задержкой запускаются, когда истекло время, указанное в SFC32 SRT_DINT.

Запуск прерывания с задержкой

Для запуска прерывания с задержкой вы должны указать в SFC32 время задержки, по истечении которого вызывается соответствующий OB прерываний с задержкой. За информацией о максимально разрешенной величине задержки времени обратитесь к литературе "Программируемые контроллеры S7-300. Аппаратное обеспечение и руководство по монтажу" и "Система автоматизации S7-400, M7-400. Данные модулей. Справочное руководство".

Приоритет OB прерываний с задержкой

По умолчанию для OB прерываний с задержкой установлены классы приоритета с 3 по 6. Вы можете назначить параметры для изменения классов приоритета.

4.2.5.4 Организационные блоки циклических прерываний (OB30 – OB38)

CPU S7 предоставляют в распоряжение OB циклических прерываний, которые прерывают циклическую обработку программы через определенные интервалы.

Циклические прерывания запускаются через определенные интервалы. Временем, от которого начинается отсчет интервалов, является переход из режима STOP в RUN.

Правила для циклических прерываний

При определении интервалов убедитесь, что между стартовыми событиями отдельных циклических прерываний имеется достаточно времени для обработки самих циклических прерываний.

Если вы назначаете параметры для отмены OB циклических прерываний, то они не могут быть более запущены. CPU распознает ошибку программирования и переходит в STOP.

Запуск циклического прерывания

Для запуска циклического прерывания вы должны с помощью STEP 7 указать интервал в блоке параметров циклических прерываний. Этот интервал всегда представляет собой целое кратное от базового тактового интервала в 1 мс.

Интервал = n x базовый тактовый интервал в 1 мс.

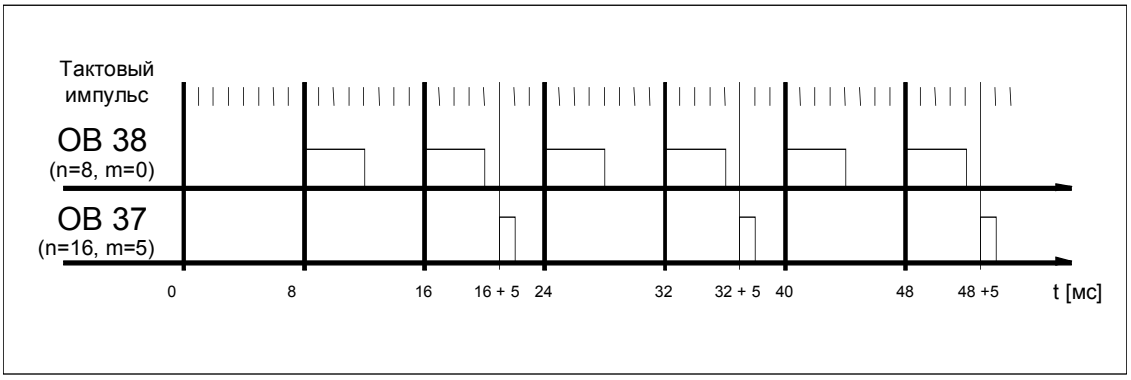
Каждый из восьми доступных OB циклических прерываний имеет интервал по умолчанию (см. следующую таблицу). Интервал по умолчанию становится действенным, когда загружается назначенный ему OB циклических прерываний. Однако вы можете назначить параметры для изменения значений по умолчанию. Для получения информации о верхнем пределе обратитесь к руководствам "Программируемые контроллеры S7-300. Аппаратное обеспечение и руководство по монтажу" и "Система автоматизации S7-400, M7-400. Данные модулей. Справочное руководство".

Сдвиг фазы в циклических прерываниях

Во избежание одновременного запуска циклических прерываний от различных OB и возможного при этом появления временной ошибки (превышение времени цикла) вы можете указать сдвиг фазы. Сдвиг фазы гарантирует, что исполнение циклического прерывания откладывается на определенное время после истечения интервала.

Сдвиг фазы = m x базовый тактовый интервал (где $0 \leq m < n$)

На следующем рисунке показано, как исполняется OB циклических прерываний с фазовым сдвигом (OB37) в сравнении с циклическим прерыванием без фазового сдвига (OB38).



Приоритет OB циклических прерываний

В следующей таблице показаны интервалы по умолчанию и классы приоритета OB циклических прерываний. Вы можете назначить параметры для изменения интервала и класса приоритета.

OB циклических прерываний	Интервал в мс	Класс приоритета
OB30	5000	7
OB31	2000	8
OB32	1000	9
OB33	500	10
OB34	200	11
OB35	100	12
OB36	50	13
OB37	20	14
OB38	10	15

4.2.5.5 Организационные блоки аппаратных прерываний (OB40 – OB47)

CPU S7 предоставляет в распоряжение OB аппаратных прерываний, которые реагируют на сигналы от модулей (например, сигнальных модулей (SM), коммуникационных процессоров (CP), функциональных модулей (FM)). С помощью STEP 7 вы можете установить, какой сигнал от параметрируемого цифрового или аналогового модуля запускает OB. В случае CP и FM используйте соответствующие диалоговые окна для назначения параметров.

Аппаратные прерывания запускаются, когда сигнальный модуль, способный на аппаратные прерывания и с разрешенным аппаратным прерыванием, передает полученный от процесса сигнал на CPU или когда функциональный модуль CPU сигнализирует о прерывании.

Правила для аппаратных прерываний

Аппаратные прерывания могут быть исполнены только в том случае, если в программе CPU имеется соответствующий организационный блок. Если это не так, в диагностический буфер вносится сообщение об ошибке и выполняется обработка асинхронной ошибки (OB80, см. Организационные блоки обработки ошибок (OB70 – OB87 / OB121 – OB122)).

Если вы при назначении параметров отменили OB аппаратных прерываний, то они не могут быть более запущены. CPU распознает ошибку программирования и переходит в STOP

Назначение параметров сигнальным модулям, способным к аппаратным прерываниям

Каждый канал сигнального модуля, способного к аппаратным прерываниям, может запустить аппаратное прерывание. Поэтому в наборах параметров сигнальных модулей, способных к прерываниям, вы должны с помощью STEP 7 указать следующее:

- Чем будет запускаться аппаратное прерывание.
- Какой OB аппаратных прерываний будет исполняться (по умолчанию для исполнения всех аппаратных прерываний назначается OB40).

С помощью STEP 7 активизируется генерирование аппаратных прерываний на функциональных модулях. Остальные параметры вы назначаете в диалоговых окнах для назначения параметров этих функциональных модулей.

Приоритет OB аппаратных прерываний

По умолчанию приоритет для OB аппаратных прерываний относится к классу приоритета от 16 до 23. Вы можете назначить параметры для изменения классов приоритета.

4.2.5.6 Организационные блоки запуска (OB100 / OB101 / OB102)

Типы запуска

Имеется три различных типа запуска:

- горячий рестарт (отсутствует в S7-300 и S7-400H)
- теплый рестарт
- холодный рестарт

В следующей таблице показано, какие OB вызывает операционная система при каждом типе запуска:

Тип запуска	Соответствующий OB
Горячий рестарт	OB101
Теплый рестарт	OB100
Холодный рестарт	OB102

Стартовые события для OB запуска

CPU выполняет запуск после следующих событий:

- после включения питания
- после перевода переключателя режимов из STOP в RUN/RUN-P
- после запроса от коммуникационной функции
- после синхронизации в мультипроцессорном режиме
- в H-системе после установления связи (только на резервном CPU)

В зависимости от стартового события, используемого CPU, и его установленных параметров вызывается соответствующий OB запуска (OB100, OB101 или OB102).

Программа запуска

Вы можете указать условия для запуска своего CPU (инициализирующие значения для RUN, пусковые значения для модулей ввода/вывода) путем записи своей программы запуска в организационный блок OB100 для теплого рестарта, OB101 для горячего рестарта или OB102 для холодного рестарта.

Нет ограничений по длине и времени выполнения программы запуска, так как контроль цикла еще не активен. В стартовой программе невозможно исполнение под управлением времени или под управлением прерываний. Во время запуска все цифровые выходы имеют сигнальное состояние 0.

Тип запуска после ручного рестарта

На CPU S7-300 возможен только ручной теплый или холодный рестарт (только CPU 318-2).

На некоторых CPU S7-400 вы можете вручную выполнять запуск с помощью переключателя режимов и переключателя типа запуска (CRST/WRST), если это разрешено при назначении параметров, которое вы выполнили с помощью STEP 7. Ручной теплый запуск возможен без специального назначения параметров.

Тип запуска после автоматического рестарта

На CPU S7-300 после включения питания возможен только теплый рестарт.

На CPU S7-400 вы можете указать, ведет ли автоматический запуск после включения питания к теплому или к горячему рестарту.

Очистка образа процесса

Когда перезапускается CPU S7-400, исполняется оставшаяся часть цикла, а таблица образа процесса на выходах по умолчанию очищается. Вы можете предотвратить очистку образа процесса, если хотите, чтобы программа пользователя продолжала работать со старыми значениями после перезапуска.

Контроль существования и типа модулей

При установке параметров вы можете решить, будут ли проверяться перед запуском модули в конфигурационной таблице, чтобы убедиться, что они существуют и что тип модуля совпадает с заданным.

Если контроль модулей активизирован, то CPU не запустится при обнаружении расхождений между конфигурационной таблицей и фактической конфигурацией.

Времена контроля

Чтобы убедиться, что программируемый контроллер запускается без ошибок, вы можете выбрать следующие времена контроля:

- максимально допустимое время для передачи параметров модулям
- максимально допустимое время для того, чтобы модули могли сообщить о своей готовности к работе после включения питания
- на CPU S7-400 максимальное время прерывания, после которого разрешен горячий рестарт.

Как только времена контроля истекают, CPU или переходит в STOP, или возможен только теплый рестарт.

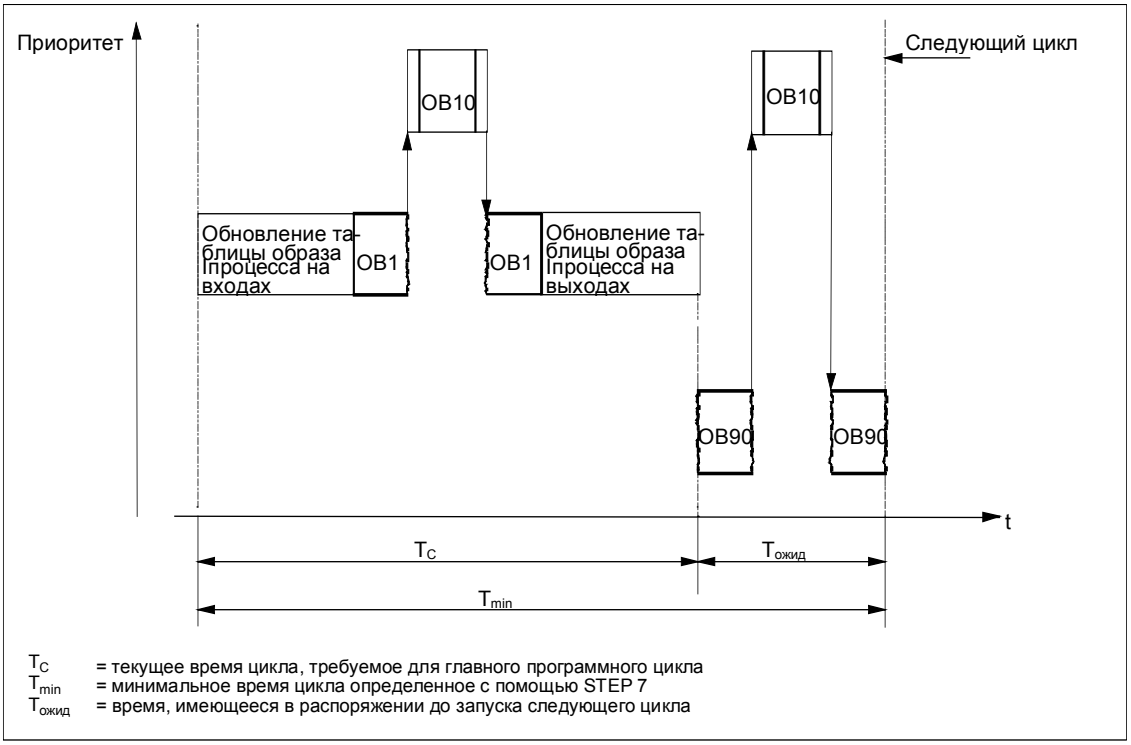
4.2.5.7 Фоновый организационный блок (OB90)

Если вы определили минимальное время цикла с помощью STEP 7, и это время больше, чем время текущего цикла, то у CPU в конце циклической программы еще есть время для работы. Это время используется для исполнения фонового OB. Если в вашем CPU OB 90 не существует, то CPU ждет, пока не истечет указанное минимальное время цикла. Следовательно, вы можете использовать OB90 для процессов, в которых время не является критическим для работы, и, таким образом, избежать потерь времени на ожидание.

Приоритет фонового OB

Фоновый OB относится к классу приоритета 29, что соответствует приоритету 0,29. Таким образом, этот OB имеет самый низкий приоритет. Его класс приоритета не может быть изменен при назначении параметров.

На следующем рисунке показан пример обработки фонового цикла, свободного цикла и OB10 (в существующих CPU).



T_C = текущее время цикла, требуемое для главного программного цикла
T_{min} = минимальное время цикла определенное с помощью STEP 7
$T_{ожид}$ = время, имеющееся в распоряжении до запуска следующего цикла

Программирование OB90

Время работы OB90 не контролируется операционной системой CPU, так что вы можете программировать в OB90 циклы любой длины. Обеспечьте непротиворечивость данных, используемых вами в фоновой программе, принимая во внимание следующее:

- события сброса OB90 (см. справочное руководство "Системное программное обеспечение для S7-300 и S7-400, Системные и стандартные функции")
- обновление образа процесса асинхронно по отношению к OB90.

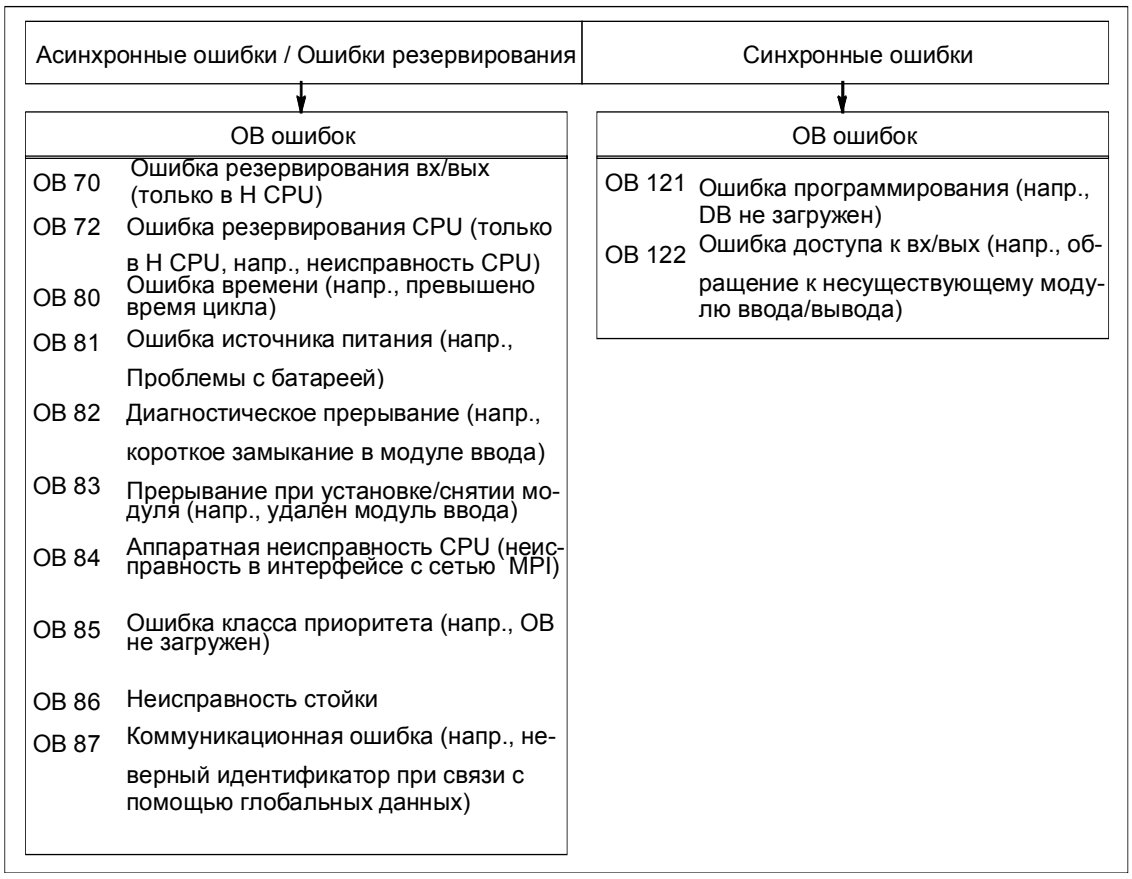
4.2.5.8 Организационные блоки обработки ошибок (OB70 – OB87 / OB121 – OB122)

Типы ошибок

Ошибки, которые могут быть обнаружены CPU S7 и на которые вы можете реагировать с помощью организационных блоков, можно разделить на две основные категории:

- Синхронные ошибки: эти ошибки могут быть поставлены в соответствие конкретной части программы пользователя. Эта ошибка происходит во время выполнения конкретной команды. Если соответствующий OB синхронных ошибок не загружен, то CPU при возникновении ошибки переходит в STOP.
- Асинхронные ошибки: эти ошибки не могут быть непосредственно поставлены в соответствие исполняемой программе пользователя. Это ошибки класса приоритета, неисправности программируемого логического контроллера (например, дефектный модуль) или ошибки резервирования. Если соответствующий OB асинхронных ошибок не загружен, то CPU при возникновении ошибки переходит в STOP (исключения: OB70, OB72, OB81).

На следующем рисунке показаны типы ошибок, которые могут возникнуть, разделенные на категории в соответствии с OB ошибок.



Использование OB для синхронных ошибок

Синхронные ошибки возникают при исполнении конкретной команды. Когда эти ошибки происходят, операционная система делает запись в стек прерываний (I-стек) и запускает OB для синхронных ошибок.

OB ошибок, вызванные как результат синхронных ошибок, исполняются как часть программы в том же классе приоритета, что и блок, который исполнялся, когда ошибка была обнаружена. Поэтому OB121 и OB122 могут обращаться к тем значениям в аккумуляторах и других регистрах, которые в них были во время возникновения прерывания. Вы можете использовать эти значения для реагирования на сбойную ситуацию, а затем вернуться к обработке своей программы (например, если происходит ошибка доступа на аналоговом модуле ввода, вы можете указать заменяющее значение в OB122 с помощью SFC44 RPL_VAL). Однако локальные данные OB ошибок требуют дополнительного места в L-стеке этого класса приоритета.

В CPU S7-400 один OB синхронных ошибок может запустить другой OB синхронных ошибок. В CPU S7-300 это невозможно.

Использование OB асинхронных ошибок

Если операционная система CPU обнаруживает асинхронную ошибку, то она запускает соответствующий OB ошибок (OB70 – OB72 и OB80 – OB87). OB для асинхронных ошибок имеют наивысший приоритет и не могут быть прерваны другими OB, если все OB асинхронных ошибок имеют одинаковый приоритет. Если более одного OB асинхронных ошибок с одинаковым приоритетом появляются одновременно, то они обрабатываются в том порядке, как они появились.

Маскирование стартовых событий

С помощью системных функций (SFC) вы можете замаскировать, отложить или заблокировать стартовые события для нескольких OB. За более подробной информацией об этих SFC и организационных блоках обратитесь к справочному руководству "Системное программное обеспечение для S7-300 и S7-400. Системные и стандартные функции".

Тип OB ошибок	SFC	Функция SFC
OB синхронных ошибок	SFC36 MSK_FLT	Маскирует отдельные синхронные ошибки. Замаскированные не запускают OB ошибок и запрограммированные реакции
	SFC37 DMSK_FLT	Демаскирует синхронные ошибки
OB асинхронных ошибок	SFC39 DIS_IRT	Блокирует все прерывания и асинхронные ошибки. Блокированные ошибки не запускают OB ошибок ни в одном из последующих циклов CPU и не запускают запрограммированные реакции
	SFC40 EN_IRT	Разблокирует прерывания и асинхронные ошибки
	SFC41 DIS_AIRT	Откладывает прерывания более высокого приоритета и асинхронные ошибки до конца OB
	SFC42 EN_AIRT	Разблокирует прерывания более высокого приоритета и асинхронные ошибки

Замечание

Если вы хотите, чтобы прерывания игнорировались, то более эффективно заблокировать их с помощью SFC, чем загружать пустой OB (содержащий BE).

5 Запуск и функционирование

5.1 Запуск STEP 7

5.1.1 Запуск STEP 7

 При запуске Windows 95/98/NT вы найдете пиктограмму для SIMATIC Manager, являющуюся стартовой точкой для программного обеспечения STEP 7 на интерфейсе Windows.

Самый быстрый способ запуска STEP 7 – позиционирование курсора на этой пиктограмме и двойной щелчок на ней. После этого открывается окно, содержащее SIMATIC Manager. Отсюда вы можете получить доступ ко всем функциям, которые вы установили для стандартного пакета и для любых дополнительных пакетов.

В качестве альтернативы вы можете запустить SIMATIC Manager через кнопку " Start [Пуск]" на панели задач в Windows 95/98/NT. Вход вы найдете в позиции "Simatic".

Замечание

Больше информации о стандартных операциях и опциях Windows вы найдете в руководстве пользователя Windows или в оперативном справочнике Windows 95/98/NT.

SIMATIC Manager

SIMATIC Manager - это основное приложение для проектирования и программирования. В нем вы можете реализовать следующие функции:

- создавать проекты
- конфигурировать и назначать параметры аппаратным средствам
- конфигурировать аппаратные сети
- программировать блоки
- отлаживать и принимать в эксплуатацию свои программы

Доступ к различным функциям спроектирован объектно-ориентированным, интуитивно понятным и легким для изучения.

Вы можете работать с SIMATIC Manager'ом одним из двух способов:

- Offline, без подключения программируемого контроллера
- Online, с подключенным программируемым контроллером

Обратите внимание на соответствующие указания по безопасности в каждом случае.

Как действовать, начиная отсюда

Задачи автоматизации создаются в виде "проектов". Вам будет легче это проделать, если перед началом работы вы прочтете следующие основные темы:

- пользовательский интерфейс
- некоторые основные рабочие этапы
- оперативная помощь

5.1.2 Запуск STEP 7 со стартовыми параметрами, используемыми по умолчанию

Начиная со STEP 7 версии 5.0, вы можете создать несколько символов в SIMATIC Manager и определить стартовые параметры в строке вызовов. Сделав это, вы сможете заставить SIMATIC Manager позиционироваться на объекте, описываемом этими параметрами. Это позволяет вам переходить к соответствующим местам в проекте немедленно, просто с помощью двойного щелчка.

Вызвав **s7tgtopx.exe**, вы можете указать следующие стартовые параметры:

/e <полный физический путь к проекту>

/o <логический путь для объекта, на котором вы хотите остановиться>

/h <идентификатор объекта> /onl или /off

Самый легкий путь для установки подходящих параметров описан ниже.

Установка параметров путем копирования и вставки

Действуйте следующим образом:

1. На своем рабочем столе создайте новую связь с файлом s7tgtopx.exe.
2. Выведите на экран диалоговое окно свойств.
3. Выберите закладку "Link [Связь]". Теперь вход "Target [Цель]" должен быть расширен следующим образом.
4. Выделите требуемый объект в SIMATIC Manager.
5. Скопируйте объект в буфер обмена с помощью комбинации клавиш CTRL+C.
6. Поместите курсор в конце входа "Target [Цель]" в закладке "Link [Связь]".
7. Вставьте содержимое буфера обмена с помощью комбинации клавиш CTRL+V.
8. Закройте диалоговое окно, щелкнув на "OK".

Пример параметров:

/e F:\SIEMENS\STEP7\S7proj\MyConfig\MyConfig.s7p

/o “1,8:MyConfig\SIMATIC 400(1)\CPU416-1\S7-Program(1)\Blocks\FB1”

/h T00112001;129;T00116001;1;T00116101;16e

Замечания о структуре пути к проекту

Путь к проекту – это физический путь в файловой системе. Нотация соглашения об универсальных именах UNC не поддерживается, так, например: F:\SIEMENS\STEP7\S7proj\MyConfig\MyConfig.s7p

Полный логический путь строится следующим образом:

[Идентификатор вида, идентификатор online]:имя проекта\{имя объекта\}*\ имя объекта

Пример: /o 1.8:MyConfig\SIMATIC 400(1)\CPU416-1\S7-Program(1)\Blocks\FB1

Замечания о структуре логического пути

Полный логический путь и идентификатор объекта могут быть созданы только с использованием функций копирования и вставки.

Однако можно указать и путь, который может быть прочитан пользователем. В вышеприведенном примере это выглядело бы так:

/o "MyConfig\SIMATIC 400(1)\CPU416-1\S7-Program(1)\Blocks\FB1”. Добавив /onl или /off, пользователь может указать, действителен ли этот путь в окне online или offline. Это не нужно указывать, если вы используете функции копирования и вставки.

Важно: Если путь содержит пробелы, то он должен быть помещен в кавычки.

5.1.3 Вызов функций помощи

Оперативная помощь

Система оперативной помощи предоставляет вам информацию в той точке, где вы можете использовать ее наиболее эффективно. Вы можете использовать оперативную помощь для доступа к информации быстро и непосредственно без необходимости поиска ее по руководствам. В оперативной помощи вы найдете следующие типы информации:

- **Contents [Содержание]**: предлагает ряд различных путей отображения справочной информации
- **Context-Sensitive Help [Контекстно-чувствительная помощь]** (клавиша F1): с помощью клавиши F1 вы получаете доступ к информации об объекте, который вы как раз выбрали с помощью мыши, или об активном диалоге или окне
- **Introduction [Введение]**: дает краткое введение в использование, основные характеристики и область функционирования приложения
- **Getting Started [Начало работы]**: дает обзор основных шагов, необходимых для начала работы с приложением
- **Using Help [Использование помощи]**: дает описание способов нахождения конкретной информации в системе оперативной помощи
- **About [О продукте]**: дает информацию о текущей версии приложения

Через меню Help [Помощь] вы можете также обратиться к темам, связанным с текущей диалоговой ситуацией, из любого окна.

Вызов оперативной помощи

Вы можете вызвать оперативную помощь одним из следующих способов:

- Выберите команду в меню Help [Помощь] в строке меню.
- Щелкните на кнопке ”Help" [Помощь] в диалоговом окне. Тогда вы получите помощь по этому диалоговому окну.
- Поместите курсор в окне или в диалоговом окне на теме, по которой вам нужна помощь, и нажмите клавишу F1 или выберите команду меню **Help > Context-Sensitive Help [Помощь > Контекстно-чувствительная помощь]**.
- Используйте курсор в виде вопросительного знака в Windows.

Последние три способа обращения к оперативной помощи известны как контекстно-чувствительная помощь.

Вызов быстрой помощи

Быстрая помощь на кнопках панели инструментов отображается, когда вы помещаете курсор на кнопке и оставляете его там на некоторое время.

Изменение размера шрифта

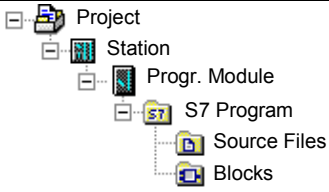
Используя команду меню **Options > Font** [**Параметры > Шрифт**] в окне помощи, вы можете установить размер шрифта на "Small [Мелкий]," ”Normal [Нормальный]" или ”Large [Крупный]".

5.2 Объекты и их иерархия

5.2.1 Объекты и их иерархия

Иерархия объектов для проектов и библиотек в STEP 7 отображается в SIMATIC Manager таким же образом, как Проводник Windows отображает структуру каталогов из папок и файлов.

На следующем рисунке показан пример иерархии объектов.

	Объект Проект Объект Станция Объект Программируемый модуль Объект Программа S7/M7 Объект Папка с исходными файлами Объект Папка с блоками

Объекты имеют следующие функции:

- носитель свойств объекта,
- папки,
- носитель функций (например, запуск конкретного приложения).

Объекты как носители свойств

Объекты могут быть носителями как функций, так и свойств (например, настроек). Когда вы выделяете объект, вы можете выполнить с ним одну из следующих функций:

- Редактировать объект, используя команду меню **Edit > Open Object [Редактировать > Открыть объект]**.
- Открыть диалоговое окно, используя команду меню **Edit > Object Properties** [**Редактировать > Свойства объекта**], и установить параметры, относящиеся к объекту.

Папка тоже может быть носителем свойств.

Замечание

Если вы хотите изменить настройки для объекта в устройстве программирования (такие как параметры модуля), то в программируемом контроллере они сначала не активны. Чтобы это произошло, в программируемый контроллер сначала должны быть загружены системные блоки данных, в которых эти настройки хранятся.

Если вы загружаете всю программу пользователя. То системные блоки данных загружаются автоматически как часть этого процесса. Если вы производите изменения в настройках после загрузки программы, то вы можете перезагрузить объект "system data [системные данные], чтобы обновить настройки в программируемом контроллере.

Объекты как папки

Папка (каталог) может содержать другие папки или объекты. Они отображаются, когда вы открываете папку.

Замечание

Мы настоятельно рекомендуем редактировать папки с помощью STEP 7, так как они могут быть физически структурированы иначе, чем компоновка, которую вы видите в SIMATIC Manager.

Объекты как носители функций

При открытии объекта появляется окно, в котором вы можете редактировать этот объект.

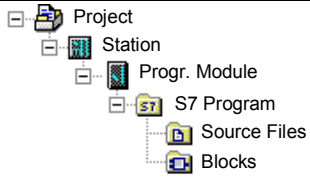
Объект является или папкой, или носителем функций. Исключением являются станции: они являются как папками (для программируемых модулей), так и носителями функций (используемых для конфигурирования аппаратуры).

- Если вы дважды щелкнете на станции, то отобразятся содержащиеся в ней объекты: программируемые модули и конфигурация станции (станция как папка).
- Если вы откроете станцию командой меню **Edit > Open Object [Редактировать > Открыть объект]**, то вы можете конфигурировать станцию и назначать ей параметры (станция как носитель функции). Эта команда меню имеет такой же эффект, как двойной щелчок на объекте "Hardware [Аппаратура]".

5.2.2 Объект Проект

Проект представляет собой совокупность всех данных и программ в решении задачи автоматизации и расположен в верхней части иерархии объектов.

Положение в изображении проекта



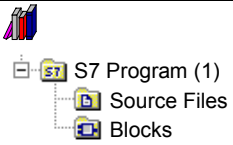
- **Объект Проект**
- Объект Станция
- Объект Программируемый модуль
- Объект Программа S7/M7
- Объект Папка с исходными файлами
- Объект Папка с блоками

Символ	Папка объекта	Выборка важных функций
	Проект	• Создание проекта • Архивирование проектов и библиотек • Печать проектной документации • Переупорядочивание • Преобразование и редактирование текстов пользователя • Вставка объектов станции оператора • Редактирование проектов более чем одним пользователем • Конвертирование проектов версии 1 • Конвертирование проектов версии 2 • Настройка интерфейса PG/PC

Символ	Объекты на уровне проекта	Выборка важных функций
	Станция: SIMATIC 300 station SIMATIC 400 station	Вставка станций Станции – это как объекты (уровень проекта), так и папки для объектов (уровень станции). Другие функции можно найти в разделе Объект Станция
	Программа S7 Программа M7	Вставка программы S7/M7 Программы S7/M7 – это как объекты (уровень проекта), так и папки для объектов (уровень станции). Другие функции можно найти в разделе Объект Программа S7/M7
	Сеть для запуска инструментов для конфигурирования сети и настройки ее свойств.	Свойства подсетей и коммуникационных узлов Обзор: связь с помощью глобальных данных Процедура для конфигурирования связи через глобальные данные

5.2.3 Объект Библиотека

Библиотека может содержать программы S7/M7 и используется для хранения блоков. Библиотека располагается в верхней части иерархии объектов

	• **Объект Библиотека** • Объект Программа S7/M7 • Объект Папка с исходными файлами • Объект Папка для блоков

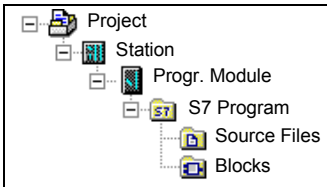
Символ	Папка объекта	Выборка важных функций
	Библиотека	• Обзор стандартных библиотек • Работа с библиотеками • Архивирование проектов и библиотек

Символ	Объекты на уровне библиотеки	Выборка важных функций
	Программа S7 Программа M7	Вставка программы S7/M7 Программы S7/M7 – это как объекты (уровень проекта), так и папки для объектов (уровень станции). Другие функции можно найти в разделе Объект Программа S7/M7

5.2.4 Объект Станция

Станция SIMATIC 300/400 представляет собой аппаратную конфигурацию S7 с одним или несколькими программируемыми модулями.

Положение в изображении проекта

	• Объект Проект • **Объект Станция** • Объект Программируемый модуль • Объект Программа S7/M7 • Объект Папка с исходными файлами • Объект Папка с блоками

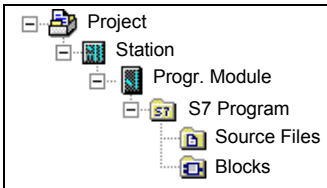
Символ	Папка объекта	Выборка важных функций
	Станция	• Вставка станции • Загрузка станции в устройство программирования • Загрузка конфигурации в программируемый контроллер • Считывание конфигурации из станции • Отображение сообщений CPU и диагностических сообщений, определенных пользователем • Диагностика аппаратуры и отображение информации о модуле • Отображение и изменение режима работы • Отображение и установка времени и даты • Стирание загрузочной/рабочей памяти и сброс CPU
	Станция SIMATIC PC	• Создание и назначение параметров станциям SIMATIC PC • Проектирование соединений для станции SIMATIC PC

Символ	Объекты на уровне станции	Выборка важных функций
	Аппаратура	• Базовая процедура для конфигурирования аппаратуры • Основные этапы конфигурирования станции • Обзор: Процедура для конфигурирования и назначения параметров центральной структуре • Базовая процедура для конфигурирования Master-системы DP • Конфигурирование мультипроцессорного режима
	Программируемый модуль	• Программируемые модули – это как объекты (уровень станции), так и папки для объектов (уровень "Программируемые модули"). Другие функции можно найти в разделе Объект Программируемый модуль

5.2.5 Объект Программируемый модуль

Этот объект представляет данные о назначении параметров программируемого модуля (CPUxxx, FMxxx, CPxxx). Системные данные модулей с не сохраняемой памятью (например, CP441) загружаются через CPU станции. Поэтому таким модулям не ставится в соответствие ни одного объекта "system data [системные данные]", и они не отображаются в иерархии проекта.

Положение в изображении проекта

	• Объект Проект • Объект Станция • **Объект Программируемый модуль** • Объект Программа S7/M7 • Объект Папка с исходными файлами • Объект Папка с блоками

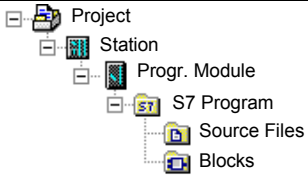
Символ	Папка объекта	Выборка важных функций
	Программируемый модуль	Обзор: Процедура для конфигурирования и назначения параметров центральной структуре Отображение сообщений CPU и диагностических сообщений, определенных пользователем Диагностика аппаратуры и отображение информации о модуле • Загрузка через платы памяти СППЗУ • Защита паролем для доступа к программируемым контроллерам • Отображение окна принудительно задаваемых величин • Отображение и изменение режима работы • Отображение и установка времени и даты • Установка эксплуатационных характеристик • Стирание загрузочной/рабочей памяти и сброс CPU • Диагностические символы в отображении Online • Деление областей памяти • Сохранение загруженных блоков на встроенном СППЗУ

Символ	Объекты на уровне "Программируемые модули"	Выборка важных функций
  	Программы: Программа S7 Программа M7 Программа	Вставка программы S7/M7 Программы S7/M7 – это как объекты (уровень проекта), так и папки для объектов (уровень программы). Другие функции можно найти в разделе Объект Программа S7/M7.
	Подключения для определения соединений внутри сети	Соединение в сеть станций внутри проекта Типы соединений и партнеры по связи Что вы должны знать о различных типах соединений Ввод нового соединения Проектирование соединений для модулей станции SIMATIC

5.2.6 Объект Программа S7/M7

Программа S7/M7 – это папка, содержащая программное обеспечение для модулей CPU S7/M7 или программное обеспечение для модулей, не являющихся CPU (например, программируемых модулей CP или FM).

Положение в изображении проекта

	• Объект Проект • Объект Станция • Объект Программируемый модуль • **Объект Программа S7/M7** • Объект Папка с исходными файлами • Объект Папка с блоками

Символ	Папка объекта	Выборка важных функций
	Программа S7	Базовая процедура для создания логических блоков Назначение номеров сообщений Создание диагностических сообщений, определяемых пользователем Преобразование и редактирование текстов пользователя Отображение сообщений CPU и диагностических сообщений, определяемых пользователем Программные мероприятия по обработке ошибок
	Программа M7	Процедура для систем M7
	Программа	Создание программного обеспечения в проекте (общее)

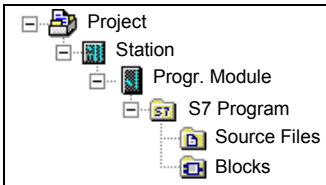
Символ	Объекты на уровне проекта	Выборка важных функций
	Таблица символов для присвоения символов сигналам и другим переменным	Абсолютная и символическая адресация Структура и компоненты таблицы символов Ввод совместно используемых символов Общие советы по вводу символов Назначение и редактирование сообщений, относящихся к символам Преобразование и редактирование текстов пользователя Конфигурирование атрибутов для управления и наблюдения со стороны оператора посредством таблицы символов Редактирование атрибутов коммуникаций Экспорт и импорт таблицы символов
	Исходный файл	Исходные файлы могут быть как объектами (уровень программы), так и папками для объектов (уровень исходных файлов). Другие функции можно найти в разделе Объект Папка с исходными файлами
	Папка с блоками	Другие функции можно найти в разделе Объект Папка с блоками

5.2.7 Объект Папка с блоками

Папка с блоками для представления offline может содержать: логические блоки (OB, FB, FC, SFB, SFC), блоки данных (DB), типы данных, определенные пользователем (UDT) и таблицы переменных (VAT). Объект Системные данные представляет блоки системных данных.

Папка с блоками для представления online содержит исполняемые части программы, которые были загружены в программируемый контроллер.

Положение в изображении проекта



- Объект Проект
- Объект Станция
- Объект Программируемый модуль
- Объект Программа S7/M7
- Объект Папка с исходными файлами
- **Объект Папка с блоками**

Символ	Папка объекта	Выборка важных функций
	Блоки	Загрузка с управлением проектом Загрузка без управления проектом Обзор доступных справочных данных Переадресация Сравнение блоков Преобразование и редактирование текстов пользователя Переходы к описаниям языков и к помощи по блокам и системным атрибутам

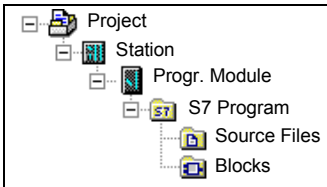
Символ	Объекты в папке с блоками	Выборка важных функций
	Блоки в целом	Базовая процедура для создания логических блоков Создание блоков Основная информация о программировании исходных файлов на AWL Сравнение блоков
	Организа-ционный блок (OB)	Дополнительные функции: Введение в типы данных и типы параметров Требования для загрузки Тестирование с помощью статуса программы Что нужно знать о тестировании в пошаговом режиме / контрольных точках Переадресация Помощь по блокам
	Функция (FC)	Дополнительные функции: Введение в типы данных и типы параметров Требования для загрузки Тестирование с помощью статуса программы Что нужно знать о тестировании в пошаговом режиме / контрольных точках Переадресация Атрибуты для блоков и параметров
	Функцио-нальный блок (FB)	Дополнительные функции: Введение в типы данных и типы параметров Использование мультиэкземпляров Требования для загрузки Тестирование с помощью статуса программы Что нужно знать о тестировании в пошаговом режиме / контрольных точках Переадресация Атрибуты для блоков и параметров Назначение и редактирование сообщений, относящихся к блокам Проектирование сообщений PCS7 Преобразование и редактирование текстов пользователя Назначение системных атрибутов параметрам

Символ	Объекты в папке с блоками	Выборка важных функций
		функционального блока
	Тип данных, определенный пользователем (UDT)	Создание блоков Базовая информация по программированию исходных файлов на AWL Введение в типы данных и типы параметров Использование типов данных, определенных пользователем, для доступа к данным Атрибуты для блоков и параметров
	Блок данных (DB)	Просмотр данных в блоках данных Просмотр описания блоков данных Требования для загрузки Использование мультиэкземпляров Программный статус блоков данных Введение в типы данных и типы параметров Атрибуты для блоков и параметров Назначение и редактирование сообщений, связанных с блоками (только для экземплярных блоков данных) Проектирование сообщений PCS7 (только для экземплярных блоков данных) Преобразование и редактирование текстов пользователя (только для экземплярных блоков данных)
	Системная функция (SFC)	Требования для загрузки Атрибуты для блоков и параметров Помощь по блокам
	Системный функциональный блок (SFB)	Требования для загрузки Атрибуты для блоков и параметров Проектирование сообщений PCS7 Преобразование и редактирование текстов пользователя Помощь по блокам
	Таблица переменных (VAT)	Базовая процедура при наблюдении и управлении с помощью таблицы переменных Введение в тестирование с помощью таблицы переменных Введение в наблюдение переменных Введение в управление переменными Введение в принудительное задание значений переменным
	Системный блок данных (SDB)	Системные блоки данных (SDB) редактируются только косвенно посредством функций: Введение в конфигурирование аппаратуры Свойства подсетей и коммуникационных узлов Обзор: Связь с помощью глобальных данных Назначение и редактирование сообщений, относящихся к символам • Требования для загрузки

5.2.8 Объект Папка с исходными файлами

Папка с исходными файлами содержит исходные программы в текстовом формате.

Положение в изображении проекта

	• Объект Проект • Объект Станция • Объект Программируемый модуль • Объект Программа S7/M7 • **Объект Папка с исходными файлами** • Объект Папка с блоками

Символ	Папка объекта	Выборка важных функций
	Исходный файл (например, файл на языке AWL)	Основная информация по программированию исходных файлов на AWL Создание исходных файлов на AWL Вставка шаблонов блоков в исходные файлы на AWL Вставка исходного кода из существующих блоков в исходные файлы на AWL Проверка непротиворечивости в исходных файлах на AWL Компилирование исходных файлов на AWL Генерирование исходных файлов на AWL из блоков Экспорт исходных файлов Импорт исходных файлов
	Шаблон сети	Создание шаблонов сетей

5.2.9 Программа S7/M7 без станции или CPU

Вы можете создавать программы, не имея заранее сконфигурированной станции SIMATIC. Это значит, что сначала вы можете работать независимо от модулей и их настроек, которые вы намерены программировать.

Создание программы S7/M7

1. Откройте соответствующий проект с помощью команды меню **File > Open [Файл > Открыть]** или активизируйте окно проекта.
2. Выберите проект в окне проекта представления offline.
3. Выберите одну из следующих команд меню в зависимости от того, для какого программируемого контроллера создается программа:
   - **Insert > Program > S7 Program [Вставка > Программа > Программа S7]**, если ваша программа должна работать на устройстве SIMATIC S7.
   - **Insert > Program > M7 Program [Вставка > Программа > Программа M7]**, если ваша программа должна работать на устройстве SIMATIC M7.

Программа S7/M7 добавляется и располагается в окне проекта непосредственно под проектом. Она содержит папку для блоков и пустую таблицу символов. Теперь вы можете создавать и программировать блоки.

Назначение программы программируемому модулю

Если вы вставляете программы, которые не зависят от конкретного модуля, вы можете легко назначить их модулю позднее путем копирования или перемещения этих программ на символ модуля с использованием функции буксировки (drag and drop).

Добавление программы к библиотеке

Если программа должна использоваться для программируемого контроллера SIMATIC S7, и вы хотите использовать ее многократно как "фонд программ", то вы можете вставить ее в библиотеку. Однако при тестировании программы должны находиться непосредственно под проектом, так как это единственный способ установления связи с программируемым контроллером.

Доступ к программируемому контроллеру

Выберите представление проекта в режиме online. Вы можете выполнить настройку адресов в диалоговом окне, содержащем свойства программы.

Примечание

При удалении станций или программируемых модулей вам будет задан вопрос, не хотите ли вы удалить также содержащуюся внутри программу. Если вы выберете отказ от удаления программы, то она будет присоединена непосредственно под проектом как программа без станции.

5.3 Пользовательский интерфейс и работа пользователя

5.3.1 Философия работы с пакетом

Цель: простая объектно-ориентированная обработка

Графический пользовательский интерфейс ориентирован на то, чтобы сделать манипулирование программным обеспечением интуитивно понятным. В программном обеспечении вы найдете объекты, знакомые вам из вашего повседневного рабочего окружения, например, станции, модули, программы, блоки.

Действия, которые вы выполняете при работе со STEP 7, включают в себя создание, выбор и манипулирование с объектами этого типа.

Отличия от проблемно-ориентированной обработки

В случае проблемно-ориентированной обработки вы должны были решить, для какой задачи какое приложение требовалось, а затем запустить это приложение.

Принцип, используемый при объектно-ориентированной обработке, состоит в том, чтобы решить, какой объект должен обрабатываться, а затем открыть этот объект, чтобы его редактировать.

При объектно-ориентированной обработке не нужны специальные знания о синтаксисе команд. Объекты представляются в пользовательском интерфейсе графическими символами, или пиктограммами, которые вы открываете с помощью команд меню или щелчком мыши.

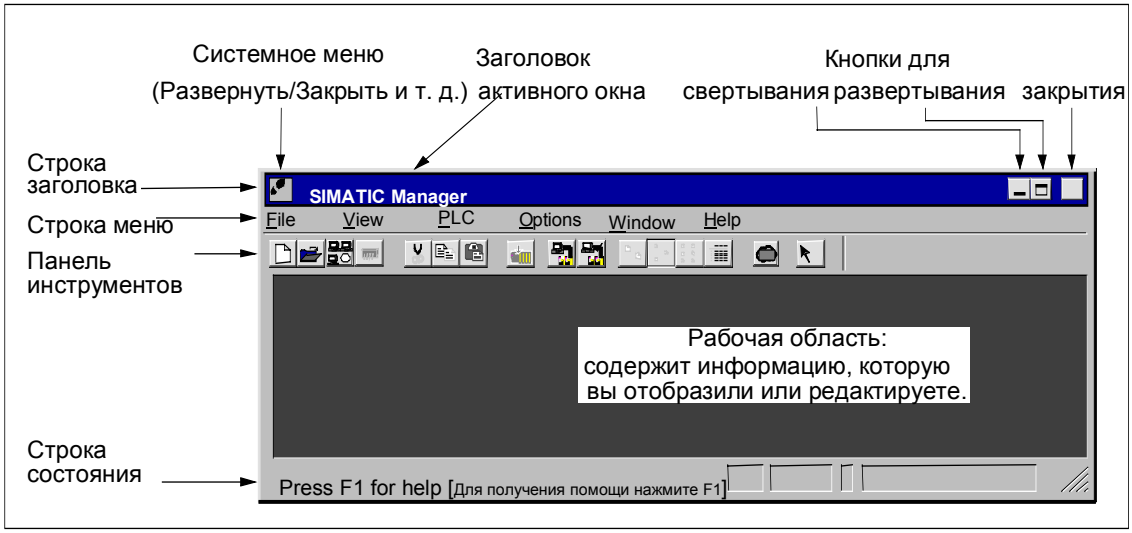
Когда вы открываете объект, автоматически запускается соответствующее программное приложение для отображения или редактирования содержимого этого объекта.

Продолжая читать…

На следующих страницах описаны некоторые основные действия, используемые при редактировании объектов. Теперь не спеша изучите эти основные этапы обработки, так как далее в этом руководстве они подробно описываться не будут.

5.3.2 Компоновка окна

На следующем рисунке показаны стандартные компоненты окна:



Пояснения к рисунку: File - файл; View - вид; PLC - ПЛК; Options – параметры; Window - окно; Help - помощь

Строка заголовка и строка меню

Строка заголовка и строка меню всегда находятся в верхней части окна. Строка заголовка содержит заголовок окна и пиктограммы для управления окном. Строка меню содержит все меню, доступные в этом окне.

Панель инструментов

Панель инструментов содержит пиктограммы (или инструментальные кнопки), которые обеспечивают быстрый доступ к часто используемым и доступным в данный момент командам строки меню с помощью однократного щелчка мышью. Краткое описание функции соответствующей кнопки отображается вместе с дополнительной информацией в строке состояния, когда вы кратковременно помещаете курсор на кнопке.

Если доступ к кнопке в текущей конфигурации невозможен, то эта кнопка становится серой.

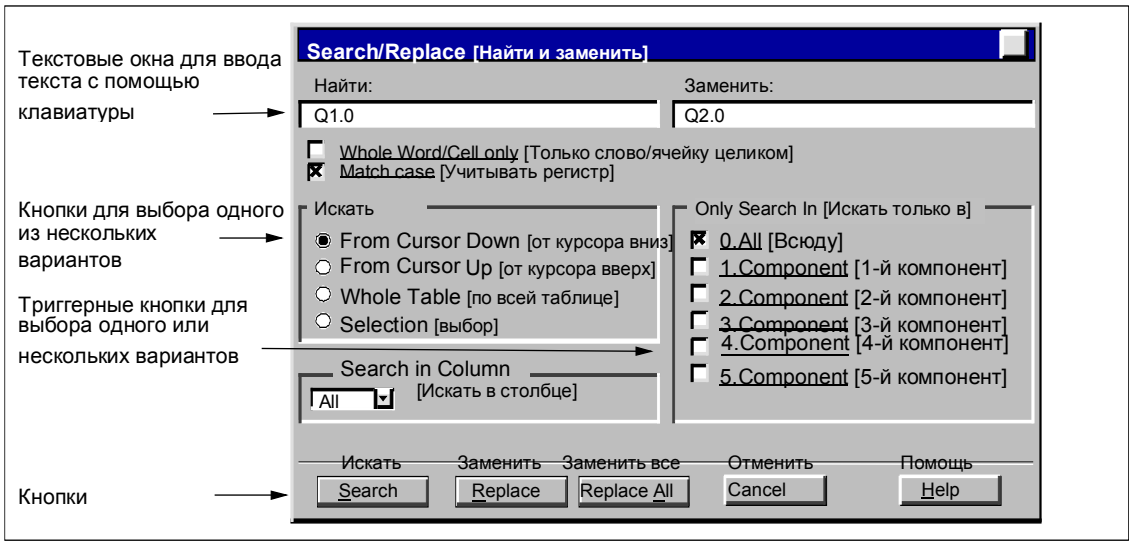
Строка состояния

Строка состояния отображает контекстно-зависимую информацию.

5.3.3 Элементы в диалоговых окнах

Ввод данных в диалоговых окнах

В диалоговых окнах вы можете вводить информацию, необходимую для исполнения конкретной задачи. Компоненты, наиболее часто встречающиеся в диалоговых окнах, объяснены на примере, представленном на следующем рисунке.

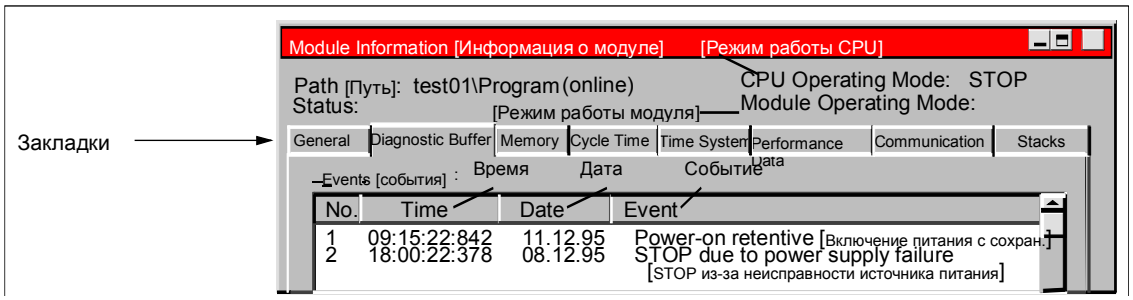


Окна списка и комбинированные окна

Рядом с текстовыми окнами иногда находится стрелка, указывающая вниз. Эта стрелка показывает, что имеются еще варианты, доступные для выбора из этого окна. Щелкните на стрелке, чтобы отрыть окно списка или комбинированное окно. Если щелкнуть на записи в списке, то она автоматически отображается в текстовом окне.

Закладки в диалоговых окнах

Содержимое некоторых диалоговых окон организовано с использованием закладок для улучшения ясности информации путем деления диалогового окна на страницы с закладками (см. рисунок внизу).



Пояснения к рисунку: General – общие данные; Diagnostic Buffer – диагностический буфер; Memory – память; Cycle Time – время цикла; Time System – система времени; Performance Data – данные о производительности; Communication – связь; Stacks – стеки.

Названия страниц с закладками показаны на закладках вдоль верхнего края диалогового окна. Для выноса конкретной страницы на передний план нужно просто щелкнуть на соответствующей закладке.

5.3.4 Создание объектов и управление ими

Некоторые основные этапы обработки одинаковы для всех объектов и не зависят от их типа. Здесь дается обзор этих стандартных последовательностей манипулирования. Знание этих стандартных процедур требуется к другим разделам данного руководства.

Обычная последовательность этапов при манипулировании объектами:

- создать объект
- выбрать объект
- выполнить действия над объектом (например, скопировать, удалить).

Установка пути для создания новых проектов/библиотек

Перед тем как впервые создавать новые проекты или библиотеки, вы должны установить путь к тому месту, где вы хотите создавать эти объекты, выбрав команду меню **Options > Customize [Параметры > Настроить]**. В закладке ”General [Общие свойства]" открывшегося диалогового окна вы можете указать имя маршрута, под которым вы хотите сохранять новые проекты или библиотеки.

Создание объектов

Мастер нового проекта STEP 7 предлагает поддержку при создании нового проекта и вставке объектов. Используйте команду меню **File > "New Project" Wizard [Файл > Мастер нового проекта]** для открытия мастера. В появившихся диалоговых окнах вы можете установить структуру своего проекта, а затем предоставить мастеру возможность создать проект для вас.

Если вы не хотите использовать мастер, то вы можете создавать проекты и библиотеки с помощью команды меню **File > New [Файл > Новый]**. Эти объекты образуют начальную точку иерархии объектов. Вы можете создавать все остальные объекты в этой иерархии с помощью команд в меню Insert [Вставка], если они не создаются автоматически. Исключением являются модули в станции SIMATIC, которые создаются при конфигурировании аппаратуры или с помощью мастера нового проекта.

Открытие объектов

Имеется несколько способов открытия объекта в подробном представлении:

- дважды щелкнуть на пиктограмме объекта
- выбрать объект, а затем команду меню **Edit > Open Object [Редактировать > Открыть объект]**. Это работает только для объектов, не являющихся папками.

Открыв объект, вы можете создавать или изменять его содержимое.

Если вы открываете объект, который не содержит других объектов, то его содержимое отображается с помощью надлежащего программного компонента в новом окне для целей редактирования. Вы не можете изменять объекты, содержимое которых уже используется где-либо еще.

Замечание

Исключение: Станции появляются как папки для программируемых модулей (когда вы дважды щелкаете на них) и для конфигурирования станции. Если вы дважды щелкнете на объекте "Hardware [Аппаратура]", то запускается приложение для конфигурирования аппаратуры. Выбор станции и выбор команды меню **Edit > Open Object [Редактировать > Открыть объект]** оказывает одинаковое действие.

Формирование иерархии объектов

Для создания иерархии объектов используйте мастер нового проекта. Когда вы открывает папку, то содержащиеся в ней объекты отображаются на экране. Теперь, используя меню Insert [Вставка], вы можете создавать в проекте другие объекты, например, дополнительные станции. В меню Insert [Вставка] активны команды только для тех объектов, которые могут быть вставлены в текущую папку.

Установка свойств объекта

Свойства объекта – это данные, принадлежащие объекту, которые определяют его поведение. Диалоговое окно для установки параметров объекта автоматически появляется, когда вы создаете новый объект, и его свойства должны быть установлены. Позднее эти свойства могут быть изменены.

С помощью команды меню **Edit > Object Properties [Редактировать > Свойства объекта]** открывается диалоговое окно, в котором вы можете отобразить или установить свойства для выбранного объекта.

С помощью команды меню **Edit > Special Object Properties [Редактировать > Специальные свойства объекта]** вы можете открывать диалоговые окна и вводить данные, требуемые для функций управления и наблюдения со стороны оператора и для проектирования сообщений.

Например, чтобы отобразить специальные свойства блока для управления и наблюдения со стороны оператора, этот блок должен быть помечен как подходящий для этих целей, то есть системный атрибут ”s7_m_c" должен быть установлен на значение ”true [истина]" в закладке ”Attributes [Атрибуты]" свойств блока.

Замечание

- Свойства папки "System Data [Системные данные]" и объекта "Hardware [Аппаратура]" не могут быть отображены или изменены.
- Вы не можете делать записи в диалоговых окнах для свойств объекта в проекте, защищенном от записи. В этом случае окна ввода имеют серый цвет.
- Если вы отображаете свойства программируемых модулей, то вы не можете редактировать отображенные параметры из соображений непротиворечивости. Чтобы редактировать эти параметры, вы должны открыть приложение "Configuring Hardware [Конфигурирование аппаратуры]".

Вырезание, вставка, копирование

Большинство объектов может быть вырезано, вставлено или скопировано, как это обычно делается в Windows. Команды для этих функций находятся в меню Edit [Редактировать].

Вы можете также копировать объекты, используя буксировку. Если вы попытаетесь переместить или скопировать в недопустимое место, то курсор в качестве предупреждения отобразит знак запрета.

Когда вы копируете объект, то копируется и вся содержащаяся в нем иерархия. Это дает возможность снова и снова использовать компоненты, которые вы создаете при решении задачи автоматизации.

Замечание

Таблица соединений в папке "Connections [Соединения]" не может быть скопирована. Учтите, что при копировании списков текстов, относящихся к оператору, воспринимаются только языки, установленные в объекте назначения.

Пошаговое руководство по копированию вы найдете под Copying Objects [Копирование объектов].

Переименование объектов

SIMATIC Manager назначает новым объектам стандартные имена. Эти имена обычно формируются из типа объекта (если несколько объектов этого типа могут быть созданы в одной и той же папке) и номера.

Например, первая программа S7 будет названа "S7 Program(1)", вторая – "S7 Program(2)" и т. д. Таблица символов называется просто "Symbols", так как в каждой папке она может быть только одна.

Вы можете изменять имена большинства объектов (и проектов) и назначать им имена, которые более соответствуют их содержанию.

Для проектов имена каталогов в пути должны содержать не более 8 символов. Иначе возможны проблемы при архивировании и использовании языка C для M7 (компилятор Borland).

Вы можете изменить имя объекта непосредственно или с помощью свойств объекта.

- Непосредственно:

  Если вы медленно дважды щелкнете на имени выбранного объекта, то вокруг текста появляется рамка. Затем вы можете редактировать имя с помощью клавиатуры.

- Использование свойств объекта:

  Выберите нужный объект и команду меню **Edit > Object Properties [Редактирование > Свойства объекта]**. Измените имя в диалоговом окне. Когда вы закрываете диалоговое окно свойств, объект переименовывается и отображается под новым именем.

Если изменение имени объекта не разрешено, то поле ввода в диалоговом окне показывается серым цветом, текущее имя отображается, а ввод текста невозможен.

Замечание

Если при редактировании имени вы перемещаете указатель мыши за пределы поля с именем и выполняете другое действие (например, выбираете команду меню), то процедура редактирования заканчивается. Измененное имя принимается и вводится, если это разрешено.

Пошаговое руководство по переименованию вы найдете под Renaming Objects [Переименование объектов].

Перемещение объектов

С помощью вы SIMATIC Manager можете перемещать объекты из одной папки в другую, даже если место назначения находится в другом проекте. Если вы перемещаете папку, то все ее содержимое тоже перемещается.

Замечание

Нельзя перемещать следующие объекты:

Соединения

Системные блоки данных (SDB) в представлении online

Системные функции (SFC) и системные функциональные блоки (SFB) в представлении online

---

Пошаговое руководство по перемещению вы найдете под Moving Objects [Перемещение объектов].

Сортировка объектов

Вы можете сортировать объекты в подробном отображении (команда меню **View > Details [Вид > Подробности]**) в соответствии с их атрибутами. Для этого щелкните на соответствующем заголовке нужного атрибута. Если вы щелкните еще раз, порядок сортировки изменится на противоположный. Блоки одного типа сортируются в соответствии с их порядковыми номерами, например, FB1, FB2, FB11, FB12, FB21, FC1.

Порядок сортировки по умолчанию

Когда вы повторно открываете проект, объекты в подробном представлении отображаются в соответствии с порядком сортировки по умолчанию. Примеры:

- Блоки показываются в порядке "OB, SDB, FB, FC, DB, UDT, VAT, SFB, SFC"
- В проекте сначала показываются станции, а затем программы S7.

Таким образом, умолчание в подробном представлении не является алфавитно-цифровым порядком по возрастанию или убыванию.

Восстановление порядка сортировки по умолчанию

После пересортировки, например, щелкнув на заголовке столбца "Object Name [Имя объекта]", вы можете восстановить порядок по умолчанию, действуя следующим образом:

- Щелкните в подробном представлении на заголовке столбца "Type [Тип]".
- Закройте проект и откройте его снова.

Удаление объектов

Вы можете удалять папки и объекты. Если вы удаляете папку, то все содержащиеся в ней объекты тоже удаляются.

Вы не можете отменить процедуру удаления. Если вы не уверены, что вам действительно не нужен некоторый объект, то лучше сначала заархивировать весь проект.

Замечание

Вы не можете удалять следующие объекты:

Соединения

Системные блоки данных (SDB) в представлении online

Системные функции (SFC) и системные функциональные блоки (SFB) в представлении online

---

Пошаговое руководство по удалению вы найдете под Deleting Objects [Удаление объектов].

5.3.5 Выбор объектов в браузере

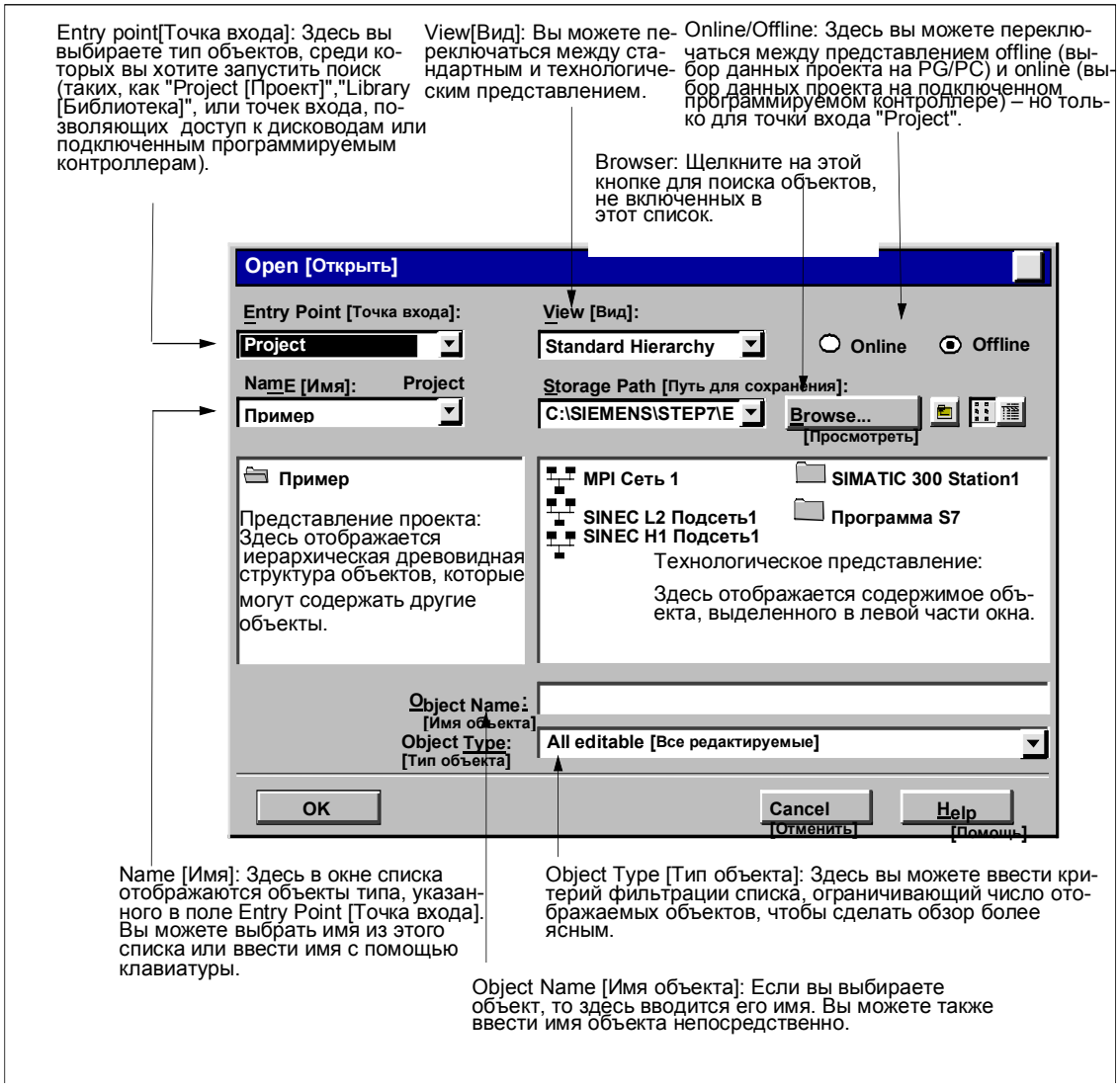
Выбор объектов в диалоговом окне (браузере – системе навигации и просмотра) – это действие, регулярно необходимое вам для большого количества различных этапов редактирования.

Вызов браузера

Диалоговое окно браузера вызывается в приложении для конфигурирования аппаратуры, например, с помощью такой команды меню, как **Station > New/Open [Станция > Новая/Открыть]** (одно исключение – окно базового приложения "SIMATIC Manager").

Структура диалогового окна браузера

В браузере у вас имеются следующие варианты выбора, показанные на следующем рисунке.



5.3.6 Память сеанса работы

SIMATIC Manager может сохранить содержимое окон (то есть открытых проектов и библиотек) и компоновку этих окон.

- С помощью команды меню **Options > Customize [Параметры > Настройка]** вы определяете, должны ли быть сохранены в конце сеанса работы содержимое окон и их компоновка. В начале следующего сеанса это содержимое и компоновка восстанавливаются. В открытых проектах курсор располагается на последней выбранной папке.
- С помощью команды меню **Window > Save Settings [Окно > Сохранить настройки]** вы сохраняете содержимое текущего окна и расположение окон.
- С помощью команды меню **Window > Restore Settings [Окно > Восстановить настройки]** вы восстанавливаете содержимое и компоновку окон, которые вы сохранили с помощью команды меню **Window > Save Settings [Окно > Сохранить настройки]**. В открытых проектах курсор располагается на последней выбранной папке.

Замечание

Содержимое окон проектов online, содержимое окна "Accessible Nodes [Доступные узлы]" и содержимое окна "S7 Memory Card [Плата памяти S7]" не сохраняется. Никакие пароли, которые вы, возможно, вводили для доступа к программируемым контроллерам (S7-300/S7-400), в конце сеанса работы не сохраняются.

5.3.7 Изменение расположения окон

Чтобы расположить все отображаемые окна каскадом друг за другом, выберите одну из следующих возможностей:

- Выберите команду меню **Window > Arrange > Cascade [Окно > Упорядочить > Каскад]**.
- Нажмите комбинацию клавиш SHIFT + F5.

Чтобы расположить все отображаемые окна на экране сверху вниз, выберите команду меню **Window > Arrange > Horizontally [Окно > Упорядочить > Горизонтально]**.

Чтобы расположить все отображаемые окна на экране слева направо, выберите команду меню **Window > Arrange > Vertically [Окно > Упорядочить > Вертикально]**.

5.3.8 Сохранение и восстановление расположения окон

Приложения STEP 7 обладают свойством, позволяющим вам сохранять текущее расположение окон и восстанавливать его на последующем этапе. Вы можете выполнить эту настройку с помощью команды меню **Options > Customize [Параметры > Настройка]**, закладка "General [Общие свойства]".

Что сохраняется?

Когда вы сохраняете компоновку окон, то записывается следующая информация:

- положение главного окна
- открытые проекты и библиотеки и их расположение относительно окна
- порядок всех расположенных каскадом окон

Замечание

Содержимое окон проектов online, содержимое окна "Accessible Nodes [Доступные узлы]" и содержимое окна "S7 Memory Card [Плата памяти S7]" не сохраняется.

Сохранение компоновки окон

Для сохранения текущего расположения окон выберите команду меню **Window > Save Settings [Окно > Сохранить настройки]**.

Восстановление компоновки окон

Для восстановления сохраненного расположения окон выберите команду меню **Window > Restore Settings[Окно > Восстановить настройки]**.

Замечание

При восстановлении окна подробно отображается только та часть иерархии, которая содержит объект, выбранный при сохранении расположения окон.

5.4 Управление с клавиатуры

5.4.1 Управление с клавиатуры

Международные названия клавиш	Немецкие названия клавиш
HOME	POS1
END	ENDE
PAGE UP	BILD AUF
PAGE DOWN	BILD AB
CTRL	STRG
ENTER	Eingabetaste [Клавиша ввода]
DEL	ENTF
INSERT	EINFG

5.4.2 Комбинации клавиш для команд меню

Любая команда меню может быть выбрана нажатием комбинации клавиши с клавишей ALT.

Нажимайте следующие клавиши в указанном порядке:

- Клавиша ALT.
- Буква, подчеркнутая в имени меню, которое вам необходимо (например, ALT, F для меню "File [Файл]" – если меню "File" включено в строку меню). Меню открывается.
- Буква, подчеркнутая в команде меню, которая вам необходима (например, N для команды меню "New [Новый]"). Если эта команда меню имеет подменю, то это подменю тоже открывается. Действуйте, как описано выше, пока вы не выберете всю команду меню, нажимая соответствующие буквы.

Как только вы введете последнюю букву в комбинации клавиш, команда меню будет выполнена.

Примеры:

Команда меню	**Комбинация клавиш**
File > Archive [Файл > Архив]	ALT, F, A
Window > Arrange > Cascade [Окно > Упорядочить > Каскад]	ALT, W, A, C

Клавиши быстрого вызова для команд меню

Команда	Клавиши быстрого вызова
New [Новый] (Меню File [Файл])	CTRL+N
Open [Открыть] (Меню File [Файл])	CTRL+O
Close [Закрыть] (Меню File [Файл])	
Compile [Компилировать] (Меню File [Файл])	CTRL+B
Print [Печатать] (Object [Объект]) (Меню File [Файл])	CTRL+P
Exit [Выход] (Меню File [Файл])	ALT+F4
Copy [Копировать] (Меню Edit [Редактировать])	CTRL+C
Cut [Вырезать] (Меню Edit [Редактировать])	CTRL+X
Paste [Вставить] (Меню Edit [Редактировать])	CTRL+V
Delete [Удалить] (Меню Edit [Редактировать])	DEL
Select All [Выделить все] (Меню Edit [Редактировать])	CTRL+A
Object Properties [Свойства объекта] (Меню Edit [Редактировать])	ALT+RETURN
Open Object [Открыть объект] (Меню Edit [Редактировать])	CTRL+ALT+O
Download [Загрузить] (Меню PLC [ПЛК])	CTRL+L
Operating Mode [Режим работы] (Меню PLC [ПЛК])	CTRL+I
Update [Обновить] (Меню View [Вид])	F5
Обновляет отображение статуса видимых CPU в представлении online	CTRL+F5
Customize [Настроить] (Меню Options [Параметры])	CTRL+ALT+E
Reference Data, Display [Справочные данные, Отобразить] (Меню Options [Параметры])	CTRL+ALT+R
Arrange, Cascade [Упорядочить, Каскад] (Меню Window [Окно])	SHIFT+F5
Arrange, Horizontally [Упорядочить, Горизонтально] (Меню Window [Окно])	SHIFT+F2
Arrange, Vertically [Упорядочить, Вертикально] (Меню Window [Окно])	SHIFT+F3
Context-Sensitive Help [Контекстно-чувствительная помощь] (Меню Help [Помощь])	F1 (Если имеется текущий контекст, например, выбранная команда меню, то открывается соответствующая тема помощи. В противном случае отображается страница с содержанием)

5.4.3 Комбинации клавиш для перемещения курсора

Перемещение курсора в строке меню/всплывающих меню

Направление	Нажать
Переместить в строку меню	F10
Переместить во всплывающее меню	SHIFT+F10
Переместить в меню, содержащее подчеркнутую букву или цифру, которую вы нажимаете на клавиатуре	ALT+подчеркнутый символ в заголовке меню
Выбрать команду меню, в которой подчеркнутая буква или цифра соответствует нажатой вами букве	Подчеркнутый символ в команде меню
Переместить на одну команду меню влево	СТРЕЛКА ВЛЕВО
Переместить на одну команду меню вправо	СТРЕЛКА ВПРАВО
Переместить на одну команду меню вверх	СТРЕЛКА ВВЕРХ
Переместить на одну команду меню вниз	СТРЕЛКА ВНИЗ
Активизировать выбранную команду меню	ENTER
Отменить выбор имени меню или закрыть открытое меню и вернуться в текст	ESC

Перемещение курсора при редактировании текста

Для перемещения	Нажмите
на одну строку вверх или на один символ влево в тексте, состоящем только из одной строки	СТРЕЛКА ВВЕРХ
на одну строку вниз или на один символ вправо в тексте, состоящем только из одной строки	СТРЕЛКА ВНИЗ
на один символ вправо	СТРЕЛКА ВПРАВО
на один символ влево	СТРЕЛКА ВЛЕВО
на одно слово вправо	CTRL+ СТРЕЛКА ВПРАВО
на одно слово влево	CTRL+ СТРЕЛКА ВЛЕВО
к началу строки	HOME
к концу строки	END
к предыдущему экрану	PAGE UP
к следующему экрану	PAGE DOWN
к началу текста	CTRL+HOME
к концу текста	CTRL+END

Перемещение курсора в диалоговых окнах

Чтобы	Нажмите
перейти из одного поля ввода в следующее (слева направо и сверху вниз)	TAB
перейти на одно поле ввода в обратном направлении	SHIFT+TAB
перейти в поле ввода или опцию, содержащую подчеркнутую букву или цифру, которую вы вводите	ALT+подчеркнутый символ в заголовке меню
произвести выбор в списке параметров	клавишу со стрелкой
открыть список параметров	ALT+СТРЕЛКА ВНИЗ
выбрать или отменить выбор объекта в списке	клавишу пробела
подтвердить ввод и закрыть диалоговое окно (кнопка "OK")	ENTER
закрыть диалоговое окно без сохранения изменений (кнопка "Cancel [Отменить]")	ESC

5.4.4 Комбинации клавиш для выделения текста

Чтобы выделить или отменить выделение	Press
одного символа за один раз справа	SHIFT+СТРЕЛКА ВПРАВО
одного символа влево	SHIFT+ СТРЕЛКА ВЛЕВО
до начала строки комментариев	SHIFT+HOME
до конца строки комментариев	SHIFT+END
одной строки текста вверх	SHIFT+ СТРЕЛКА ВВЕРХ
одной строки текста вниз	SHIFT+ СТРЕЛКА ВНИЗ
до предыдущего экрана	SHIFT+PAGE UP
до следующего экрана	SHIFT+PAGE DOWN
текста до начала файла	CTRL+SHIFT+HOME
текста до конца файла	CTRL+SHIFT+END

5.4.5 Комбинации клавиш для обращения к оперативной помощи

Чтобы	Нажмите
открыть помощь	F1 (Если имеется текущий контекст, например, выбранная команда меню, то открывается соответствующая тема помощи. В противном случае отображается страница с содержанием)
активизировать символ вопросительного знака для контекстно-чувствительной помощи	SHIFT+F1
закрыть окно помощи и вернуться в приложение	ALT+F4

5.4.6 Комбинации клавиш для переключения между окнами

Чтобы	Нажмите
переходить между подокнами окна	F6
вернуться в предыдущее подокно при отсутствии фиксируемого окна	Shift+F6
переходить в документе между окном документа и фиксируемым окном (например, окном описания переменных). Если фиксируемых окон нет, то вы можете использовать эту комбинацию клавиш для возврата в предыдущее подокно.	Shift+F6
переходить между окнами документа	Ctrl+F6
вернуться в предыдущее окно документа	Shift+Ctrl+F6
переходить между окнами, не содержащими документов (среда разработки приложений и фиксируемые окна в этой среде; при возврате в среду разработки приложений эта комбинация клавиш активизирует окно документа, которое было активно последним)	Alt+F6
вернуться в предыдущее окно, не содержащее документа	Shift+Alt+F6
закрыть текущее окно	Ctrl+F4

6 Создание и редактирование проекта

6.1 Структура проекта

Проекты используются для хранения данных и программ, которые создаются, когда вы собираете вместе решение задачи автоматизации. Данные, собранные в проекте, включают в себя:

- конфигурационные данные о структуре аппаратного обеспечения и параметрах модулей,
- конфигурационные данные о коммуникациях в сетях и
- программы для программируемых модулей.

Главной задачей при создании проекта является подготовка этих данных для программирования.

Данные хранятся в проекте в виде объектов. Объекты в проекте упорядочены в виде древовидной структуры (иерархии проекта). Отображение этой иерархии в окне проекта похоже на отображение Проводника в Windows 95. Другой внешний вид имеют только пиктограммы объектов.

Верхняя часть иерархии проекта структурирована следующим образом:

1-й уровень: проект

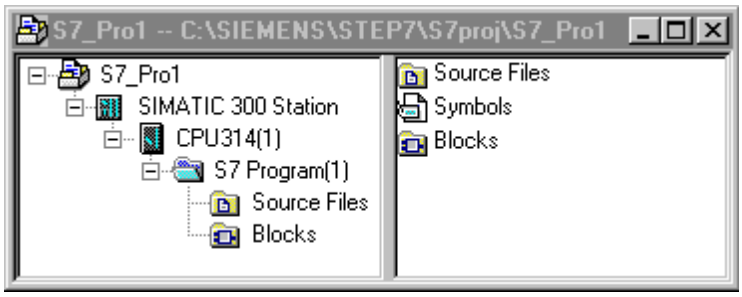
2-й уровень: подсети, станции или программы S7/M7

3-й уровень: зависит от объекта на уровне 2

Окно проекта

Окно проекта разделено на две половины. В левой половине показана древовидная структура проекта. Правая половина показывает объекты, содержащиеся в объекте, открытом в левой половине, в выбранном представлении (крупные символы, мелкие символы, список или подробности).

Щелкните в левой половине окна на квадратике со знаком плюс, чтобы отобразить полную древовидную структуру проекта. Полученная в результате структура будет выглядеть похоже на структуру, представленную на следующем рисунке.



В верхней части иерархии объектов находится объект ”S7_Pro1" как пиктограмма для всего проекта. Он может быть использован для отображения свойств проекта и служит в качестве папки для сетей (для конфигурирования сетей), станций (для конфигурирования аппаратуры) и для программ S7 и M7 (для создания программного обеспечения). Объекты, находящиеся в проекте, отображаются в правой половине окна проекта, когда вы выбираете пиктограмму проекта. Объекты в верхней части этого типа иерархии объектов (как библиотек, так и проектов) образуют начальную точку в диалоговых окнах, используемых для выбора объектов.

Отображение проекта

Вы можете отобразить структуру проекта в окнах проекта для данных, доступных на устройстве программирования, в представлении компонентов "offline", а для данных, доступных в системе программируемых контроллеров, в представлении компонентов "online".

Дополнительное представление, которое вы можете установить, доступно, если установлен соответствующий дополнительный пакет: технологическое представление.

Замечание

Конфигурирование аппаратуры и сетей может быть выполнено только в представлении "offline".

6.2 Создание проекта

6.2.1 Создание проекта

Для конструирования решения своей задачи автоматизации с использованием среды разработки управления проектом вам будет нужно создать новый проект. Новый проект создается в каталоге, который вы установили для проектов в закладке "General [Общие свойства]", когда вы выбрали команду меню **Options > Customize [Параметры > Настройка]**.

Замечание

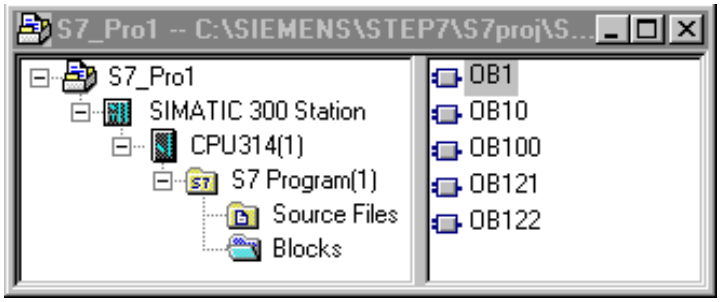
SIMATIC Manager разрешает имена длиной более восьми символов. Однако имя каталога проекта урезается до восьми символов. Поэтому имена проектов должны различаться в своих первых восьми символах. Регистр для имен не имеет значения.

Пошаговое руководство по созданию проекта вы найдете в разделе Создание проекта вручную или Создание проекта с помощью мастера.

Создание проекта с помощью мастера

Самый легкий путь создания нового проекта – это использование мастера нового проекта. Для открытия мастера используйте команду меню **File > "New Project" Wizard [Файл > Мастер нового проекта]**. Мастер подсказывает вам о необходимости ввести требуемые детали в диалоговых окнах, а затем создает для вас проект. Кроме станции, CPU, папки с программами, папки с исходными файлами, папки с блоками и OB1 вы можете выбрать существующие OB для обработки ошибок и аварийных сигналов.

На следующем рисунке показан пример проекта, созданного с помощью мастера.



Создание нового проекта вручную

Вы можете создать новый проект, используя также команду меню **File > New [Файл > Новый]** в SIMATIC Manager. Он уже содержит объект "MPI Subnet [Подсеть MPI]".

Альтернативные процедуры

При редактировании проекта вы свободны относительно порядка, в котором вы выполняете большинство задач. Создав проект, вы можете выбрать один из следующих методов:

- сначала сконфигурируйте аппаратуру, а затем создайте для нее программное обеспечение или
- начните с создания программного обеспечения, независимого от какой-либо аппаратуры.

Альтернатива 1: сначала сконфигурируйте аппаратуру

Если вы хотите сначала сконфигурировать аппаратуру, действуйте, как описано в томе 2 руководства "Конфигурирование аппаратуры с помощью STEP 7". Когда вы это сделаете, папки "S7 Program [Программа S7]" и "M7 Program [Программа M7]", необходимые для создания программного обеспечения, уже будут вставлены. Затем продолжайте, вставляя объекты, необходимые для создания программ. Затем создайте программное обеспечение для программируемых модулей.

Альтернатива 2: сначала создайте программное обеспечение

Вы можете также создавать программное обеспечение, не конфигурируя предварительно аппаратуру; это можно будет сделать позднее. Структура аппаратного обеспечения станции не должна быть обязательно установлена для ввода ваших программ.

Основная последовательность действий такова:

1. Вставьте в свой проект необходимые папки для программного обеспечения (S7/M7 Programs [Программы S7/M7]). Здесь вы просто решаете, должна ли папка с программами содержать аппаратуру S7 или аппаратуру M7.
2. Затем создайте программное обеспечение для программируемых модулей.
3. Сконфигурируйте свое аппаратное обеспечение.
4. Как только вы сконфигурировали аппаратуру, вы можете привязать программа M7 или S7 к CPU.

6.2.2 Вставка станций

В проекте станция представляет аппаратную структуру программируемого контроллера и содержит данные для конфигурирования и назначения параметров отдельным модулям.

Новые проекты, созданные с помощью мастера нового проекта уже содержат станцию. В противном случае вы можете создать станцию с помощью команды меню **Insert > Station [Вставить > Станция]**.

Вы можете выбирать среди следующих станций:

- станция SIMATIC 300
- станция SIMATIC 400
- станция SIMATIC H
- станция SIMATIC PC
- PC/устройство программирования
- SIMATIC S5
- другие станции, т. е. не SIMATIC S7/M7 и SIMATIC S5

Станция вставляется с предустановленным именем (например, SIMATIC 300 Station(1), SIMATIC 300 Station(2), и т. д.). Если вы хотите, то можете заменить это имя станции любым подходящим именем.

Пошаговое руководство по вставке станции вы найдете под Inserting a Station [Вставка станции].

Конфигурирование аппаратуры

При конфигурировании аппаратуры вы определяете CPU и все модули в своем программируемом контроллере с помощью каталога модулей. Приложение для конфигурирования аппаратуры запускается двойным щелчком на станции.

Для каждого программируемого модуля, который вы создаете в своей конфигурации, автоматически создаются программа S7 или M7 и таблица соединений (объект ”Connections [Соединения]"), как только вы сохранили конфигурацию аппаратуры и вышли из приложения. Проекты, созданные с помощью мастера нового проекта, уже содержат эти объекты.

Пошаговое руководство по конфигурированию аппаратуры вы найдете под Configuring the Hardware [Конфигурирование аппаратуры], а подробную информацию – под Basic Steps for Configuring a Station [Основные этапы конфигурирования станции].

Создание таблицы соединений

Пустая таблица соединений (объект ”Connections [Соединения]") создается автоматически для каждого программируемого модуля. Таблица соединений используется для определения коммуникационных связей между программируемыми модулями в сети. Когда она открывается, на экране появляется окно, содержащее таблицу, в которой вы определяете соединения между программируемыми модулями.

Подробную информацию вы найдете в разделе Соединение в сеть станций внутри проекта.

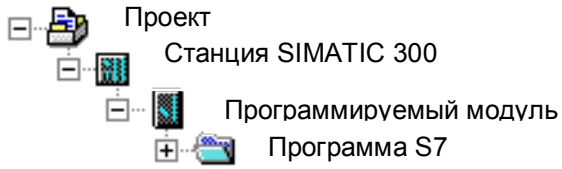
Следующие шаги

Создав конфигурацию аппаратуры, вы можете создавать программное обеспечение для своих программируемых модулей (см. также Вставка программы S7/M7).

6.2.3 Вставка программы S7/M7

Программное обеспечение для программируемых модулей хранится в папках объектов. Для программируемых модулей SIMATIC S7 такая папка объектов называется "S7 Program [Программа S7]", а для модулей SIMATIC M7 она называется "M7 Program [Программа M7]".

На следующем рисунке показан пример Программы S7 в программируемом модуле в станции SIMATIC 300.



Существующие компоненты

Программа S7/M7 создается автоматически для каждого программируемого модуля как контейнер для программного обеспечения.

Во вновь созданной папке S7 program [Программа S7] уже существуют следующие объекты:

- таблица символов (объект ”Symbols [Символы]")
- папка "Blocks [Блоки]" для хранения первого блока
- папка "Source Files [Исходные файлы]" для исходных файлов

Во вновь созданной папке M7 program [Программа M7] уже существуют следующие объекты:

- таблица символов (объект ”Symbols [Символы]")
- папка "Blocks [Блоки]"

Создание блоков S7

Вы хотите создавать программы в виде списка операторов (AWL), контактного плана (KOP) или функционального плана (FUP). Чтобы сделать это, выделите существующий объект "Blocks [Блоки]" и выберите команду меню **Insert > S7 Block [Вставка > Блок S7]**. В подменю вы можете выбрать тип блока, который вы хотите создать (например, блок данных, тип данных, определенный пользователем (UDT), функцию, функциональный блок, организационный блок или таблицу переменных (VAT)).

Теперь вы можете открыть пустой блок и начать ввод программы в виде списка операторов, контактного плана или функционального плана. Более подробную информацию об этом вы найдете в разделе Основная последовательность действий при создании логических блоков и в руководствах по языкам AWL, KOP и FUP.

Замечание

Объект ”System Data [Системные данные]" (SDB), который, возможно, существует в программе пользователя, был создан системой. Вы можете его открыть, но не можете вносить в него изменения во избежание появления противоречивости. Он используется для внесения изменений в конфигурацию после загрузки программы и для загрузки этих изменений в программируемый контроллер.

Использование блоков из стандартных библиотек

Для создания пользовательских программ вы можете использовать также блоки из стандартных библиотек, поставляемых с программным обеспечением. Обращение к библиотекам производится с помощью команды меню **File > Open [Файл > Открыть]**. Дальнейшую информацию об использовании стандартных библиотек и о создании своих собственных библиотек вы найдете в разделе Работа с библиотеками и в оперативной помощи.

Создание исходных файлов/схем CFC

Вы хотите создать исходный файл на определенном языке программирования или схему CFC. Чтобы сделать это, выберите в программе S7 объект "Source Files [Исходные файлы]" или "Charts [Схемы]", а затем выберите команду меню **Insert > S7 Software [Вставить > Программное обеспечение S7]**. В подменю вы можете выбрать исходный файл, соответствующий вашему языку программирования. Теперь вы можете открыть пустой исходный файл и начать ввод своей программы. Больше информации вы найдете в разделе Основная информация о программировании исходных файлов на AWL.

Создание программ для M7

Вы хотите создавать программы для операционной системы RMOS для программируемых модулей серии M7. Чтобы сделать это, выделите программу M7, а затем выберите команду меню **Insert > M7 Software [Вставить > Программное обеспечение M7]**. В подменю вы можете выбрать объект, соответствующий вашему языку программирования или операционной системе. Теперь вы можете открыть созданный вами объект, чтобы получить доступ к соответствующей среде программирования.

Создание таблицы символов

Пустая таблица символов (объект ”Symbols [Символы]") создается автоматически при создании папки Программа S7/M7. Когда вы открываете таблицу символов, появляется окно ”Symbol Editor [Редактор символов]", отображающее таблицу символов, где вы можете определять символы. Дополнительную информацию вы найдете в разделе Ввод нескольких совместно используемых символов в таблицу символов.

Вставка внешних исходных файлов

Вы можете создавать и редактировать исходные файлы с помощью любого редактора ASCII. Затем вы можете импортировать эти файлы в свой проект и компилировать их для создания отдельных блоков.

Блоки, создаваемые при компиляции импортированного исходного файла, хранятся в папке ”Blocks [Блоки]".

Дополнительную информацию вы найдете под Inserting External Source Files [Вставка внешних исходных файлов].

6.3 Редактирование проекта

6.3.1 Редактирование проекта

Открытие проекта

Чтобы открыть существующий проект, введите команду меню **File > Open [Файл > Открыть]**. Затем выберите проект в открывшемся диалоговом окне. После этого открывается окно проекта.

Замечание

Если нужный вам проект не отображается в списке проектов, щелкните на кнопке "Browse [Просмотреть]". В браузере вы сможете искать другие проекты и включать любые проекты, которые вы найдете в список проектов. Записи в списке проектов вы можете изменять с помощью команды меню **File > Manage** [Файл > Управлять].

Копирование проекта

Вы можете скопировать проект, сохранив его под другим именем с помощью команды меню **File > Save As [Файл > Сохранить как...]**.

Такие части проекта, как станции, программы, блоки и т. д., вы можете копировать с помощью команды меню **Edit > Copy [Редактировать > Копировать]**.

Пошаговое руководство по копированию проекта вы найдете под Copying a Project [Копирование проекта] и Copying Part of a Project [Копирование части проекта].

Удаление проекта

Проект удаляется с помощью команды меню **File > Delete [Файл > Удалить]**.

Такие части проекта, как станции, программы, блоки и т. д., удаляются с помощью команды меню **Edit > Delete [Редактировать > Удалить]**.

Пошаговое руководство по удалению проекта вы найдете под Deleting a Project [Удаление проекта] и Deleting Part of a Project [Удаление части проекта].

7 Определение символов

7.1 Абсолютная и символическая адресация

В программе STEP 7 вы работаете с такими операндами, как сигналы входов/выходов, меркеры, счетчики, таймеры, блоки данных и функциональные блоки. Вы можете обратиться к этим операндам по их абсолютным адресам, но ваша программа будет читаться значительно легче, если вы воспользуетесь символическими именами (символами) для этих адресов (например, Двигатель_А_включить или другие идентификаторы в соответствии с системой кодов, принятой в вашей компании или отрасли промышленности). После этого к операнду в вашей пользовательской программе можно будет обратиться с помощью этого символа.

Абсолютные адреса

Абсолютный адрес состоит из идентификатора адреса и положения в памяти (например, Q 4.0, I 1.1, M 2.0, FB21).

Символические адреса

Вы можете облегчить чтение своей программы и упростить поиск неисправностей, назначив абсолютным адресам символические имена.

STEP 7 может преобразовывать символические имена в требуемые абсолютные адреса автоматически. Если вы предпочитаете обращаться к массивам, структурам, блокам данных, локальным данным, логическим блокам и типам данных, определенным пользователем, с помощью символических имен, то вы должны сначала назначить символические имена абсолютным адресам, прежде чем вы сможете обратиться к ним символически.

Например, вы можете назначить символическое имя ДВИГАТЕЛЬ_ВКЛЮЧЕН адресу Q 4.0, а затем использовать ДВИГАТЕЛЬ_ВКЛЮЧЕН как адрес в операторе программы. С помощью символических адресов легче распознавать, насколько элементы в программе соответствуют компонентам вашего проекта управления процессом.

Замечание

В символическом имени не допускается использование двух последовательных знаков подчеркивания (например, ДВИГАТЕЛЬ__ВКЛЮЧЕН) (идентификатор переменной).

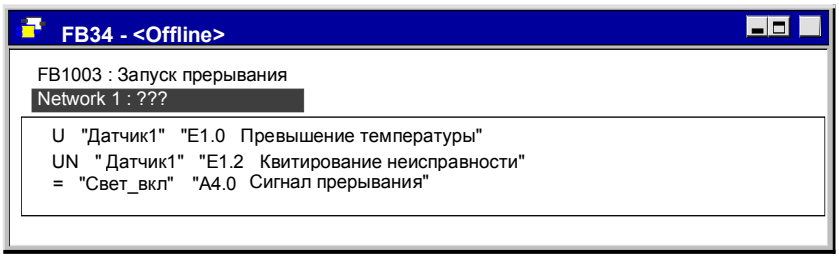
Поддержка при программировании

В языках программирования контактный план, функциональный план и список команд вы можете вводить операнды, параметры и имена блоков как абсолютные адреса и как символы.

С помощью команды меню **View > Display > Symbolic Representation [Вид > Отобразить > Символическое представление]** вы можете переключаться между абсолютным и символическим представлением адресов.

Для облегчения программирования с использованием символической адресации вы можете отображать абсолютный адрес и комментарий, связанный с символом. Эту информацию вы можете активизировать с помощью команды меню **View > Display > Symbol Information [Вид > Отобразить > Информация о символах]**. Это значит, что комментарий к строке, следующий за каждым оператором AWL, будет содержать больше информации. Вы не можете редактировать это отображение; любые изменения можно делать только в таблице символов или в таблице описания переменных.

На следующем рисунке показана символическая информация в AWL.



При распечатке блока на принтер выводится текущее представление экрана с комментариями к командам или комментариями к символам.

7.2 Совместно используемые и локальные символы

Символическое представление позволяет работать с имеющими смысл символическими именами вместо абсолютных адресов. Для облегчения программирования и улучшения документирования программы может быть эффективно использована комбинация кратких символов и длинных комментариев.

Следует различать локальные (относящиеся к блоку) и совместно используемые символы.

	Совместно используемые символы	**Локальные символы**
Область действия	Действителен во всей программе пользователя Может быть использован всеми блоками Имеет один и тот же смысл во всех блоках Должен быть уникален во всей программе пользователя	Известен только блоку, в котором был определен Один и тот же символ может быть использован в разных блоках для разных целей
Допустимые символы	Буквы, цифры, специальные символы Диакритические знаки, отличные от 0x00, 0xFF, и апострофы Символическое имя должно быть заключено в кавычки, если в нем использованы специальные символы	Буквы Цифры Знак подчеркивания (_).
Использование	Вы можете определить совместно используемые символы для: входных/выходных сигналов (E, EB, EW, ED, A, AB, AW, AD) Периферийных входов и выходов (PE, PA) меркеров (M, MB, MW, MD) таймеров (T)/ счетчиков (Z) логических блоков (FB, FC, SFB, SFC) блоков данных (DB) типов данных, определенных пользователем (UDT) таблицы переменных (VAT)	Вы можете определить локальные символы для: параметров блока (вход, выход, вход/выход), статических данных блока временных данных блока
Где определены?	Таблица символов	Таблица описания переменных для блока

7.3 Отображение совместно используемых или локальных символов

В разделе кодов программы совместно используемые и локальные символы различаются следующим образом:

- Символические имена из таблицы символов (совместно используемые) отображаются в кавычках **"**..**"**.
- Символическим именам из таблицы описания переменных блока (локальным) предшествует символ "**#**".

Вам нет необходимости вводить кавычки или "**#**". При вводе программы к KOP, FUP или AWL контроль синтаксиса добавляет эти символы автоматически.

Если вас беспокоит возможность конфликта, например, из-за того, что некоторые символические имена используются как в таблице символов, так и в таблице описания переменных, то вы должны явно кодировать совместно используемые символы, если вы хотите их использовать. В этом случае любые символы без соответствующего кодирования интерпретируются как переменные, относящиеся к блоку (локальные).

Кодирование совместно используемых символических имен необходимо также, если символическое имя содержит пробелы.

При программировании в исходном файле на AWL применяются такие же специальные символы и правила их использования. В режиме свободного редактирования кодовые символы не добавляются автоматически, но они необходимы во избежание конфликтов.

Замечание

С помощью команды меню **View > Display > Symbolic Representation [Вид > Отобразить > Символическое представление]** вы можете переключать отображение между объявленной совместно используемой символикой и абсолютными адресами.

7.4 Таблица символов для совместно используемых символических имен

7.4.1 Таблица символов для совместно используемых символических имен

Совместно используемые символические имена определяются в таблице символов.

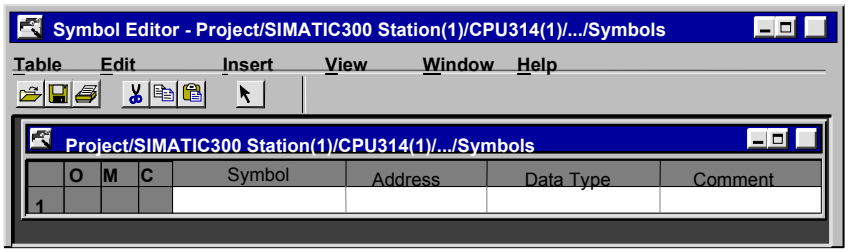
Пустая таблица символов (объект ”Symbols [Символы]") создается автоматически при создании папки S7 program [Программа S7] или M7 program [Программа M7].

Область действия

Таблица символов действительна только для модуля, с которым связана программ. Если вы хотите использовать одни и те же символические имена в нескольких различных CPU, то вы сами должны обеспечить совпадение записей в различных таблицах символов (например, копированием таблицы).

7.4.2 Структура и компоненты таблицы символов

Структура таблицы символов



Пояснения к рисунку: Symbol Editor – редактор символов; Table – таблица; Edit – редактировать; Insert – вставить; View – вид; Window – окно; Help – помощь; Symbol – символ(ическое имя); Address – адрес; Data Type – тип данных; Comment – комментарий.

Столбцы O/M/C

Столбцы O/M/C показывают, были ли символическому имени назначены специальные свойства:

- O означает, что символ может управляться и наблюдаться с помощью WinCC.
- M означает, что этому символу было назначенное относящееся к нему сообщение (SCAN).
- C означает, что символу назначены коммуникационные свойства (могут быть выбраны только с помощью NCM).

Символическое имя (Symbol)

Символическое имя не должно быть длиннее 24 символов. Таблица символов может содержать не более 16380 символических имен.

В таблице символов вы не можете назначать символические имена для адресов в блоках данных (DBD, DBW, DBB, DBX). Их имена назначаются в описании блоков данных.

Для организационных блоков (OB) и некоторых системных функциональных блоков (SFB) и системных функций (SFC) уже существуют предварительно определенные записи для таблицы символов, которые вы можете импортировать при редактировании таблицы символов для своей программы S7. Файл импорта хранится в каталоге STEP 7 под ...\S7data\Symbol\Symbol.sdf

Адрес (Address)

Адрес – это аббревиатура для определенной области памяти и положения в ней.
Пример: Вход E 12.1

Синтаксис адреса контролируется при его вводе. Проверяется также, может ли адрес быть назначен указанному типу данных.

Тип данных (Data Type)

Вы имеете возможность выбора из ряда типов данных, доступных в STEP 7. Поле типов данных уже содержит тип данных по умолчанию, который вы можете изменить, если это необходимо. Если сделанное вами изменение непригодно для данного адреса и его синтаксис неверен, то при выходе из поля появляется сообщение об ошибке.

Комментарий (Comment)

Все символическим именам могут быть назначены комментарии. Комбинирование кратких символических имен и более подробных комментариев делает создание программы более эффективным, документацию вашей программы более полной. Комментарий может иметь длину до 80 символов.

Преобразование в переменные языка C

Вы можете выбрать символические имена в таблице символов для программы M7 и, используя дополнительный пакет программ ProC/C++, преобразовать их в соответствующие переменные языка C.

7.4.3 Адреса и типы данных, разрешенные в таблице символов

По всей таблице символов должен использоваться только один набор мнемонических обозначений. Переключение между мнемоникой SIMATIC (немецкой) и мнемоникой IEC (английской) должно выполняться в SIMATIC Manager с помощью команды меню **Options > Customize [Параметры > Настройка]** в закладке "Language [Язык]".

IEC	SIMATIC	Описание	Тип данных	Диапазон значений
I	E	Входной бит	BOOL	0.0 – 65535.7
IB	EB	Входной байт	BYTE, CHAR	0 – 65535
IW	EW	Входное слово	WORD, INT, S5TIME	0 – 65534
ID	ED	Входное двойное слово	DWORD, DINT, REAL, TOD, TIME	0 – 65532
Q	A	Выходной бит	BOOL	0.0 – 65535.7
QB	AB	Выходной байт	BYTE, CHAR	0 – 65535
QW	AW	Выходное слово	WORD, INT, S5TIME	0 – 65534
QD	AD	Входное двойное слово	DWORD, DINT, REAL, TOD, TIME	0 – 65532
M	M	Меркерный бит	BOOL	0.0 – 65535.7
MB	MB	Меркерный байт	BYTE, CHAR	0 – 65535
MW	MW	Меркерное слово	WORD, INT, S5TIME	0 – 65534
MD	MD	Меркерное двойное слово	DWORD, DINT, REAL, TOD, TIME	0 – 65532
PIB	PEB	Периферийный входной байт	BYTE, CHAR	0 – 65535
PQB	PAB	Периферийный выходной байт	BYTE, CHAR	0 – 65535

IEC	SIMATIC	Описание	Тип данных	Диапазон значений
PIW	PEW	Периферийное входное слово	WORD, INT, S5TIME	0 – 65534
PQW	PAW	Периферийное выходное слово	WORD, INT, S5TIME	0 – 65534
PID	PED	Периферийное входное двойное слово	DWORD, DINT, REAL, TOD, TIME	0 – 65532
PQD	PAD	Периферийное выходное двойное слово	DWORD, DINT, REAL, TOD, TIME	0 – 65532
T	T	Таймер	TIMER	0 – 65535
C	Z	Счетчик	COUNTER	0 – 65535
FB	FB	Функциональный блок	FB	0 – 65535
OB	OB	Организационный блок	OB	1 – 65535
DB	DB	Блок данных	DB, FB, SFB, UDT	1 – 65535
FC	FC	Функция	FC	0 – 65535
SFB	SFB	Системный функциональный блок	SFB	0 – 65535
SFC	SFC	Системная функция	SFC	0 – 65535
VAT	VAT	Таблица переменных		0 – 65535
UDT	UDT	Тип данных, определенный пользователем	UDT	0 – 65535

7.4.4 Неполные и неуникальные символы в таблице символов

Неполные символы

Имеется возможность хранить неполные символические имена. Например, вы можете ввести сначала только символическое имя, а соответствующий адрес добавить позднее. Это значит, что вы можете прервать свою работу над таблицей символов в любое время, сохранить промежуточный результат и завершить свою работу в другое время. Когда же дело дойдет до использования этого символа для создания программного обеспечения (без появления какого бы то ни было сообщения об ошибке), вы должны будете уже ввести символическое имя, адрес и тип данных.

Как появляются неуникальные символы

Неуникальные символы появляются, когда вы вставляете в таблицу символов символ, имя которого и/или адрес уже были использованы в другой строке таблицы. Это значит, что как новый, так и существующий символы не уникальны.

Это происходит, например, когда вы копируете и вставляете символ, чтобы слегка изменить детали в этой копии.

Выделение не уникальных символов

В таблице символов неуникальные символы выделяются графически (цветом, шрифтом). Это изменение в их представлении означает, что они еще нуждаются в редактировании. Вы можете отобразить все символы или отфильтровать их вид так, чтобы на экране отображались только уникальные или только неуникальные символы.

Как сделать символы уникальными?

Неуникальный символ становится уникальным при изменении компонента (имени и/или адреса), который делал его неуникальным. Если два символа неуникальны, и вы изменяете один из них так, чтобы сделать его уникальным, то другой символ тоже становится уникальным.

7.5 Ввод совместно используемых символов

7.5.1 Ввод совместно используемых символов

Имеются три метода ввода символов, которые могут быть использованы для программирования на последующих этапах:

- Через диалоговое окно
  Вы открываете диалоговое окно в том окне, где вы вводите программу, и определяете новый символ или переопределяете уже существующий. Эта процедура рекомендуется для определения отдельных символов, например, если вы понимаете, что символ пропущен, или вы хотите исправить символ при записи программы. Это сохраняет ваше отображение во всей таблице символов.
- Непосредственно в таблице символов
  Вы можете вводить символы и их абсолютные адреса непосредственно в таблицу символов. Эта процедура рекомендуется, если вы хотите ввести несколько символов и в то время как вы создаете таблицу символов для проекта, уже назначенные символы отображаются на экране, облегчая обзор символов.
- Импорт таблиц символов из других редакторов таблиц
  Вы можете создавать данные для таблицы символов в любом редакторе таблиц, с которым вам удобно работать (например, Microsoft Excel), а затем импортировать созданный вами файл в таблицу символов.

7.5.2 Общие советы по вводу символов

Для ввода новых символов в таблицу символов поместите курсор в первую пустую строку таблицы и заполните ячейки. Вы можете вставить новую, пустую строку перед текущей строкой в таблице символов с помощью команды меню **Insert > Symbol [Вставить > Символ]**. Вы можете копировать и модифицировать существующие записи с помощью команд из меню редактирования (Edit). Сохраните, а затем закройте таблицу символов. Вы можете сохранить также символы, которые были определены не полностью.

При вводе в таблицу свойств символов, вам следует принять во внимание следующие особенности:

Столбец	Замечание
Symbol [Символ]	Имя должно быть уникальным для всей таблицы символов. Когда вы подтверждаете ввод в этом поле или покидаете поле, неуникальный символ выделяется. Символическое имя может содержать до 24 символов. Кавычки (") не допускаются.
Address [Адрес]	Когда вы подтверждаете ввод в этом поле или покидаете поле, производится контроль допустимости введенного адреса.
Data Type [Тип данных]	При вводе адреса этому полю автоматически назначается тип данных по умолчанию. Если вы меняете это умолчание, то программа проверяет, соответствует ли новый тип данных адресу.
Comment [Комментарий]	Вы можете ввести здесь комментарии, чтобы кратко объяснить функции символических имен (не более 80 символов). Ввод комментариев не обязателен.

7.5.3 Ввод отдельных совместно используемых символов в диалоговое окно

Описанная ниже процедура показывает, как можно изменять символы или определять новые символы в диалоговом окне во время программирования блоков без отображения таблицы символов.

Эта процедура полезна, если вы хотите отредактировать только отдельное символическое имя. Если вы хотите редактировать несколько символических имен, то вам следует открыть таблицу символов и работать с ней непосредственно.

Активизация отображения символов в блоке

Отображение символических имен в окне открытого блока активизируется с помощью команды меню **View > Display > Symbolic Representation [Вид > Отобразить > Символическое представление]**. Перед командой меню появляется метка, чтобы показать, что символическое представление активно.

Определение символов при вводе программ

1. Убедитесь, что в окне блока включено символическое представление (команда меню View > Display > Symbolic Representation [Вид > Отобразить > Символическое представление]).
2. Выберите абсолютный адрес в разделе кодов своей программы, которому вы хотите назначить символическое имя.
3. Выберите команду меню **Edit > Symbol [Редактировать > Символ]**.
4. Заполните диалоговое окно и закройте его, подтвердив свои записи, щелкнув на ”OK" и обеспечив ввод символа.

Определенный символ вводится в таблицу символов. Любые записи, которые привели бы к появлению неуникальных символов, отвергаются с сообщением об ошибке.

Редактирование в таблице символов

С помощью команды меню **Options > Symbol Table [Параметры > Таблица символов]** вы можете открыть таблицу символов для ее редактирования.

7.5.4 Ввод нескольких совместно используемых символов в таблицу символов

Открытие таблицы символов

Есть несколько путей открытия таблицы символов:

- Дважды щелкнуть на таблице символов в окне проекта.
- Выделить таблицу символов в окне проекта и выбрать команду меню **Edit > Open Object [Редактировать > Открыть объект]**.

Таблица символов для активной программы отображается в собственном окне. Теперь вы можете создавать символы или редактировать их. При открытии таблицы символов впервые после ее создания она пуста.

Ввод символов

Для ввода новых символов в таблицу символов поместите курсор в первую пустую строку таблицы и заполните ячейки. Вы можете вставить новые пустые строки перед текущей строкой в таблице символов с помощью команды меню **Insert > Symbol [Вставить > Символ]**. Вы можете копировать и модифицировать существующие записи с помощью команд из меню редактирования (Edit). Сохраните, а затем закройте таблицу символов. Вы можете сохранить также символы, которые были определены не полностью.

Сортировка символов

Записи данных в таблице символов могут быть рассортированы в алфавитном порядке по символическим именам, адресам, типам данных или комментариям.

Вы можете изменить способ сортировки таблицы с помощью команды меню **View > Sort [Вид > Сортировать]**, чтобы открыть диалоговое окно и определить вид рассортированного представления.

Фильтрация символов

Вы можете использовать фильтр для выбора подмножества записей в таблице символов.

С помощью команды меню **View > Filter [Вид > Фильтр]** вы открываете диалоговое окно "Filter [Фильтр]".

Вы можете определить критерии, которым должны удовлетворять записи, чтобы быть включенными в отфильтрованное отображение. Вы можете фильтровать в соответствии с:

- символическими именами, адресами, типами данных, комментариями
- символами, имеющими атрибут управления и наблюдения со стороны оператора, символами; символами, обладающими коммуникационными свойствами; символами для двоичных переменных, связанных с сообщениями (битовая память или вход процесса)
- символами, имеющими статус "valid [действительный]", "invalid (non-unique, incomplete) [недействительный (неуникальный, неполный)]".

Отдельные критерии объединяются с помощью логической операции И (AND). Отфильтрованные записи начинаются с указанных строк.

Если вы хотите знать больше о параметрах диалогового окна "Filter [Фильтр], откройте контекстно-чувствительную оперативную помощь, нажав F1.

7.5.5 Установка приоритетов адресов

В диалоговом окне для свойств программы S7 вы можете установить, символическое или абсолютное значение имеет приоритет, когда открываются блоки, если были сделаны изменения в таблице символов. В более ранних, чем V5, версиях STEP 7 приоритет всегда отдавался абсолютному значению.

Пример:

Сохраненный блок содержит команду "A Symbol_A", где Symbol_A определен в таблице символов для абсолютного значения I 0.1. Теперь таблица символов изменена, и блок открыт еще раз. Установка приоритета адресов оказывает следующий эффект на эту команду:

Приоритет адреса	Изменение назначения "Symbol_A = I 0.1"	Команда при открытии блока	Объяснение
Абсолютное значение	Symbol_A = I 0.2	A I 0.1	В команде отображается абсолютное значение I 0.1, так как этому адресу более не соответствует символическое имя.
Абсолютное значение	Symbol_B = I 0.1	A Symbol_B	В команде отображается новый символ для все еще действительного абсолютного значения I 0.1.
Символ	Symbol_A = I 0.2	A Symbol_A	Команда остается той же самой. Выводится сообщение, информирующее вас об изменении назначения символа.
Символ	Symbol_B = I 0.1	A Symbol_A	Команда помечается как неверная (красный текст), так как Symbol_A более не определен.

7.5.6 Экспорт и импорт таблиц символов

Вы можете экспортировать текущую таблицу символов в текстовый файл, чтобы иметь возможность редактировать ее с помощью любого текстового редактора.

Вы можете также импортировать таблицы, созданные с помощью другого приложения, в свою таблицу символов и продолжить редактирование здесь. Функция импорта может быть использована, например, для включения в таблицу символов списка соответствия переменных, созданного с помощью STEP5/ST, после конвертирования.

Форматы файлов можно выбирать из *.SDF, *.ASC, *.DIF и *.SEQ.

Правила для экспорта

Вы можете экспортировать всю таблицу символов, отфильтрованное подмножество этой таблицы или строки, выбранные в отображении таблицы.

Свойства символов, которые вы можете установить с помощью команды меню **Edit > Special Object Properties [Редактировать > Специальные свойства объекта]**, не экспортируются.

Правила для импорта

- Для часто используемых системных функциональных блоков (SFB), системных функций (SFC) и организационных блоков (OB) предварительно определенные записи для таблицы символов уже существуют в файле...\S7DATA\SYMBOL\SYMBOL.SDF, который вы можете импортировать, если это необходимо.
- Свойства символов, которые могут быть установлены с помощью команды меню **Edit > Special Object Properties [Редактировать > Специальные свойства объекта]**, не принимаются во внимание при экспорте и импорте.

7.5.7 Форматы файлов для импорта/экспорта таблицы символов

Импортированы в таблицу символов или экспортированы из нее могут быть следующие форматы файлов:

- Формат файла ASCII (ASC)
- Формат обмена данными (Data Interchange Format, DIF)
  Вы можете открывать, редактировать и сохранять DIF-файлы в Microsoft Excel.
- Формат системных данных (System Data Format, SDF)
  Вы можете открывать, редактировать и сохранять SDF-файлы в Microsoft Access.
- Для импорта и экспорта данных в приложение Microsoft Access и из него используйте формат файла SDF.
- В Access выберите формат файла ”Text (with delimiters) [Текст (с ограничителями)]".
- Используйте двойные кавычки (") в качестве ограничителя текста.
- Используйте запятую (,)в качестве ограничителя ячеек.
- Список назначений (SEQ)
  Предостережение: При экспорте таблицы символов в файл типа .SEQ комментарии, имеющие длину более 40 символов, обрезаются после 40-го символа.

Формат файла ASCII (ASC)

Тип файла	*.ASC
Структура:	Длина записи, запятая-ограничитель, запись
Пример:	126,green_phase_ped. T 2 TIMER Длительность зеленой фазы для пешеходов 126,red_ped. Q 0.0 BOOL Красный для пешеходов

Формат обмена данными (DIF)

Тип файла	*.DIF
Структура:	DIF-файл состоит из заголовка файла и данных:

Заголовок	TABLE [ТАБЛИЦА]	Запуск DIF-файла
	0,1	
	"<Заголовок>"	Строка комментария
	VECTORS [векторы]	Количество записей в файле
	0,<число записей>	
	""	
	TUPLES [кортежи]	Количество полей с данными в записи
	0,<число столбцов>	
	""	
	DATA [данные]	Идентификатор конца заголовка и начало данных
	0,0	
	""	
Данные (на запись)	<тип>,<числовое значение>	Идентификатор типа данных, числовое значение
	<Строка>	Алфавитно-цифровая часть или
	V	Алфавитно-цифровая часть не используется

Заголовок: заголовок файла должен содержать типы записей TABLE, VECTORS, TUPLES и DATA в указанном порядке. Перед данными (DATA) DIF-файлы могут содержать, кроме того, необязательные типы записей. Они, однако, игнорируются редактором символов.

Данные: в разделе данных каждый элемент состоит из трех частей: идентификатора типа данных, числового значения и алфавитно-цифровой части.

Вы можете открывать, редактировать и сохранять DIF-файлы в Microsoft Excel. Вы не должны использовать диакритические знаки, умлауты или специальные лингвистические символы.

Формат системных данных (SDF)

Тип файла	*.SDF
Структура:	Строки в кавычках, части разделены запятыми
Пример:	"green_phase_ped.","T 2","TIMER","Длительность зеленой фазы для пешеходов" "red_ped.","Q 0.0","BOOL","Красный для пешеходов"

Для открытия SDF-файла в Microsoft Access вы должны выбрать формат файла 'Text (with delimiter) [Текст (с ограничителем)]'. Используйте двойные кавычки (") в качестве ограничителя текста и запятую (,)в качестве ограничителя полей.

Список назначений (SEQ)

Тип файла	*.SEQ
Структура:	TAB Адрес TAB Символ TAB Комментарий CR
Пример:	T 2 green_phase_ped. Длительность зеленой фазы для пешеходов Q 0.0 red_ped. Красный для пешеходов

TAB означает клавишу табуляции (09H),
CR означает возврат каретки с помощью клавиши RETURN (0DH).

8 Создание блоков и библиотек

8.1 Выбор метода редактирования

В зависимости от языка программирования, который вы используете для создания программы, вы можете вводить свою программу в режиме пошагового (инкрементного) ввода и/или в режиме свободного редактирования текста.

Редакторы пошагового ввода для языков программирования контактный план, функциональный план, список команд и S7 Graph

В редакторах пошагового ввода для KOP, FUP, AWL и S7 Graph вы создаете **блоки**, которые хранятся в программе пользователя. Вам следует выбрать режим пошагового ввода, если вы хотите контролировать то, что вы вводите, немедленно. Этот режим особенно пригоден для начинающих. В режиме пошагового ввода синтаксис каждой строки или элемента проверяется немедленно, как только они вводятся. Любая ошибка отображается и должна быть исправлена до завершения ввода. Синтаксически правильные вводы автоматически компилируются и хранятся в программе пользователя.

Использование любого символа должно быть определено до редактирования команд. Если какие-то символы недоступны, то блок не может быть скомпилирован полностью; эта внутренне противоречивая версия может быть, однако, сохранена.

Редакторы свободного редактирования (текстовые) для языков программирования список команд, S7 SCL и S7 HiGraph

В редакторах, предназначенных для режима свободного редактирования, вы создаете **исходные файлы**, которые затем компилируются в блоки.

Вам следует выбирать режим свободного редактирования для быстрого ввода программы.

В режиме свободного редактирования программа или блок редактируется в текстовом файле, а затем текстовый файл компилируется.

Текстовые файлы (исходные файлы) хранятся в папке исходных файлов вашей программы S7, например, как **исходный файл на AWL** или **исходный файл на SCL**. Исходный файл может содержать код для одного или нескольких блоков. С помощью текстовых редакторов для AWL и SCL вы можете генерировать код для **OB, FB, FC, DB и UDT** (типы данных, определенные пользователем), но вы можете создать и всю программу пользователя. Вся программа для CPU (т. е. все блоки) может содержаться в одном единственном текстовом файле.

При компиляции исходного файла соответствующие блоки создаются и сохраняются в программе пользователя. Любые используемые символы должны быть определены до компиляции. Любые ошибки сообщаются соответствующим компилятором во время компиляции.

Для компиляции важно придерживаться определенного синтаксиса, соответствующего языку программирования. Проверка синтаксиса происходит только тогда, когда вы выбираете команду контроля непротиворечивости или когда исходный файл компилируется в блоки.

8.2 Выбор языка программирования

8.2.1 Выбор языка программирования

Установка языка программирования для редактора

Какой язык программирования и какой тип редактора вы хотите использовать, вы устанавливаете в свойствах объекта при создании конкретного блока или исходного файла. Это свойство определяет, какой редактор будет запускаться при открытии блока или исходного файла.

Запуск редактора

Необходимый языковый редактор запускается в SIMATIC Manager двойным щелчком на соответствующем объекте (блоке, исходном файле и т. д.), выбором команды меню **Edit > Open Object [Редактировать > Открыть объект]** или выбором соответствующей кнопки на панели инструментов.

Для создания программы S7 в вашем распоряжении имеются языки программирования, перечисленные в таблице. Такие типы представления языка программирования STEP 7, как KOP, FUP и AWL, включены в стандартный пакет программного обеспечения STEP 7. Вы можете купить другие языки программирования в виде дополнительных пакетов программ.

Затем у вас появляется выбор из ряда различных философий программирования (контактный план, функциональный план, список команд, язык высокого уровня, последовательное управление или граф состояний) и выбор между текстовым и графическим языком программирования.

Выбором языка программирования вы определяете также допустимый тип режима ввода (•).

Язык программирования	Группа пользователей	Применение	Пошаговый ввод	Режим свободного редактирования	Блок может быть документирован из CPU
Список команд AWL	Пользователи, предпочитающие программировать на языке, подобном машинному коду	Программы, оптимальные с точки зрения времени выполнения и требований к памяти	•	•	•
Контактный план KOP	Пользователи, привыкшие работать со схемами соединений	Программирование устройств логического управления	•	–	•
Функциональный план FUP	Пользователи, знакомые с логическими блоками булевой алгебры	Программирование устройств логического управления	•	–	•
SCL (Structured Control Language, Язык структурного управления) Дополнительный пакет	Пользователи, программировавшие на таких языках высокого уровня, как PASCAL или C	Программирование задач обработки данных	–	•	–
S7 Graph Дополнительный пакет	Пользователи, желающие работать с ориентацией на технологические функции с малым объемом программирования или не имеющие опыта работы с ПЛК	Удобное описание последовательных процессов	•	–	•
HiGraph Дополнительный пакет	Пользователи, желающие работать с ориентацией на технологические функции с малым объемом программирования или не имеющие опыта работы с ПЛК	Удобное описание асинхронных, не последовательных процессов	–	•	–
CFC Дополнительный пакет	Пользователи, желающие работать с ориентацией на технологические функции с малым объемом программирования или не имеющие опыта работы с ПЛК	Описание непрерывных процессов	–	–	–

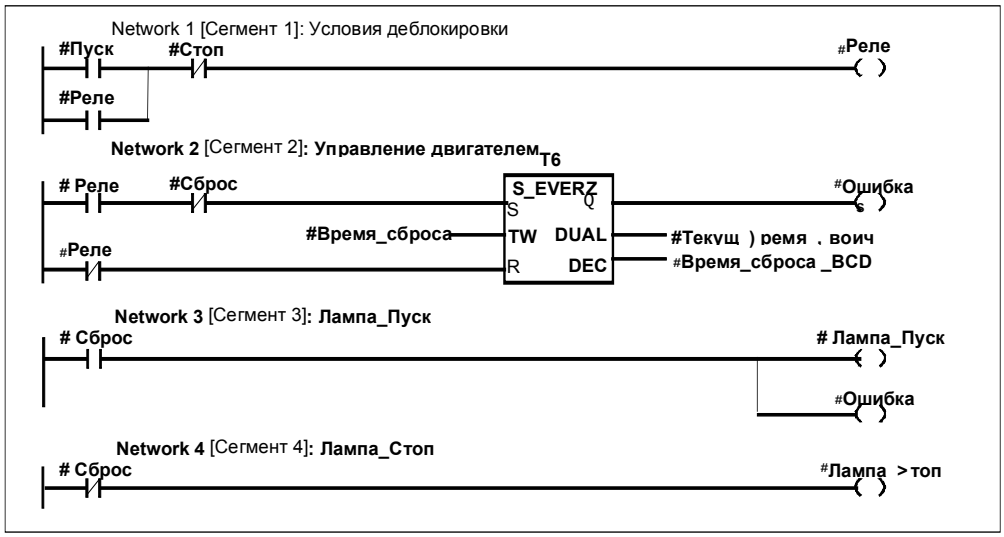
Если блоки не содержат ошибок, то вы можете переключаться между представлениями ваших блоков в виде контактного плана, функционального плана или списка команд. Части программы, которые не могут быть отображены на языке, к которому вы перешли, отображаются в виде списка команд.

Вы можете создавать блоки из исходных файлов в списке команд, а также декомпилировать их обратно в исходные файлы.

8.2.2 Язык программирования Контактный план (КОР)

Графический язык программирования Контактный план (КОР) основан на представлении коммутационных схем. Элементы коммутационной схемы, такие как нормально открытые контакты и нормально замкнутые контакты, группируются в сегменты. Один или несколько сегментов образуют раздел кодов логического блока.

Пример сегментов в контактном плане



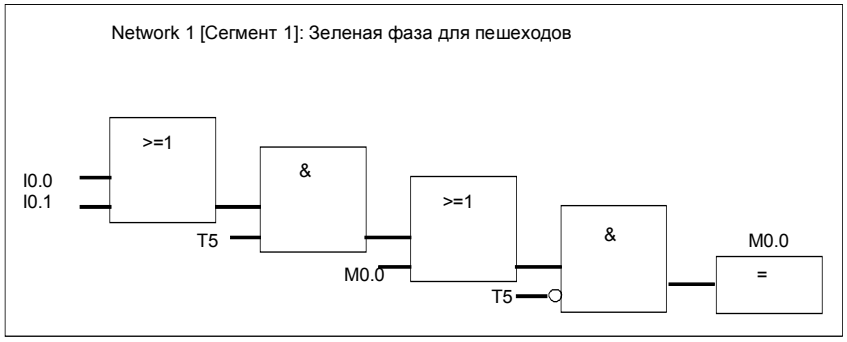
Язык программирования Контактный план включен в стандартный пакет программного обеспечения STEP 7. Создание программ в контактном плане выполняется в редакторе пошагового ввода.

8.2.3 Язык программирования Функциональный план (FUP)

Язык программирования Функциональный план (FUP) использует для представления логики графические логические символы, известные из булевой алгебры. Сложные функции, такие как математические, также могут быть представлены непосредственно в соединении с логическими блоками.

Язык программирования FUP включен в стандартный пакет программного обеспечения STEP 7.

Пример сегмента в FUP



Программы в FUP создаются в редакторе пошагового ввода.

8.2.4 Язык программирования Список команд (AWL)

Представление языка программирования Список команд (AWL) – это текстовый язык, подобный машинному коду. Каждая команда соответствует шагу работы CPU при обработке программы. Несколько команд могут быть связаны друг с другом, образуя сегменты.

Пример сегментов в списке команд

```
        Network 1[Сегмент 1]: Управление выпускным клапаном
A(
 O   #Открыть
 O   #Катушка
)
AN   #Закрыть
=    # Катушка
        Network 2 [Сегмент 2]: Отобразить "Клапан открыт"
A   # Катушка
=    #Открыт

        Network 3 [Сегмент 3]: Отобразить "Клапан закрыт"
AN   # Катушка
=    #Закрыт
```

Язык программирования Список команд включен в стандартный пакет программного обеспечения STEP 7. Вы можете редактировать блоки S7 в этом представлении языка с помощью редакторов пошагового ввода или создавать свою программу с помощью редактора, работающего в режиме свободного редактирования в исходном файле на AWL, а затем компилировать ее в блоки.

8.2.5 Язык программирования S7 SCL

Язык программирования SCL (Structured Control Language [Структурированный язык управления]), доступный как дополнительный пакет, – это текстовый язык высокого уровня, определение которого в целом соответствует стандарту Международной электротехнической комиссии IEC 1131-3. Этот паскалеобразный язык благодаря своим командам высокого уровня упрощает в сравнении с AWL программирование циклов и условных переходов. Поэтому SCL пригоден для расчетов, включая формулы, сложные оптимизационные алгоритмы или управление большими объемами данных

Создание программ на S7 SCL производится в режиме свободного редактирования в исходном файле.

Пример:

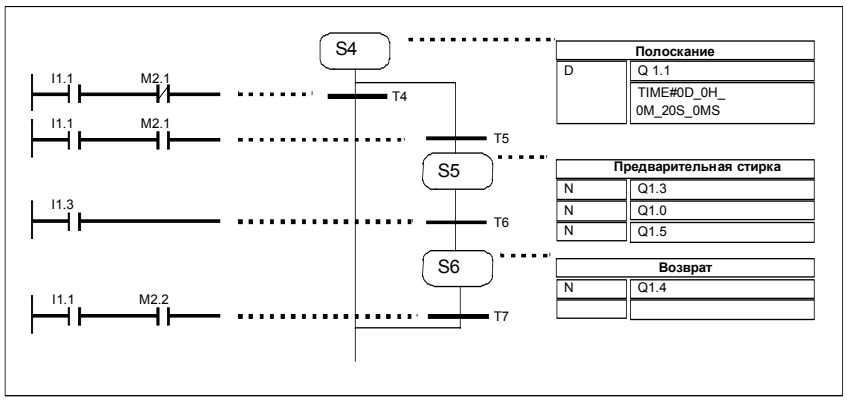
```
FUNCTION_BLOCK FB20
VAR_INPUT
ENDVAL:                 INT;
END_VAR
VAR_IN_OUT
IQ1 :           REAL;
END_VAR
VAR
INDEX:                  INT;
END_VAR

BEGIN
CONTROL:=FALSE;
FOR INDEX:= 1 TO ENDVALUE DO
    IQ1:= IQ1 * 2;
    IF IQ1 >10000 THEN
            CONTROL = TRUE
    END_IF
END_FOR;
END_FUNCTION_BLOCK
```

8.2.6 Язык программирования S7 Graph (последовательное управление)

Графический язык программирования S7 Graph, доступный в виде дополнительного пакета, дает возможность программирования устройств последовательного управления. Это включает в себя создание последовательности шагов, определение содержания каждого шага и определение переходов. Вы программируете содержание шагов на специальном языке программирования (похожем на список команд) и вводите переходы в редакторе цепных логических схем (модернизированная версия языка КОР).

S7 Graph очень ясно представляет сложные последовательности и делает программирование и поиск неисправностей более эффективными.

Пример последовательного управления в S7 Graph



Создаваемые блоки

С помощью редактора S7 Graph программируется функциональный блок, который содержит генератор последовательности шагов. Соответствующий экземплярный блок данных содержит данные для этого генератора, например, параметры FB, условия для шагов и переходов. Вы можете обеспечить автоматическое создание этого экземплярного блока данных в редакторе S7 Graph.

Исходный файл

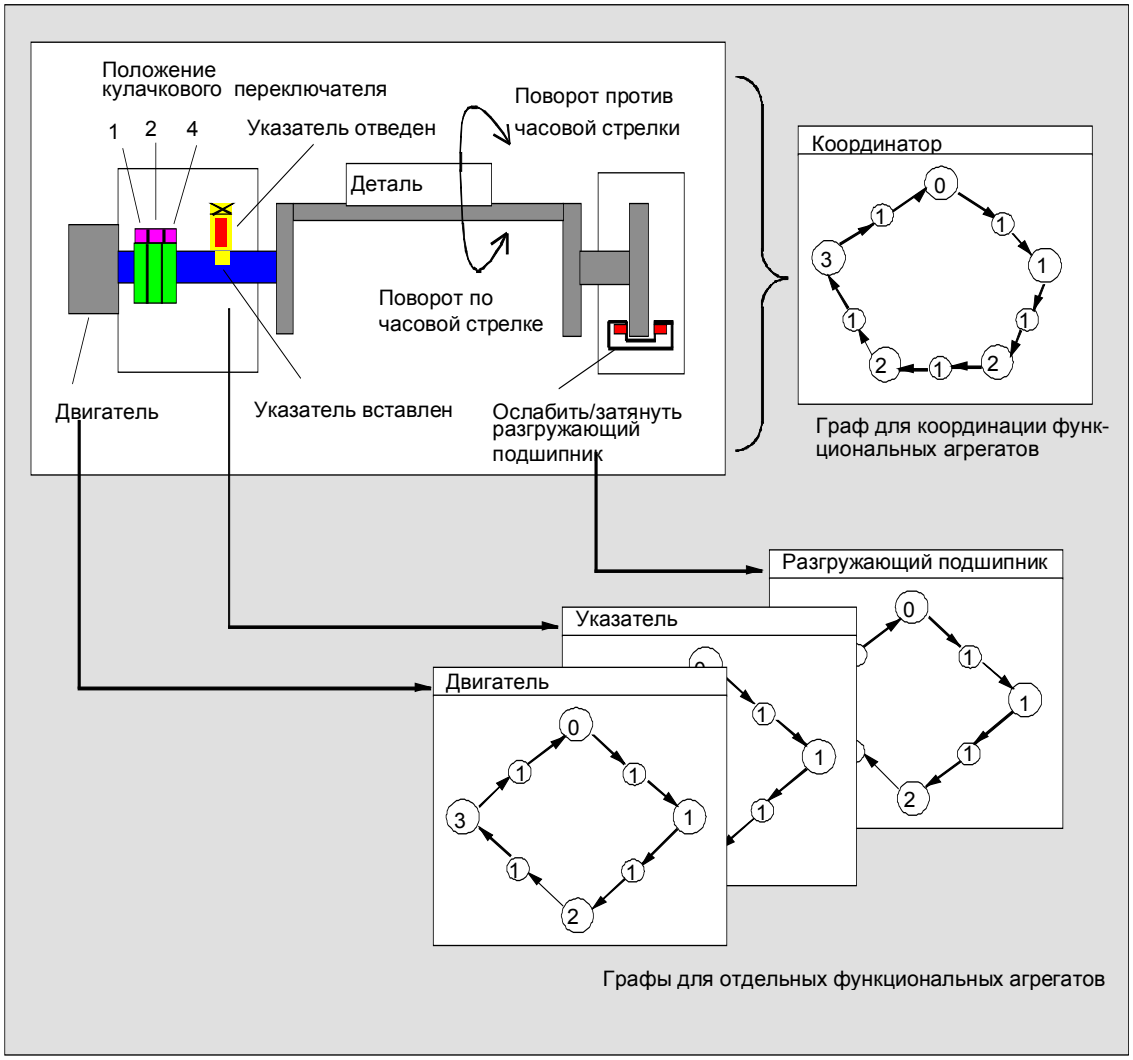
Из функционального блока, созданного в S7 Graph, может быть сгенерирован текстовый исходный файл, который может интерпретироваться панелями оператора или текстовыми дисплеями интерфейса с оператором для отображения генератора последовательности шагов.

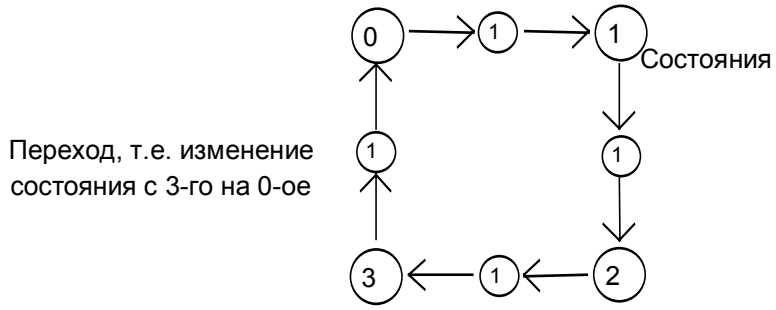
8.2.7 Язык программирования S7 HiGraph (граф состояний)

Графический язык программирования S7 HiGraph, доступный в качестве дополнительного пакета, позволяет программировать ряд блоков в вашей программе как графы состояний. Это разделяет вашу установку на отдельные функциональные агрегаты, каждый из которых может принимать различные состояния. Для изменения состояний определяются переходы. Вы описываете действия, поставленные в соответствие состояниям, и условия для переходов между состояниями на языке, похожем на список команд.

Вы создаете граф для каждого функционального агрегата, который описывает поведение этого агрегата. Графы для установки объединяются в группы графов. Для синхронизации функциональных агрегатов между графами может производиться обмен сообщениями.

Ясное представление переходов между состояниями функционального агрегата делает возможным систематическое программирование и облегчает поиск ошибок. В отличие от S7 Graph, в S7 HiGraph в каждый момент времени активно только одно состояние (в S7 Graph: "шаг"). На следующем рисунке показано, как создавать графы для функциональных агрегатов (пример).





Группа графов хранится в исходном файле HiGraph в папке "Source Files [Исходные файлы]" под программой S7. Затем исходный файл компилируется в блоки S7 для программы пользователя.

Синтаксис и формальные параметры проверяются на последнем элементе графа (при закрытии рабочего окна). Адреса и символы проверяются при компиляции исходного файла.

8.2.8 Язык программирования S7 CFC

Дополнительный пакет программного обеспечения CFC (*Continuous Function Chart [Схема непрерывных функций]*) – это язык программирования, используемый для графического связывания сложных функций.

Язык программирования S7 CFC используется для связывания существующих функций. Вам нет необходимости программировать самим многие стандартные функции, вместо этого вы можете использовать библиотеки, содержащие стандартные блоки (например, для логических, математических функций, функций управления и обработки данных). Для использования CFC вам не нужны детальные знания в области программирования или специальные знания о программном управлении, и вы можете сосредоточиться на технологии, используемой в вашей отрасли промышленности.

Созданная программа хранится в виде схем CFC. Они находятся в папке "Charts [Схемы]" под программой S7. Эти схемы затем компилируются для формирования блоков S7 для программы пользователя.

Возможно, вы сами захотите создать подлежащие соединению блоки, в этом случае вы программируете их для SIMATIC S7 с помощью одного из языков программирования S7, а для SIMATIC M7 – с помощью C/C++.

8.3 Создание боков

8.3.1 Папка блоков

Вы можете создать программу для CPU S7 в виде:

- блоков
- исходных файлов.

Папка "Blocks [Блоки]" доступна для хранения блоков под программой S7..

Эта папка блоков содержит блоки, необходимые вам для загрузки в CPU S7 для решения вашей задачи автоматизации. Эти загружаемые блоки включают в себя логические блоки (OB, FB, FC) и блоки данных (DB). Пустой организационный блок OB1 автоматически создается вместе с папкой блоков, так как вам всегда потребуется этот блок для исполнения вашей программы в CPU S7.

Папка блоков содержит также следующие объекты:

- Типы данных, определенные пользователем (UDT), которые созданы вами. Они облегчают программирование, но не загружаются в CPU.
- Таблицы переменных (VAT), которые вы можете создать для наблюдения и изменения переменных для отладки своей программы. Таблицы переменных не загружаются в CPU.
- Объект "System Data [Системные данные]" (блоки системных данных), содержащий системную информацию (конфигурацию и параметры системы). Эти системные блоки данных создаются и снабжаются данными, когда вы конфигурируете аппаратуру.

- Системные функции (SFC) и системные функциональные блоки (SFB), нужные вам для вызова в вашей пользовательской программе. Вы не можете сами редактировать SFC и SFB.

За исключением системных блоков данных (которые могут быть созданы и отредактированы только через программу конфигурирования программируемого логического контроллера), все блоки в программе пользователя редактируются с помощью соответствующего редактора. Этот редактор запускается автоматически при двойном щелчке на соответствующем блоке.

Замечание

Блоки, запрограммированные в виде исходных файлов, а затем скомпилированные, тоже хранятся в папке блоков.

8.3.2 Типы данных, определенные пользователем (UDT)

Типы данных, определенные пользователем, – это специальные структуры данных, создаваемые вами самими, которые вы можете использовать во всей программе S7, как только они были определены.

- Типы данных, определенные пользователем, могут использоваться, как и элементарные или составные типы данных, в описании переменных логических блоков (FC, FB, OB) или как тип данных для переменных блока данных (DB). Их преимущество состоит в том, что вам нужно определить специальную структуру данных только один раз, чтобы иметь возможность использовать ее столько раз, сколько вы желаете, и назначать ее любому количеству переменных.
- Типы данных, определенные пользователем, могут использоваться как шаблон для создания блоков данных с одинаковой структурой данных, т. е. вы создаете структуру один раз, а затем создаете блоки данных простым назначением типа данных, определенного пользователем. (Пример: Рецепт: структура блока данных всегда одна и та же, различны только используемые количества компонентов).

Типы данных, определенные пользователем, создаются в SIMATIC Manager или в редакторе пошагового ввода аналогично другим блокам.

Структура типа данных, определенного пользователем

Когда вы открываете тип данных, определенный пользователем, на экране появляется новое рабочее окно, отображающее описание этого типа данных, определенного пользователем, в табличной форме.

- Первая и последняя строка уже содержат описания STRUCT и END_STRUCT для начала и конца типа данных, определенного пользователем. Эти строки вы редактировать не можете.
- Тип данных, определенный пользователем, редактируется вводом ваших элементов в соответствующие столбцы, начиная со второй строки таблицы описаний.
- Структура типов данных, определенных пользователем, может состоять из:
- элементарных типов данных
- составных типов данных
- существующих типов данных, определенных пользователем.

Типы данных, определенные пользователем, в программе S7 не загружаются в CPU S7. Они или создаются непосредственно с использованием редактора пошагового ввода и редактируются, или создаются при компиляции исходных файлов.

8.3.3 Свойства блоков

Вы можете более легко идентифицировать создаваемые вами блоки, используя свойства блоков. Вы можете также защитить эти блоки от несанкционированных изменений.

Свойства блока следует редактировать, когда он открыт. Кроме свойств, которые вы можете редактировать, диалоговое окно свойств отображает также данные только для информации: эту информацию вы редактировать не можете.

Свойства блока и системные атрибуты отображаются также в SIMATIC Manager в свойствах объекта для блока. Здесь вы можете редактировать только свойства NAME [имя], FAMILY [семейство], AUTHOR [автор] и VERSION [версия].

Свойства объекта редактируются после вставки блока через SIMATIC Manager. Если блок был создан с помощью одного из редакторов, а не в SIMATIC Manager, эти элементы (язык программирования) сохраняются автоматически в свойствах объекта.

Замечание

Мнемоника, которую вы хотите использовать для программирования блоков S7, может быть установлена с помощью команды меню **Options > Customize [Параметры > Настройка]** в закладке появляющегося диалогового окна "Language [Язык]".

Таблица свойств блока

При вводе свойств блока вы должны соблюдать последовательность, показанную в следующей таблице:

Ключевое слово / свойство	Значение	Пример
[KNOW_HOW_PROTECT]	Защита блока; блок, скомпилированный с этой опцией, не позволяет просматривать свой раздел кодов.	KNOW_HOW_PROTECT
[AUTHOR:]	Имя автора: название компании, отдела или другое имя (не более 8 символов без пробелов)	AUTHOR : Siemens, но не ключевое слово
[FAMILY:]	Название семейства блоков: например, controllers (не более 8 символов без пробелов)	FAMILY : controllers, но не ключевое слово
[NAME:]	Имя блока (не более 8 символов)	NAME : PID, но не ключевое слово
[VERSION: int1 . int2]	Номер версии блока (оба числа между 0 и 15, т. е. от 0.0 до 15.15)	VERSION : 3.10
[CODE_VERSION1]	Идентификатор того, может ли функциональный блок иметь мультиэкземпляры, описанные или нет. Если вы хотите описать мультиэкземпляры, то функциональный блок не должен иметь этого свойства	CODE_VERSION1
[UNLINKED] только для DB	Блок данных со свойством UNLINKED не связан с программой.	
[READ_ONLY] только для DB	Защита от записи для блоков данных; их данные могут быть прочитаны, но не могут быть изменены	FAMILY= Examples [Примеры] VERSION= 3.10 READ_ONLY

Защита блока KNOW_HOW_PROTECT [защита ноу-хау] имеет следующие последствия:

- Если вы захотите посмотреть скомпилированный блок позднее в редакторе пошагового ввода AWL, FUP или КОР, то раздел кодов блока не будет отображаться на экране.
- Таблица описания переменных блока отображает только переменные типов var_in, var_out и var_in_out. Переменные типов var_stat и var_temp остаются скрытыми.

Соответствие: свойство блока – тип блока

В следующей таблице показано, какие свойства для каких типов блоков могут быть описаны:

Свойство	OB	FB	FC	DB	UDT
KNOW_HOW_PROTECT	•	•	•	•	–
AUTHOR	•	•	•	•	–
FAMILY	•	•	•	•	–
NAME	•	•	•	•	–
VERSION	•	•	•	•	–
UNLINKED	–	–	–	•	–
READ_ONLY	–	–	–	•	–

Свойство KNOW_HOW_PROTECT может быть установлено в исходном файле при программировании блока. Оно отображается в диалоговом окне "Block Properties [Свойства блока]", но не может быть изменено.

8.3.4 Атрибуты для блоков и параметров

Описание атрибутов можно найти в помощи по системным атрибутам.

8.4 Работа с библиотеками

8.4.1 Работа с библиотеками

Библиотеки служат для хранения повторно используемых программных компонентов для SIMATIC S7/M7. Программные компоненты могут быть скопированы в библиотеку из существующих проектов или созданы непосредственно в библиотеке независимо от других проектов.

Вы можете сэкономить себе много усилий и времени на программирование, если вы храните блоки, которые вы хотите использовать многократно, в библиотеке в программе S7. Вы можете копировать их оттуда в программу пользователя, где они необходимы.

Для создания программ S7/M7 в библиотеке применимы те же функции, что и для проектов, за исключением отладки.

Создание библиотек

Вы можете создавать библиотеки совершенно так же, как и проекты, используя команду меню **File > New [Файл > Новый]**. Новая библиотека создается в каталоге, который вы назначили для библиотек в закладке "General [Общие свойства]", когда вы выбрали команду меню **Options > Customize [Параметры > Настройка]**.

Замечание

SIMATIC Manager допускает имена, имеющие длину более восьми символов. Однако, имя библиотечного каталога урезается до восьми символов. Поэтому имена библиотек должны различаться в своих первых восьми символах. Имена не чувствительны к регистру. Когда каталог открывается в браузере, снова отображается полное имя, но при поиске каталога появляется только усеченное имя.

Имейте в виду, что вы не можете использовать блоки из библиотек новой версии STEP 7 в проектах старой версии STEP 7.

Открытие библиотек

Для открытия существующей библиотеке введите команду меню **File > Open [Файл > Открыть]**. Затем выберите библиотеку в открывшемся диалоговом окне. После этого открывается окно библиотеки.

Замечание

Если вы не можете найти нужную вам библиотеку, щелкните на кнопке "Browse [Просмотреть]" в диалоговом окне " Open [Открыть]". После этого стандартный браузер Windows отображает структуру каталогов, в которой вы можете искать библиотеку.

Обратите внимание на то, имя файла всегда соответствует первоначальному имени библиотеки, когда она была создана, т. е. любые изменения имени, сделанные в SIMATIC Manager, не реализуются на файловом уровне.

При выборе библиотеки она добавляется к списку библиотек. Вы можете изменять записи в списке библиотек с помощью команды меню **File > Manage [Файл > Управлять]**.

Копирование библиотек

Библиотека копируется путем сохранения ее под другим именем с помощью команды меню **File > Save As [Файл > Сохранить как...]**.

Часть библиотеки копируется так же, как программы, блоки, исходные файлы и т. д., с помощью команды меню **Edit > Copy [Редактировать > Копировать]**.

Удаление библиотеки

Библиотека удаляется с помощью команды меню **File > Delete [Файл > Удалить]**.

8.4.2 Иерархическая структура библиотек

Библиотеки, как и проекты, имеют иерархическую структуру:

- Библиотеки могут содержать программы S7/M7.
- Программа S7 может содержать одну папку "Blocks [Блоки]" (программа пользователя), одну папку "Source Files [Исходные файлы]", одну папку "Charts [Схемы]" и один объект "Symbols [Символы]" (таблица символов).
- Программа M7 может содержать схемы и программу на языке C для программируемых модулей M7, а также объект "Symbols [Символы]" (таблица символов) и папку "Blocks [Блоки]" для блоков данных и таблиц переменных.
- Папка "Blocks [Блоки]" содержит блоки, которые могут быть загружены в CPU S7. Таблицы переменных (VAT) и типы данных, определенные пользователем, в этой папке не загружаются в CPU.
- Папка "Source Files [Исходные файлы]" содержит исходные файлы для программ, созданных на различных языках программирования.
- Папка "Charts [Схемы]" содержит схемы CFC (только в случае, если установлен дополнительный пакет программного обеспечения S7 CFC).

Когда вы вставляете новую программу S7/M7, папка "Blocks [Блоки]", папка "Source Files [Исходные файлы]" и объект "Symbols [Символы]" вставляются в нее автоматически.

8.4.3 Обзор стандартных библиотек

Стандартный пакет STEP 7 содержит стандартные библиотеки (версия 2/ версия 3):

- stlibs (V2): стандартная библиотека версии 2
- stlib3.x: стандартная библиотека версии 3

Стандартные библиотеки содержат следующие компоненты:

- **builtin/Built In:** системные функциональные блоки (SFB) и системные функции (SFC)
- **fblib1/FB Lib 1:** блоки для конвертирования программ STEP 5
- **fblib2/FB Lib 2:** стандартные функции для общего использования
- **iec/IEC:** блоки для функций IEC, таких как для обработки информации о дате и времени, для операций сравнения, для обработки строк и для выбора максимума и минимума
- **stdobs/Std OBs:** стандартные организационные блоки (OB)

Стандартная библиотека для версии 3 содержит также следующие компоненты:

- **PID Control:** функциональные блоки (FB) для PID-регулятора
- **Net DP:** функции (FC) для децентрализованной периферии и FDL-соединений

При установке дополнительных программных пакетов могут быть добавлены другие библиотеки.

Удаление и установка поставляемых библиотек

Вы можете удалить поставляемые библиотеки в SIMATIC Manager, а затем вновь их установить. Для установки библиотек вы должны снова выполнить программу установки (Setup) STEP 7 V5.0 с самого начала.

Замечание

Когда вы устанавливаете STEP 7, поставляемые библиотеки всегда копируются. Если вы редактируете эти библиотеки, то измененные библиотеки при повторной установке STEP 7 будут переписаны оригинальными.

Поэтому вы должны скопировать поставляемые библиотеки перед выполнением изменений, а затем редактировать только копии.

9 Создание логических блоков

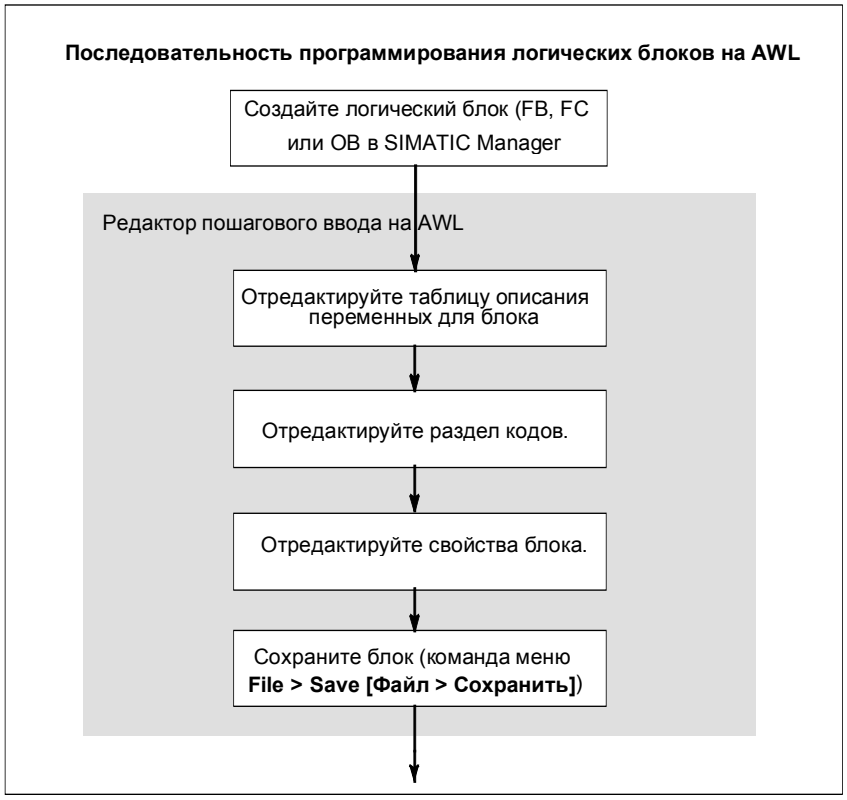
9.1 Основы создания логических блоков

9.1.1 Основная последовательность действий для создания логических блоков

Логические блоки (OB, FB, FC) включают в себя раздел описания переменных, раздел кодов, а также имеют свойства. При программировании вы должны редактировать следующие три части:

- **Таблица описания переменных**: В таблице описания переменных вы определяете параметры, системные атрибуты для параметров и локальные переменные блока.
- **Раздел кодов**: В разделе кодов вы программируете код блока для обработки программируемым контроллером. Он состоит из одного или нескольких сегментов. Для создания сегментов вы можете использовать, например, языки программирования контактный план (КОР), функциональный план (FUP) или список команд (AWL).
- **Свойства блока**: Свойства блока содержат дополнительную информацию, такую как метка времени или путь, вводимый системой. Кроме того, вы можете ввести свои собственные детали, относящиеся к имени, семейству, версии и автору, и вы можете назначить системные атрибуты для блоков.

В принципе не имеет значения, в каком порядке редактируются эти части логического блока. Конечно, вы можете также корректировать их и делать дополнения.



Замечание

Если вы хотите использовать символы в таблице символов, то вам сначала следует проверить их полноту и сделать необходимые корректировки.

9.1.2 Установки по умолчанию для программного редактора КОР/ AWL/FUP/ (LAD/STL/FBD)

Перед началом программирования вы должны познакомиться с настройками в редакторе, чтобы облегчить себе и сделать более удобным программирование.

С помощью команды меню **Options > Customize [Параметры > Настройка]** вы открываете диалоговое окно с закладками. В закладке "Editor [Редактор]" вы можете сделать следующие установки по умолчанию для программирования блоков:

- Шрифт (тип и размер) в тексте и таблицах.
- Языковое представление, которое вы предпочитаете использовать (КОР, AWL или FUP). В соответствии с вашим вводом блок в дальнейшем создается или открывается в КОР, AWL или FUP. Вы все еще можете просматривать блок на следующем этапе в другом представлении языка, однако, с некоторыми ограничениями.
- Хотите ли вы, чтобы символы и комментарии отображались с новым блоком.

Вы можете изменить эти настройки для языка, комментариев и символов во время редактирования с помощью команд в меню **View [Вид]**.

Вы можете изменить цвета для выделения, например, сегментов или строк команд в закладке "LAD/FBD (или КОР/FUP)"

9.1.3 Права доступа к блокам и исходным файлам

При редактировании проекта часто используется общая база данных, имея в виду, что несколько человек могут захотеть получить доступ к одному и тому же блоку или источнику данных одновременно.

Права доступа на чтение/запись назначаются следующим образом:

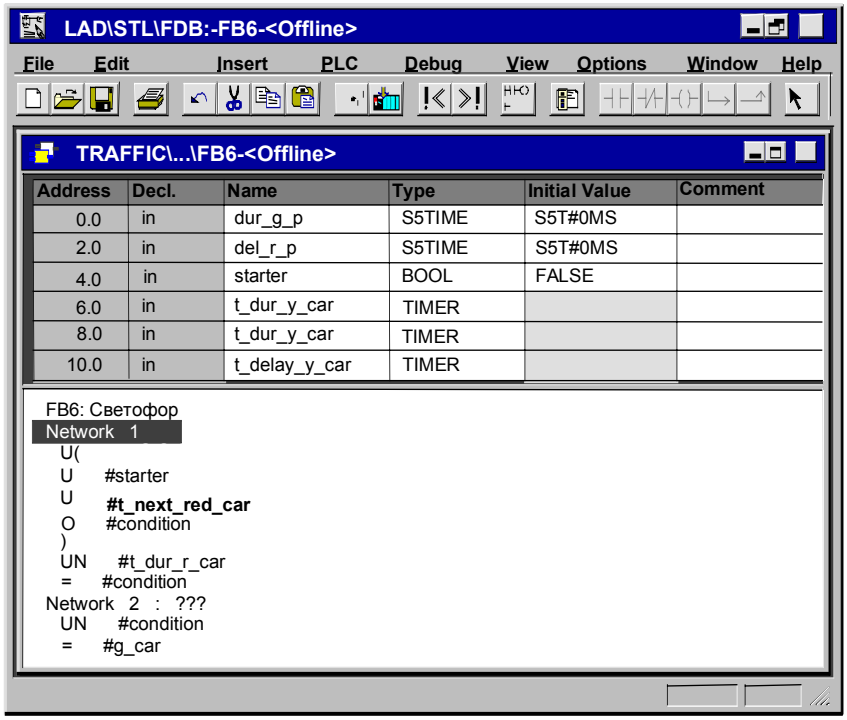
- Редактирование offline:
  Когда вы пытаетесь открыть блок/исходный файл, проверяется, имеете ли вы право доступа к объекту на 'запись'. Если блок/исходный файл уже открыт, вы можете работать только с копией. Если вы затем пытаетесь сохранить копию, система запрашивает, хотите ли вы переписать оригинал или сохранить копию под новым именем.

- Редактирование online:
  Когда вы открываете блок online через сконфигурированное соединение, соответствующий блок offline блокируется, предотвращая его одновременное редактирование.

9.1.4 Команды из каталога элементов программы

Каталог элементов программы предоставляет список элементов KOP, AWL и FUP, а также уже описанные мультиэкземпляры, уже запрограммированные блоки и блоки из библиотек. К нему можно обратиться с помощью команды меню **View > Catalog [Вид > Каталог]**. Элементы программы могут быть вставлены в раздел кодов с помощью команды меню **Insert > Program Elements [Вставить > Элементы программы]**.

Пример каталога элементов программы в AWL



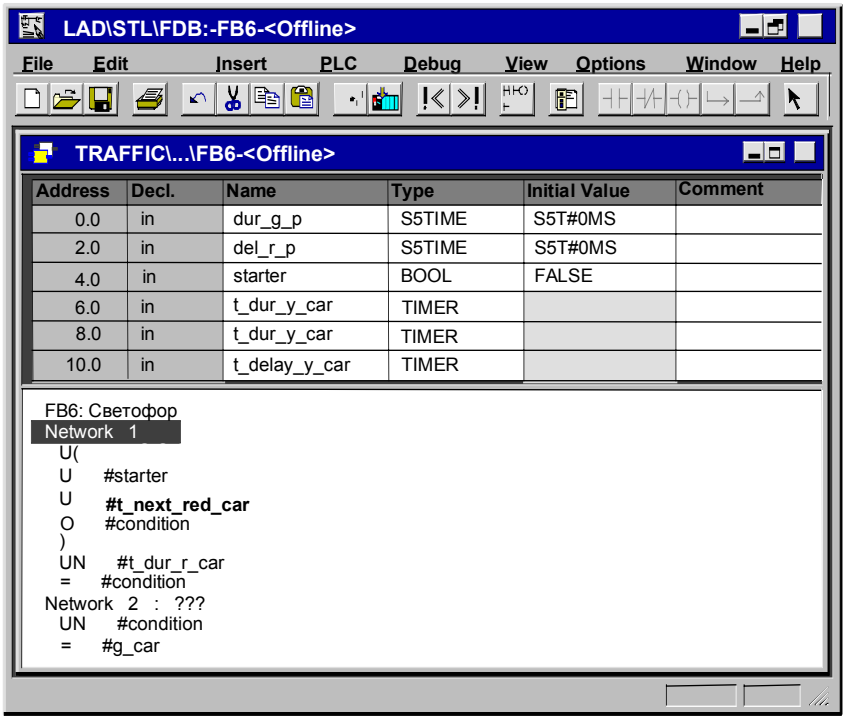
Пояснения к рисунку: File – файл; Edit – редактировать; Insert – вставить; PLC – ПЛК; Debug – отладка; View – вид; Options – параметры; Window – окно; Help – помощь; Address – адрес; Decl(aration) – описание; Name – имя; Type – тип; Initial Value – начальное значение; Comment – комментарий

9.2 Редактирование таблицы описания переменных

9.2.1 Использование описания переменных в логических блоках

При открытии логического блока появляется окно, содержащее в верхней половине таблицу описания переменных для блока, а в нижней половине – раздел кодов, в котором редактируется текущий код блока.

Пример: Таблица описания переменных и раздел кодов в AWL



Пояснения к рисунку: File – файл; Edit – редактировать; Insert – вставить; PLC – ПЛК; Debug – отладка; View – вид; Options – параметры; Window – окно; Help – помощь; Address – адрес; Decl(aration) – описание; Name – имя; Type – тип; Initial Value – начальное значение; Comment – комментарий

В таблице описания переменных определяются локальные переменные, относящиеся к блоку, включая формальные параметры блока и системные атрибуты для параметров. Результатом этого является следующее:

- При описании в стеке локальных данных резервируется достаточно памяти для временных переменных, а в случае функциональных блоков – для статических переменных в экземплярном DB, присоединяемом позднее.
- При установке входных, выходных и проходных (in/out) параметров, вы также определяете "интерфейс" для вызова блока в программе.
- При описании переменных в функциональном блоке эти переменные (за исключением временных переменных) определяют также структуру данных для каждого экземплярного блока данных, связанного с этим функциональным блоком.
- Установкой системных атрибутов вы назначаете специальные свойства параметрам для сообщений и конфигурации соединений, функций взаимодействия с оператором и конфигурации управления процессом.

9.2.2 Связь между таблицей описания переменных и разделом кодов

Таблица описания переменных и раздел кодов логических блоков тесно связаны друг с другом, так как имена из таблицы описания переменных используются в разделе кодов. Поэтому любые изменения в описании переменных влияют на весь раздел кодов.

Действие в описании переменной	Реакция в разделе кодов
Правильный новый ввод	Если имеет место недопустимый код, ранее не описанная переменная теперь становится действительной
Правильное изменение имени без изменения типа	Символ немедленно отображается всюду со своим новым именем
Правильное имя заменяется недопустимым именем	Код остается неизменным
Недопустимое имя заменяется правильным именем	Если имеет место недопустимый код, он становится допустимым
Изменение типа	Если имеет место недопустимый код, он становится допустимым, а если код допустимый, то он может стать недопустимым
Удаление переменной (символического имени), используемой в коде	Правильный код становится недопустимым

Изменения в комментариях, неправильный ввод новой переменной, изменение начального значения или удаление неиспользуемой переменной не оказывают влияния на раздел кодов.

9.2.3 Структура таблицы описания переменных

Таблица описания переменных содержит поля для ввода адресов, типа описания, символического имени, типа данных, начального значения и комментария к переменным. Каждая строка таблицы предназначена для описания одной переменной. Переменные типа ARRAY [массив] или STRUCT [структура] требуют нескольких строк.

Правильные типы данных для локальных данных блоков различных типов вы найдете в приложении "Назначение типов данных локальным данным логических блоков".

Столбец	Значение	Примечания	Редактирование
Address [Адрес]	Адрес в формате БАЙТ.БИТ	У типов данных, для которых требуется более одного байта, адрес показывает распределение памяти переходом к следующему байтовому адресу. Обозначения: * : размер элемента массива в байтах, + : начальный адрес элемента относительно начала структуры (STRUCT), = : полная потребность структуры (STRUCT) в памяти.	Системный ввод: адрес назначается системой и отображается при прекращении ввода описания.
Variable [Переменная]	Символическое имя переменной	Символ переменной должен начинаться с буквы. Зарезервированные ключевые слова не допускаются.	Необходимо
Declaration [Описание]	Тип описания, "назначение" переменной	В зависимости от типа блока возможны следующие варианты: входные параметры "in" выходные параметры "out" проходные параметры "in_out" статические переменные "stat" временные переменные "temp"	Назначаются системой в зависимости от типа блока
Data type [Тип данных]	Тип данных переменной (BOOL, INT, WORD, ARRAY и т. д.)	Вы можете выбирать элементарные типы данных с помощью меню, всплывающего при нажатии правой кнопки мыши.	Необходимо
Initial value [Начальное значение]	Начальное значение, если программное обеспечение не должно использовать значение по умолчанию.	Должно быть совместимо с типом данных. Когда вы сохраняете блок данных в первый раз, начальное значение используется как фактическое значение, если вы явно не определили фактические значения для переменных.	Не обязательно
Comment [Комментарий]	Комментарий для документирования		Не обязательно

Умолчание

При открытии вновь созданного блока открывается таблица описания переменных по умолчанию. В ней перечисляются в установленном порядке только типы описаний, действительные для выбранного типа блока (in, out, in_out, stat, temp). Когда вы создаете новый организационный блок (OB), отображается стандартное описание переменных, значения которых вы можете изменить.

Столбцы, не подлежащие редактированию в таблице описаний

Столбец	Запись
Адрес	Адрес назначается системой и отображается при завершении ввода описания.
Тип описания	Тип описания определяется положением описания внутри таблицы. Это гарантирует, что переменные могут вводиться только в правильном порядке в соответствии с типами описаний. Если вы хотите изменить тип описания, вырежьте это описание и вставьте его снова под новым типом описаний.

9.2.4 Общие замечания о таблице описания переменных

Для редактирования таблицы вы можете использовать обычные функции из меню **Edit [Редактировать]**. Для облегчения редактирования используйте контекстно-чувствительное меню, появляющееся при нажатии правой клавиши мыши. При вводе типа данных вы тоже можете воспользоваться поддержкой правой клавиши мыши.

Выбор в таблицах описания переменных

Отдельные строки выбираются щелчком на соответствующей защищенной от записи ячейке с адресом. Вы можете выбрать несколько строк, относящихся к одному и тому же типу описания, удерживая нажатой клавишу SHIFT. Выбранные строки выделяются черным цветом.

Массивы (ARRAY) выбираются щелчком на ячейке с адресом соответствующей строки.

Если вы хотите **выбрать структуру**, щелкните мышкой на ячейке с адресом первой или последней строки (в которой расположено ключевое слово STRUCT или END_STRUCT). Отдельные описания внутри структуры выбираются щелчком мыши на соответствующей ячейке с адресом для строки.

При вводе структур внутри другой структуры эта иерархия отображается соответствующим сдвигом вправо имен переменных.

Отмена действий

В таблице описания переменных для отмены самой последней операции вырезания или удаления вы можете использовать команду меню **Edit > Undo [Редактировать > Отменить]**.

9.3 Мультиэкземпляры в таблице описания переменных

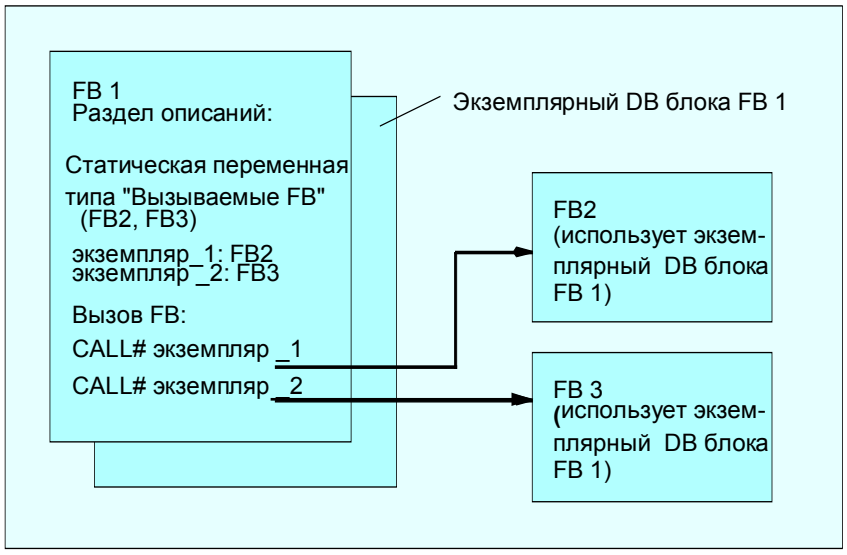
9.3.1 Использование мультиэкземпляров

Возможно, что вы захотите или будете вынуждены использовать ограниченное количество экземплярных блоков данных из-за эксплуатационных характеристик (например, емкости памяти) используемого вами CPU S7. Если другие существующие функциональные блоки вызываются в некотором FB в вашей пользовательской программе (иерархия вызовов FB), то вы можете вызывать эти другие функциональные блоки без их собственных (дополнительных) экземплярных блоков данных.

Воспользуйтесь следующим решением:

- Включите функциональные блоки, которые вы хотите вызвать, в описание переменных вызывающего функционального блока в качестве статических переменных.
- В этом функциональном блоке вызывайте другие функциональные блоки без их собственных (дополнительных) экземплярных блоков данных.
- Это сосредоточивает экземплярные данные в одном экземплярном блоке данных, т. е. вы можете использовать имеющееся в вашем распоряжении количество блоков данных более эффективно.

Следующий пример иллюстрирует описанное решение: FB2 и FB3 используют экземплярный DB функционального блока FB1, из которого они вызываются.



Единственное требование: Вы должны "сказать" вызывающему функциональному блоку, какие экземпляры вы вызываете и какого типа (FB) эти экземпляры. Эти данные должны быть внесены в раздел описаний вызывающего функционального блока. Используемый функциональный блок должен иметь по крайней мере одну переменную или параметр из области данных (VAR_TEMP использоваться не могут).

Не используйте мультиэкземплярные блоки данных, если при работе CPU предполагается производить изменения в режиме online. Беспроблемная перезагрузка гарантируется только при использовании экземплярных блоков данных.

9.3.2 Правила описания мультиэкземпляров

Для описания мультиэкземпляров применяются следующие правила:

- Описание мультиэкземпляров возможно только в функциональных блоках, созданных с помощью STEP 7 версии 2 и выше (см. Block Attribute [Атрибут блока] в свойствах функционального блока).
- Чтобы описать мультиэкземпляры, функциональный блок должен быть создан как блок, способный работать с мультиэкземплярами (установка по умолчанию, начиная со STEP 7 версии x.x; может быть деактивирована в редакторе с помощью **Options > Customize [Параметры > Настройка]**).
- Функциональному блоку, в котором описан мультиэкземпляр, должен быть поставлен в соответствие экземплярный блок данных.
- Мультиэкземпляр может быть описан только как статическая переменная (тип описания "stat").

Замечание

Вы можете создавать мультиэкземпляры и для системных функциональных блоков.

Если функциональный блок не был создан как блок, способный иметь мультиэкземпляры, а вы хотите, чтобы он имел это свойство, вы можете сгенерировать из этого функционального блока исходный файл, в котором вы затем удаляете свойство блока CODE_VERSION1, после чего компилируете функциональный блок снова.

9.4 Общие замечания по редактированию команд и комментариев

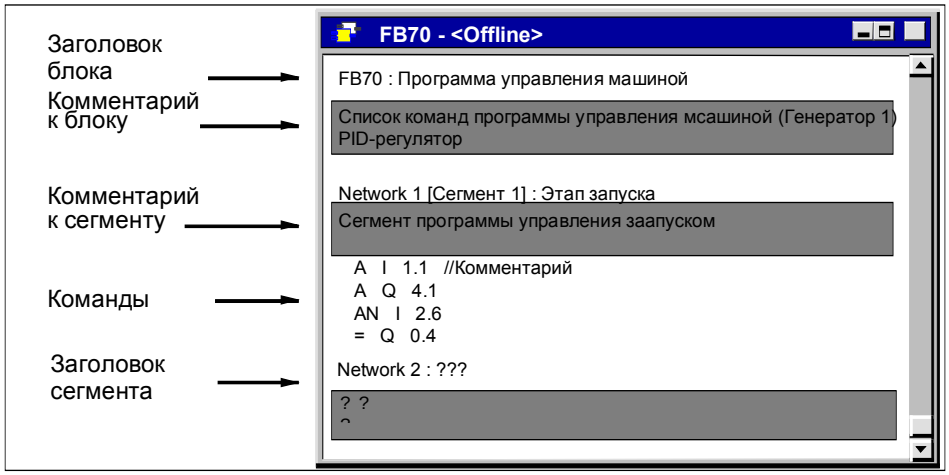
9.4.1 Структура раздела кодов

В разделе кодов вы программируете последовательность операций для своего логического блока путем ввода соответствующих команд в сегментах в зависимости от выбранного языка программирования. После ввода команды редактор немедленно выполняет проверку синтаксиса и отображает ошибку красным курсивом.

Раздел кодов логического блока обычно включает в себя ряд сегментов, которые составлены в виде списка команд.

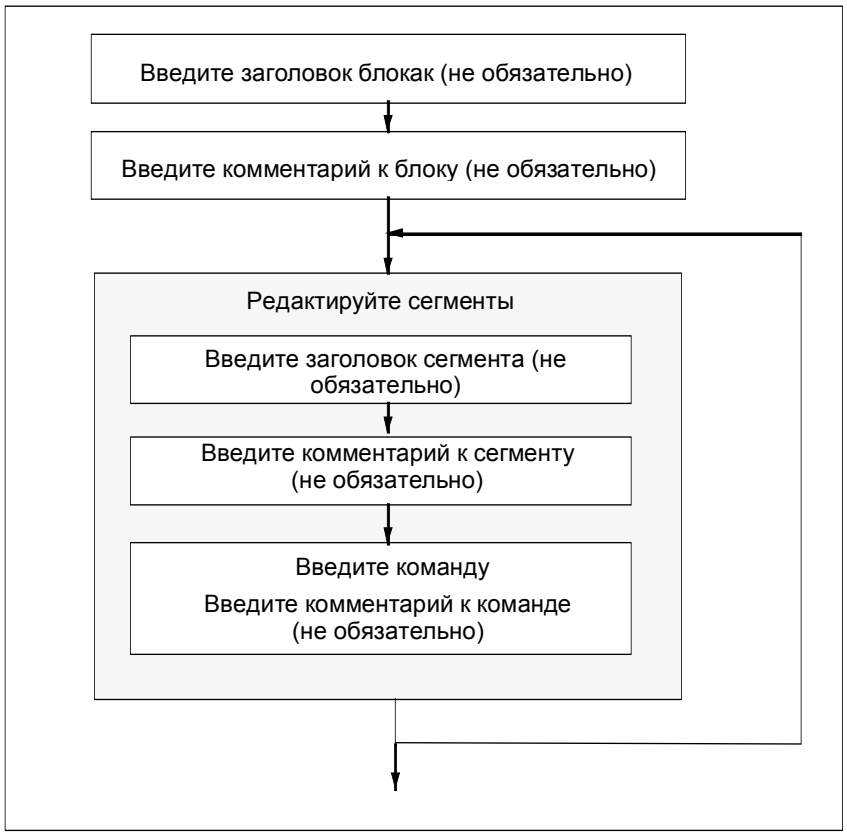
В разделе кодов вы можете редактировать заголовок блока, комментарии к блоку, заголовок сегмента, комментарии к сегменту и строки команд внутри сегментов.

Структура раздела кодов на примере языка программирования AWL



9.4.2 Процедура ввода команд

Отдельные части раздела кодов вы можете редактировать в любом порядке. Мы рекомендуем вам действовать следующим образом, когда вы программируете блок в первый раз:



Вы можете производить изменения в режиме вставки или замены. Переключение между этими режимами производится с помощью клавиши INSERT.

9.4.3 Ввод в программу совместно используемых символов

Вы можете вставлять символы в раздел кодов своей программы с помощью команды меню **Insert > Symbol [Вставить > Символ]**. Если курсор находится в начале, в конце или в середине строки, то тем самым уже выбран символ, который запускается вместе с этой строкой – если он существует. Если вы изменяете строку, то выбор в списке обновляется.

Разделительными знаками для начала и конца строки являются, например, пробел, точка, двоеточие. Внутри совместно используемых символов разделительные знаки не интерпретируются.

Для ввода символов действуйте следующим образом:

1. Введите первую букву желаемого символа в программе.
2. Одновременно нажмите CTRL и J, чтобы отобразить список символов. Первый символ, начинающийся с введенной вами буквы, уже выделен.
3. Введите этот символ, нажав RETURN, или выберите другой символ.

После этого вместо первой буквы вводится этот символ, заключенный в кавычки.

В общем случае происходит следующее: если курсор расположен в начале, в конце или внутри строки, то при вводе символа эта строка заменяется символом, заключенным в кавычки.

9.4.4 Заголовок и комментарии к блокам и сегментам

Благодаря комментариям ваша программа легче читается, что облегчает прием в эксплуатацию и повышает эффективность поиска ошибок. Они являются важной частью программной документации и, конечно, должны использоваться.

Комментарии в программах, представленных в виде контактного плана, функционального плана и списка команд

Доступны следующие комментарии:

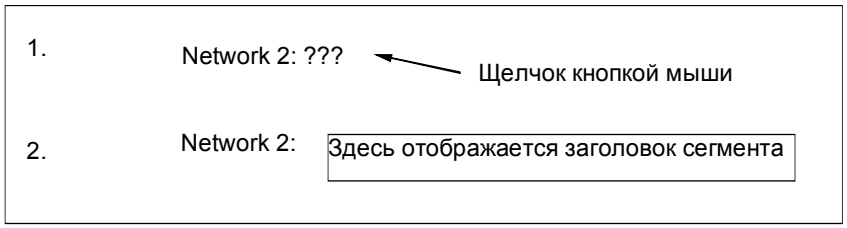
- Заголовок блока (не более 64 знаков)
- Комментарий к блоку: документирует весь логический блок, например, назначение блока
- Заголовок сегмента: (не более 64 знаков)
- Комментарий к сегменту: документирует функции отдельного сегмента
- Столбец комментариев в таблице описания переменных: комментирует описанные локальные данные
- Комментарий к символу: комментарии, введенные для операнда при определении символического имени в таблице символов.
  Вы можете отобразить эти комментарии с помощью команды меню **View > Display > Symbol Information [Вид > Отобразить > Информация о символах]**.

В разделе кодов логического блока вы можете вводить заголовок блока, заголовки сегментов, комментарии к блоку и комментарии к сегментам.

Заголовок блока или заголовок сегмента

Для ввода заголовка блока или сегмента поместите курсор на трех вопросительных знаках справа от слова Block [блок] или Network [сегмент] (например, Network 1 : ???). Открывается текстовое окно, в котором вы можете ввести заголовок. Он может иметь длину до 64 символов.

Комментарии к блоку относятся ко всему логическому блоку. Они могут комментировать функциональное назначение блока. Комментарии к сегментам относятся к отдельным сегментам и документируют подробности, касающиеся этого сегмента.



Комментарии к блоку и комментарии к сегменту

Вы можете включать и выключать серые поля комментариев с помощью команды меню **View > Display > Comments [Вид > Отобразить > Комментарии]**. Двойной щелчок на поле комментария открывает текстовое окно, в котором вы можете теперь вводить примечания. Для комментариев к блоку и комментариев к сегментам вам отводится 64 Кбайта на блок.

9.4.5 Функция поиска ошибок в разделе кодов

Ошибки в разделе кодов легко распознаются по их красному цвету. Чтобы облегчить поиск ошибок, находящихся вне видимого поля экрана, редактор предлагает две функции поиска **Edit > Go To > Previous Error/Next Error [Редактировать > Перейти к > Предыдущей ошибке/Следующей ошибке]**.

Поиск ошибок не ограничивается одним сегментом. Это значит, что при поиске просматривается весь раздел кодов, а не только один сегмент или область, в данный момент видимая на экране.

Если вы активизируете строку состояния с помощью команды меню **View > Status Bar [Вид > Строка состояния]**, то там отображаются примечания к найденным ошибкам.

Вы можете также исправлять ошибки и вносить изменения в режиме замены. Переключение между режимами вставки и замены производится с помощью клавиши INSERT.

9.5 Редактирование команд КОР в разделе кодов

9.5.1 Настройки для программирования контактного плана

Установка формата контактного плана

Вы можете установить формат для создания программ в виде контактного плана. Выбираемый вами формат (A4 книжная ориентация/альбомная ориентация/максимальный размер) оказывает влияние на количество элементов контактного плана, которые могут быть отображены в одной цепи.

1. Выберите команду меню Options > Customize [Параметры > Настройка].
2. В появившемся диалоговом окне выберите закладку "LAD/FBD (или KOP/FUP)".
3. Выберите требуемый формат из окна списка "Layout [Размещение]". Введите требуемый размер формата.

Настройки для печати

Если вы хотите распечатать раздел кодов контактного плана, вы должны установить подходящий размер страницы, прежде чем вы начнете программировать раздел кодов.

Настройки в закладке "LAD/FBD (KOP/FUP)"

В закладке "LAD/FBD (KOP/FUP)", куда вы попадаете с помощью команды меню **Options > Customize [Параметры > Настройка]**, вы можете выполнять следующие основные настройки:

- Layout [Размещение]: Оно определяет размер отображения ваших сегментов. В зависимости от размера, который вы выбираете, вы можете разместить различные количества элементов контактного плана друг за другом в одном сегменте. Эта настройка имеет также значение при распечатке вашего блока.
- Width of address field [Ширина адресного поля]: Здесь устанавливается ширина текстового окна для адресов (символических или абсолютных). Если ширина адреса превышена, то вставляется символ перевода строки (возврат каретки). В случае символической адресации имеет смысл выбирать адресное поле большей величины, а в случае абсолютной адресации достаточно поля меньших размеров.
- Element representation [Представление элементов]: Вы можете выбирать между двухмерным и трехмерным отображением элементов контактного плана.
- Line / Color [Линия / Цвет]: Для выбранного элемента, контакта, статус выполняется, статус не выполняется.

9.5.2 Правила ввода элементов контактного плана

Описание представления языка программирования в виде контактного плана вы найдете в руководстве "КОР для S7-300/400 – Программирование блоков" или в оперативной помощи к контактному плану.

Сегмент контактного плана может состоять из ряда элементов, расположенных в нескольких ветвях. Все элементы и ветви должны быть соединены; левая шина не считается соединением(IEC 1131–3).

При программировании в контактном плане вы должны соблюдать ряд руководящих указаний. Сообщения об ошибках проинформируют вас о любых сделанных вами ошибках.

Закрытие сегмента контактного плана

Каждый сегмент контактного плана должен быть закрыт с помощью катушки или блока. Для закрытия сегмента не должны использоваться следующие элементы контактного плана:

- блоки сравнения
- катушки для промежуточных выводов _/(#)_/
- катушки для анализа положительного _/(P)_/ или отрицательного _/(N)_/ фронта

Размещение блоков

Начальной точкой ветви для подключения блока всегда должна быть левая шина. В ветви перед блоком могут находиться логические операции или другие блоки.

Размещение катушек (coils)

Катушки размещаются автоматически на правом конце сегмента, образуя конец ветви.

Исключения: Катушки для промежуточных выводов _/(#)_/ и для анализа положительного _/(P)_/ или отрицательного _/(N)_/ фронта не могут размещаться ни на левом, ни на правом краю ветви. Не разрешаются они и в параллельных ветвях.

Некоторые катушки требуют булевой логической операции, а некоторые катушки не должны иметь булевой логической операции.

Катушки, требующие булевой логики:

- выход _/( ), установка выхода _/(S), сброс выхода _/(R)
- промежуточный выход _/(#)_/, положительный фронт _/(P)_/, отрицательный фронт _/(N)_/
- все счетчики и таймеры
- переход по отрицанию _/(JMPN)
- включение главного управляющего реле _/(MCR<)
- сохранение VKE (RLO) в бите BR _/(SAVE)
- возврат _/(RET)

Катушки, не допускающие булевой логики:

- активизация главного управляющего реле _/(MCRA)
- деактивизация главного управляющего реле _/(MCRD)
- открытие блока данных _/(OPN)
- выключение главного управляющего реле _/(MCR>)

Все остальные катушки могут как иметь булеву логику, так и не иметь ее.

Следующие катушки **не должны использоваться как параллельные выходы**:

- переход по отрицанию _/(JMPN)
- переход _/(JMP)
- вызов из катушки _/(CALL)
- возврат _/(RET)

Деблокирующий вход/Деблокирующий выход

Деблокирующий вход "EN" и деблокирующий выход "ENO" блоков может быть подключен, но это не обязательно.

Удаление и замена

Если ветвь состоит только из одного элемента, то при удалении этого элемента удаляется и вся ветвь.

При удалении блока удаляются также все ветви, подключенные к булевым входам блока, за исключением главной ветви.

Режим замены может использоваться для простой замены элементов одного и того же типа.

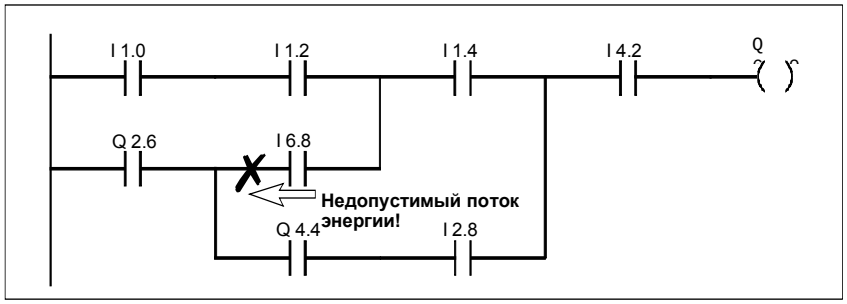
Параллельные ветви

- Чертите параллельные ветви слева направо.
- Параллельные ветви открываются вниз и закрываются вверх.
- Параллельная ветвь всегда открывается после выделенного элемента контактного плана.
- Параллельная ветвь всегда закрывается после выделенного элемента контактного плана.
- Для удаления параллельной ветви удалите все элементы в этой ветви. Когда в ветви удаляется последний элемент, ветвь удаляется автоматически.

9.5.3 Недопустимые логические операции в контактном плане

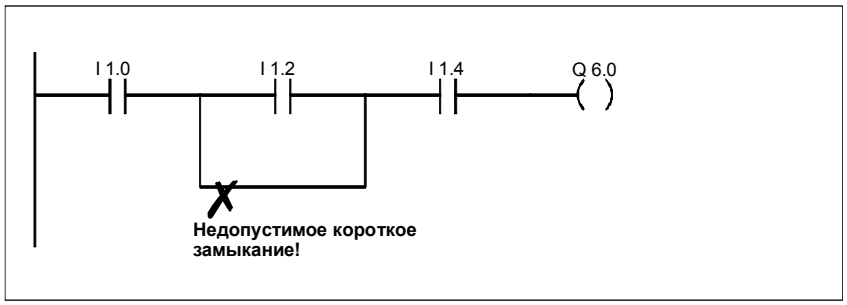
Поток энергии справа налево

Нельзя создавать ветви, которые могут вызвать поток энергии в противоположном направлении. Пример показан на следующем рисунке: при нулевом состоянии сигнала на I 1.4 поток энергии через I 6.8 был бы направлен справа налево, что недопустимо.



Короткое замыкание

Не могут создаваться ветви, вызывающие короткое замыкание. Пример показан на следующем рисунке:



9.6 Редактирование команд FUP в разделе кодов

9.6.1 Настройки для программирования функционального плана

Установка формата функционального плана

Вы можете установить формат для создания программ в виде функционального плана. Выбираемый вами формат (A4 книжная ориентация/альбомная ориентация/максимальный размер) оказывает влияние на количество элементов функционального плана, которые могут быть отображены в одной цепи.

1. Выберите команду меню Options > Customize [Параметры > Настройка].
2. В появившемся диалоговом окне выберите закладку "LAD/FBD (или KOP/FUP)".
3. Выберите требуемый формат из окна списка "Layout [Размещение]". Введите требуемый размер формата.

Настройки для печати

Если вы хотите распечатать раздел кодов функционального плана, вы должны установить подходящий размер страницы, прежде чем вы начнете программировать раздел кодов.

Настройки в закладке "LAD/FBD (KOP/FUP)"

В закладке "LAD/FBD (KOP/FUP)", куда вы попадаете с помощью команды меню **Options > Customize [Параметры > Настройка]**, вы можете выполнять следующие основные настройки:

- Layout [Размещение]: Оно определяет размер отображения ваших сегментов. В зависимости от размера, который вы выбираете, вы можете разместить различные количества элементов функционального плана друг за другом в одном сегменте. Эта настройка имеет также значение при распечатке вашего блока.
- Width of address field[Ширина адресного поля]: Здесь устанавливается ширина текстового окна для адресов (символических или абсолютных). Если ширина адреса превышена, то вставляется символ перевода строки (возврат каретки). В случае символической адресации имеет смысл выбирать адресное поле большей величины, а в случае абсолютной адресации достаточно поля меньших размеров.
- Element representation[Представление элементов]: Вы можете выбирать между двухмерным и трехмерным отображением элементов функционального плана.
- Line / Color[Линия / Цвет]: Для выбранного элемента, контакта, статус выполняется, статус не выполняется.

9.6.2 Правила ввода элементов функционального плана

Описание представления языка программирования в виде функционального плана вы найдете в руководстве "FUP для S7-300/400 – Программирование блоков" или в оперативной помощи к функциональному плану.

Сегмент функционального плана может состоять из ряда элементов. Все элементы должны быть соединены (IEC 1131–3).

При программировании в FUP вы должны соблюдать ряд руководящих указаний. Сообщения об ошибках проинформируют вас о любых сделанных вами ошибках.

Ввод и редактирование адресов и параметров

Когда вставляется элемент FUP, то в качестве маркеров для адресов и параметров используются символы ??? и … .

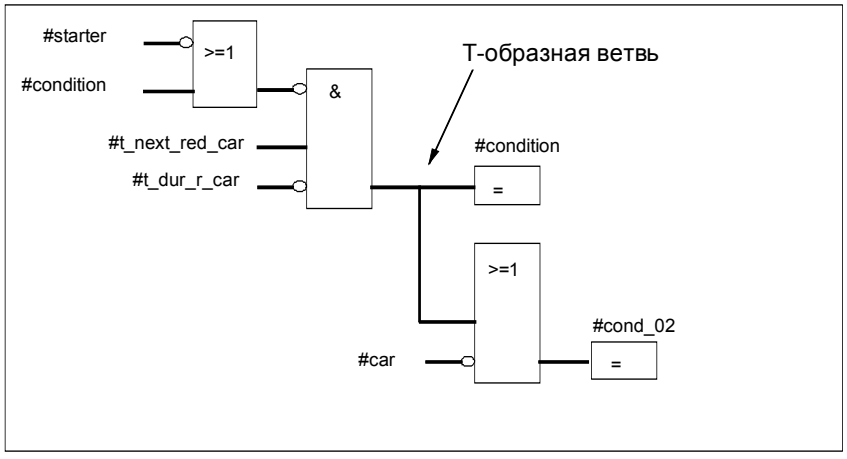
- Красные символы ??? стоят вместо адресов и параметров, которые должны быть подключены.
- Черные символы ... стоят вместо адресов и параметров, которые могут быть подключены.

Если вы поместите указатель мыши на маркерах, то отобразится ожидаемый тип данных.

Размещение блоков

Стандартные блоки (триггеры, счетчики, таймеры, математические операции и т. д.) могут быть добавлены к блокам с двоичными логическими операциями (&, >=1, XOR). Исключением из этого правила являются блоки сравнения.

В сегменте не могут быть запрограммированы отдельные логические операции с отдельными выходами. Вы можете, однако, назначить несколько присваиваний последовательности логических операций с помощью T-образной ветви. На следующем рисунке показан сегмент с двумя присваиваниями.



На правом конце логической цепочки могут быть размещены только следующие блоки, замыкающие эту цепочку:

- установка значения счетчика
- назначение параметров и прямой счет, назначение параметров и обратный счет
- назначение параметров и запуск импульсного таймера, назначение параметров и запуск таймера с удлиненным импульсом
- назначение параметров и запуск таймера с задержкой включения/выключения

Некоторые блоки требуют булевой логической операции, а некоторые блоки не должны иметь булевой логической операции.

Блоки, требующие булевой логики:

- выход, установка выхода, сброс выхода _/[R]
- промежуточный выход _/[#]_/, положительный фронт _/[P]_/, отрицательный фронт _/[N]_/
- все блоки счетчиков и таймеров
- переход по отрицанию _/[JMPN]
- включение главного управляющего реле _/[MCR<]
- сохранение VKE (RLO) в бите BR _/[SAVE]
- возврат _/[RET]

Блоки, не допускающие булевой логики:

- активизация главного управляющего реле [MCRA]
- деактивизация главного управляющего реле [MCRD]
- открытие блока данных [OPN]
- выключение главного управляющего реле [MCR>]

Все остальные блоки могут как иметь булевы логические операции, так и не иметь их.

Деблокирующий вход/Деблокирующий выход

Деблокирующий вход "EN" и деблокирующий выход "ENO" блоков может быть подключен, но это не обязательно.

Удаление и замена

При удалении блока удаляются также все ветви, подключенные к булевым входам блока, за исключением главной ветви.

Режим замены может использоваться для простой замены элементов одного и того же типа.

9.7 Редактирование команд AWL в разделе кодов

9.7.1 Настройки для программирования списка команд

Установка мнемоники

В вашем распоряжении для выбора имеются два вида мнемоник:

- SIMATIC или
- международная.

Мнемоника устанавливается в SIMATIC Manager командой меню **Options > Customize [Параметры > Настройка]** в закладке "Language [Язык]" до открытия блока. При редактировании блока мнемонику изменить нельзя.

Свойства блоков редактируются в их собственном диалоговом окне.

В редакторе вы можете открыть несколько блоков и редактировать их по очереди по мере надобности.

9.7.2 Правила ввода команд AWL

Описание представления языка программирования в виде списка команд вы найдете в руководстве "AWL для S7-300/400 – Программирование блоков" или в оперативной помощи по AWL (Language Descriptions [описания языков]).

При вводе операторов AWL в пошаговом режиме необходимо соблюдать следующие основные требования:

- Важен порядок, в котором программируются блоки. Вызываемые блоки должны программироваться раньше, чем вызывающие.
- Оператор состоит из метки (не обязательна), команды, операнда (адреса) и комментария (не обязателен).
  Пример: M001: A I 1.0 //Комментарий
- Каждый оператор находится в своей собственной строке.
- В блок можно ввести до 999 сегментов.
- Каждый сегмент может иметь примерно до 2000 строк. Если вы распахиваете или, наоборот, сжимаете изображение, то вы можете соответственно больше или меньше строк.
- При вводе команд и абсолютных адресов нет разницы между верхним и нижним регистром.

9.8 Корректировка вызовов блока

9.8.1 Корректировка вызовов блоков

Вы можете использовать команду меню **Edit > Call > Update [Редактировать > Вызов > Корректировать]** в окне "LAD/STL/FBD – Programming S7 Blocks [KOP/AWL/FUP – Программирование блоков S7]" для автоматической корректировки вызовов блоков или типов данных, определенных пользователем, которые стали недопустимыми после выполнения следующих изменений интерфейса:

- Вставка новых параметров
- Удаление параметров
- Изменение имен параметров
- Изменение типа параметров
- Изменение порядка параметров.

При назначении формальных и фактических параметров вы должны следовать следующим правилам в указанном порядке:

1. **Те же имена параметров:**
   Фактические параметры назначаются автоматически, если имя формального параметра осталось тем же самым.
   Особый случай: В контактном и функциональном плане предшествующая связь для параметров двоичного входа может быть назначена автоматически только тогда, когда тип данных (BOOL) остается тем же самым. Если тип данных изменился, то предшествующая связь сохраняется в виде открытой ветви.
2. **Те же типы данных параметров:**
   После того как были назначены параметры с тем же именем, еще не назначенные фактические параметры назначаются формальным параметрам с тем же типом данных как "старым" формальным параметрам.
3. **То же самое расположение параметров:**
   После того как вы выполнили правила 1 и 2, любые фактические параметры, которые еще не были назначены, теперь назначаются формальным параметрам в соответствии с их расположением в "старом" интерфейсе.
4. Если фактические параметры не могут быть назначены с использованием трех вышеописанных правил, то они удаляются или, в случае двоичных предшествующих связей в контактном или функциональном плане, они сохраняются в виде открытых ветвей.

После выполнения этих действий проверьте сделанные вами изменения в таблице описания переменных и в разделе кодов программы.

9.9 Сохранение логических блоков

9.9.1 Сохранение логических блоков

Чтобы ввести вновь созданные блоки или изменения в разделе кодов логических блоков или в таблицах описаний в базу данных устройства программирования, вы должны сохранить соответствующий блок. После этого данные записываются на жесткий диск устройства программирования.

Для сохранения блоков на жестком диске устройства программирования:

1. Активизируйте рабочее окно блока, который вы хотите сохранить.
2. Выберите одну из следующих команд меню:
   - **File > Save [Файл > Сохранить]** сохраняет блок под тем же именем.
   - **File > Save As [Файл > Сохранить как...]** сохраняет блок под в другой программе пользователя S7 или под другим именем. Введите новый путь или новое имя блока в появившемся диалоговом окне.

В обоих случаях блок сохраняется, если его синтаксис не содержит ошибок. Синтаксические ошибки обнаруживаются немедленно при создании блока, а затем отображаются на экране красным цветом. Эти ошибки должны быть исправлены до того, как блок может быть сохранен.

Замечание

- Вы можете также сохранять блоки или исходные файлы под другими проектами или библиотеками в SIMATIC Manager (например, с помощью буксировки).
- Вы можете сохранять только блоки или полные программы пользователя на плате памяти в SIMATIC Manager.
- Если при сохранении или компиляции больших блоков возникают проблемы, то вам следует реорганизовать проект. Чтобы сделать это, используйте команду меню **File > Reorganize [Файл > Реорганизовать]** в SIMATIC Manager. Затем попытайтесь сохранить или скомпилировать снова.

9.9.2 Корректировка интерфейсов в функции, функциональном блоке или UDT

Если вам нужно исправить интерфейс в FB, FC или UDT, то во избежание конфликта меток времени действуйте следующим образом:

1. Сгенерируйте исходный файл на AWL из блока, который вы хотите изменить, и всех прямо или косвенно связанных с ним блоков.
2. Сохраните изменения в исходном файле, который вы сгенерировали.
3. Скомпилируйте измененный исходный файл обратно в блоки.

Вы теперь можете сохранить/загрузить изменения интерфейсов.

9.9.3 Как избежать ошибок при вызове блоков?

STEP 7 заменяет данные в регистре DB

При выполнении различных команд STEP 7 изменяет регистры CPU S7-300/S7-400. Например, содержимое регистров DB и DI меняется местами, когда вы вызывает FB. Это позволяет открывать экземплярный DB вызванного FB без потери адреса предыдущего экземплярного DB.

Если вы работаете с абсолютной адресацией, то могут произойти ошибки при обращении к данным, сохраненным в регистрах. В некоторых случаях заменяются адреса в регистре AR1 (адресный регистр 1) и в регистре DB. Это значит, что вы можете произвести чтение или запись по неверным адресам.

Опасность
Имеется опасность нанесения ущерба собственности и людям:

1. при использовании вызовов FC, FB и мультиэкземпляров
2. при обращении к DB с использованием полных абсолютных адресов (например, DB20.DBW10)
3. при обращении к переменным составного типа данных.

Возможно изменение содержимого регистров блоков данных (DB и DI), адресных регистров (AR1, AR2) и аккумуляторов (ACCU1, ACCU2).

Кроме того, вы не можете использовать бит VKE (RLO) слова состояния как дополнительный (неявный) параметр при вызове FB или FC.

При использовании упомянутых выше методов программирования вы должны убедиться, что вы сохранили и восстановили это содержимое сами; иначе могут возникнуть ошибки.

Сохранение правильных данных

Содержимое регистра DB может вызвать критические ситуации, если вы обращаетесь к абсолютным адресам данных, используя сокращенный формат. Например, если вы подразумеваете, что открыт DB20 (и что его номер сохранен в регистре DB), то вы можете указать DBX0.2 для обращения к данным в бите 2 байта 0 блока данных, адрес которого введен в регистр DB (иными словами, DB20). Если, однако, регистр DB содержит номер другого блока данных, то вы обратитесь к неверным данным.

Вы можете избежать ошибок при обращении к данным регистра DB, используя следующие методы адресации данных:

- Используйте символический адрес
- Используйте полный абсолютный адрес (например, *DB20.DBX0.2*)

Если вы используете эти методы адресации, то STEP 7 автоматически открывает правильный DB. Если вы используете регистр AR1 для косвенной адресации, то вы всегда должны загружать в AR1 правильный адрес.

Ситуации, в которых регистры изменяются

Манипулирование адресными регистрами для косвенной адресации имеет смысл только в AWL. Другие языки не поддерживают косвенный доступ к адресным регистрам.

Адаптация регистра DB компилятором должна приниматься в расчет во всех языках программирования, чтобы обеспечить корректную передачу параметров при вызове блоков.

Содержимое адресного регистра AR1 и регистра DB вызывающего блока переписывается в следующих ситуациях:

Ситуация	Описание
С фактическими параметрами из DB	Как только вы назначили фактический параметр блоку из DB (например, DB20.DBX0.2) STEP 7 открывает этот DB (DB20) и адаптирует содержимое регистра DB. Затем программа после вызова блока работает с адаптированным DB.
При вызове блоков вместе с данными сложных типов	После того как блок был вызван из FC, передающей вызываемому блоку компоненты формального параметра, относящегося к сложному типу данных (строка, массив, структура или UDT), содержимое регистров AR1 и DB вызывающего блока изменяется. То же относится и к вызову из FB, если параметр находится в области VAR_IN_OUT вызывающего блока.
При обращении к компонентам сложных типов данных	Когда FB обращается к компоненте формального параметра, относящегося к сложному типу данных, в области VAR_IN_OUT (строка, массив, структура или UDT), STEP 7 использует адресный регистр AR1 и регистр DB. Это значит, что содержимое обоих регистров изменяется. Когда FC обращается к компоненте формального параметра, относящегося к сложному типу данных, в области VAR_IN_OUT (строка, массив, структура или UDT), STEP 7 использует адресный регистр AR1 и регистр DB. Это значит, что содержимое обоих регистров изменяется.

Замечание

Когда FB вызывается из блока версии 1, фактический параметр для первого булева параметра IN или IN_OUT передается неправильно, если команда, предшествующая вызову, не ограничивает VKE (RLO). В этом случае он логически комбинируется с существующим VKE (RLO).

Когда вызывается FB (одиночный экземпляр или мультиэкземпляр), то производится запись в адресный регистр AR2.

Если адресный регистр AR2 изменяется в FB, то нет гарантии, что этот FB будет исполнен правильно.

10 Создание блоков данных

10.1 Основная информация о создании блоков данных

Блок данных (DB) – это блок, в котором вы можете, например, хранить значения, необходимые для доступа к вашей машине или установке. В отличие от логического блока, который программируется с помощью одного из языков программирования – контактного плана, функционального плана или списка команд, блок данных содержит только раздел описания переменных. Это значит, что раздел кодов здесь неуместен, то же относится и к программированию сегментов.

Когда вы открываете блок данных, то вы можете просматривать его описание или данные. Вы можете переключаться между этими двумя видами представления с помощью команд меню **View > Declaration View [Вид > Отображение описания]** и **View > Data View [Вид > Отображение данных]**.

Отображение описания

Отображение описания используется, если вы хотите:

- просматривать или определять структуры данных совместно используемых блоков данных,
- просматривать структуры данных блоков данных, связанных с типом данных, определенным пользователем (UDT), или
- просматривать структуры данных блоков данных, связанных с функциональным блоком (FB).

Структура данных блоков данных, которые связаны с функциональным блоком или с типом данных, определенным пользователем, не может быть изменена. Чтобы изменить их, вы должны сначала изменить соответствующий FB или UDT, а затем создать новый блок данных.

Отображение данных

Отображение данных используется, если вы хотите изменять данные. Вы можете выводить на экран, вводить или изменять текущее значение каждого элемента только в режиме отображения данных. В отображении данных блоков данных элементы переменных, относящихся к составным типам данных, перечисляются по отдельности с их полными именами.

Разница между экземплярными блоками данных и совместно используемыми блоками данных

Совместно используемый блок данных не ставится в соответствие логическому блоку. Он содержит значения, необходимые для установки или машины, и может быть вызван непосредственно в любой точке программы.

Экземплярный блок данных – это блок, который непосредственно поставлен в соответствие логическому блоку, такому как функциональный блок. Он создается автоматически, при сохранении запрограммированного функционального блока. Экземплярный блок данных содержит данные, которые хранились в функциональном блоке в таблице описания переменных.

10.2 Отображение описания блоков данных

У блоков данных, не используемых совместно, отображение описания не может быть изменено.

Столбец	Объяснение
Address [Адрес]	Отображает адрес, который STEP 7 назначает переменной автоматически, когда вы заканчиваете вводить описание.
Declaration [Описание]	Этот столбец отображается только для экземплярных блоков данных. Он показывает, как описаны переменные в таблице описания переменных функционального блока: Входной параметр (IN) Выходной параметр (OUT) Проходной параметр (IN_OUT) Статические данные (STAT)
Name [Имя]	Введите здесь символическое имя, которое вы хотите назначить каждой переменной.
Type [Тип]	Введите тип данных, который вы хотите назначить переменной (BOOL, INT, WORD, ARRAY и т. д.). Переменные могут иметь элементарный тип данных, составной тип данных или тип данных, определенный пользователем.
Initial Value [Начальное значение]	Здесь вы можете ввести начальное значение, если вы не хотите, чтобы программное обеспечение использовало значение по умолчанию для введенного типа данных. Все значения должны быть совместимы с типом данных. Когда вы сохраняете блок данных в первый раз, начальное значение используется как текущее значение, если вы не определили явно текущие значения для переменных.
Comment [Комментарий]	Ввод комментария в это поле помогает документированию переменных. Комментарий может содержать до 80 символов.

10.3 Отображение данных, содержащихся в блоках данных

Отображение данных показывает текущие значения всех переменных в блоке данных. Изменять эти значения вы можете только в режиме отображения данных. Табличное представление в этом отображении одинаково для всех совместно используемых блоков данных. Для экземплярных блоков данных на экран выводится дополнительный столбец "Declaration [Описание].

Для переменных, относящихся к составным типам данных или к типам данных, определенным пользователем, элементы в отображении данных выводятся в своих собственных строках с полным символическим именем. Если элементы находятся в области IN_OUT экземплярного блока данных, то указатель в столбце "Actual Value [Текущее значение]" показывает на составной тип данных или тип данных, определенный пользователем.

В отображении данных имеются следующие столбцы:

Столбец	Объяснение
Address [Адрес]	Отображает адрес, который STEP 7 автоматически назначает переменной.
Declaration [Описание]	Этот столбец отображается только для экземплярных блоков данных. Он показывает, как описаны переменные в таблице описания переменных функционального блока: Входной параметр (IN) Выходной параметр (OUT) Проходной параметр (IN_OUT) Статические данные (STAT)
Name [Имя]	Символическое имя, назначенное переменной в таблице описания переменных. Вы не можете редактировать это поле в режиме отображения данных.
Type [Тип]	Отображает тип данных, определенный для переменной. Для совместно используемых блоков данных здесь перечислены только элементарные типы данных, так как элементы в отображении данных с составными типами данных или с типами данных, определенными пользователем, перечисляются по отдельности. Для экземплярных блоков данных типы параметров также отображаются, для проходных параметров (IN_OUT), относящихся к составным типам данных или к типам данных, определенным пользователем, указатель указывает тип данных в столбце "Actual Value [Текущее значение]".
Initial Value [Начальное значение]	Начальное значение, которое вы ввели для переменной, если вы не хотите, чтобы программное обеспечение использовало значение по умолчанию для указанного типа данных. Когда вы сохраняете блок данных в первый раз, начальное значение используется как текущее значение, если вы не определили явно текущие значения для переменных.
Actual Value [Текущее значение]	Offline: Значение, которое переменная имела, когда блок данных был открыт, или на которое вы изменили ее последний раз и сохранили (даже если вы открывали блок данных online, это отображение не корректируется). Online: Текущее значение при открытии блока данных отображается, но не обновляется автоматически. Для обновления отображения нажмите F5. Вы можете редактировать это поле, если оно не относится к проходному параметру (IN_OUT) с составным или определенным пользователем типом данных. Все значения должны быть совместимы с типом данных.
Comment [Комментарий]	Комментарий, введенный для документирования переменной. Вы не можете редактировать это поле в режиме отображения данных.

10.4 Редактирование и сохранение блоков данных

10.4.1 Ввод структуры данных совместно используемых блоков данных

Если вы открываете блок данных, не поставленный в соответствие типу данных, определенному пользователем, или функциональному блоку, то вы можете определить его структуру в режиме отображения описания блока данных. У блоков данных, которые не используются совместно, отображение описания не может быть изменено.

1. Откройте совместно используемый блок данных, т. е. блок, не связанный с UDT или FB.
2. Выведите на экран отображение описания блока данных, если это отображение уже не установлено.
3. Определите структуру, заполнив выведенную на экран таблицу в соответствии с приведенной ниже информацией.

У блоков данных, которые не используются совместно, отображение описания не может быть модифицировано.

Столбец	Объяснение
Address [Адрес]	Отображает адрес, который STEP 7 назначает переменной автоматически, когда вы заканчиваете вводить описание.
Name [Имя]	Введите здесь символическое имя, которое вы хотите назначить каждой переменной.
Type [Тип]	Введите тип данных, который вы хотите назначить переменной (BOOL, INT, WORD, ARRAY и т. д.). Переменные могут относиться к элементарному типу данных, составному типу данных или типу данных, определенному пользователем.
Initial Value [Начальное значение]	Здесь вы можете ввести начальное значение, если вы не хотите, чтобы программное обеспечение использовало значение по умолчанию для введенного типа данных. Все значения должны быть совместимы с типом данных. Когда вы сохраняете блок данных в первый раз, начальное значение используется как текущее значение, если вы не определили явно текущие значения для переменных.
Comment [Комментарий]	Ввод необязательного комментария в это поле помогает документированию переменных. Комментарий может содержать до 80 символов.

10.4.2 Ввод и отображение структуры данных блоков данных, относящихся к FB (экземплярные DB)

Ввод

Когда вы связываете блок данных с функциональным блоком (экземплярный DB), описание переменных функционального блока определяет структуру блока данных. Любые изменения могут быть сделаны только в соответствующем функциональном блоке.

1. Откройте соответствующий функциональный блок (FB).
2. Отредактируйте таблицу описания переменных функционального блока.
3. Снова создайте экземплярный блок данных.

Отображение

В отображении описания экземплярного блока данных вы можете увидеть, как были описаны переменные в функциональном блоке.

1. Откройте блок данных.
2. Выведите на экран отображение описания этого блока данных, если это отображение уже не установлено.
3. Дополнительная информация о выведенной на экран таблице приведена ниже.

У блоков данных, которые не используются совместно, отображение описания не может быть изменено.

Столбец	Объяснение
Address [Адрес]	Отображает адрес, который STEP 7 автоматически назначает переменной.
Declaration [Описание]	Этот столбец показывает, как описаны переменные в таблице описания переменных функционального блока: Входной параметр (IN) Выходной параметр (OUT) Проходной параметр (IN_OUT) Статические данные (STAT) Описанные временные локальные данные функционального блока в экземплярном блоке данных отсутствуют.
Name [Имя]	Символическое имя, назначенное переменной в таблице описания переменных функционального блока.
Type [Тип]	Отображает тип данных, назначенный в описании переменных функционального блока. Переменные могут относиться к элементарным типам данных, к составным типам данных или к типам данных, определенным пользователем. Если в функциональном блоке, для вызова которого были определены статические переменные, вызываются дополнительные функциональные блоки, то функциональный блок или системный функциональный блок (SFB) тоже может быть указан здесь как тип данных.

Столбец	Объяснение
Initial Value [Начальное значение]	Начальное значение, которое вы ввели для переменной в описании переменных функционального блока, если вы не хотите, чтобы программное обеспечение использовало значение по умолчанию. Когда вы сохраняете блок данных в первый раз, начальное значение используется как текущее значение, если вы не определили явно текущие значения для переменных.
Comment [Комментарий]	Комментарий, введенный в описание переменных функционального блока для документирования элемента данных. Вы не можете редактировать это поле.

Замечание

В блоках данных, поставленных в соответствие функциональному блоку, вы можете редактировать только текущие значения переменных. Для ввода текущих значений переменных вы должны находиться в режиме отображения данных блока данных.

10.4.3 Ввод структуры данных типов данных, определенных пользователем (UDT)

1. Откройте тип данных, определенный пользователем (UDT).
2. Выведите на экран отображение описания, если это отображение уже не установлено.
3. Определите структуру UDT, указав последовательность переменных, их тип данных и, если необходимо, начальное значение, используя информацию из нижеприведенной таблицы.
4. Ввод переменной завершается при выходе из строки путем нажатия клавиши TAB или RETURN.

Столбец	Объяснение
Address [Адрес]	Отображает адрес, который STEP 7 назначает переменной автоматически, когда вы заканчиваете вводить описание.
Name [Имя]	Введите здесь символическое имя, которое вы хотите назначить каждой переменной.
Type [Тип]	Введите тип данных, который вы хотите назначить переменной (BOOL, INT, WORD, ARRAY и т. д.). Переменные могут иметь элементарный тип данных, составной тип данных или тип данных, определенный пользователем.
Initial Value [Начальное значение]	Здесь вы можете ввести начальное значение, если вы не хотите, чтобы программное обеспечение использовало значение по умолчанию для введенного типа данных. Все значения должны быть совместимы с типом данных. Когда вы сохраняете экземпляр типа данных, определенного пользователем (или переменную, или блок данных) в первый раз, начальное значение используется как текущее значение, если вы не определили явно текущие значения для переменных.
Comment [Комментарий]	Ввод комментария в это поле помогает документированию переменных. Комментарий может содержать до 80 символов.

10.4.4 Ввод и отображение структуры данных блоков данных, относящихся к UDT

Ввод

Когда вы ставите в соответствие блок данных типу данных, определенному пользователем, структура данных определенного пользователем типа данных определяет структуру блока данных. Любые изменения могут быть сделаны только в соответствующем типе данных, определенном пользователем.

1. Откройте тип данных, определенный пользователем (UDT).
2. Отредактируйте структуру типа данных, определенного пользователем.
3. Снова создайте блок данных.

Отображение

В режиме отображения описания блока данных вы можете только увидеть, как переменные были описаны в типе данных, определенном пользователем.

1. Откройте блок данных.
2. Выведите на экран отображение описания этого блока данных, если это отображение уже не установлено.
3. Дополнительная информация о выведенной на экран таблице приведена ниже.

Отображение описания не может быть изменено. Любые изменения могут быть сделаны только в соответствующем типе данных, определенном пользователем.

Столбец	Объяснение
Address [Адрес]	Отображает адрес, который STEP 7 автоматически назначает переменной.
Name [Имя]	Символическое имя, назначенное переменной в описании переменных типа данных пользователя.
Type [Тип]	Отображает тип данных, назначенный в описании переменных типа данных, определенного пользователем. Переменные могут относиться к элементарным типам данных, к составным типам данных или к типам данных, определенным пользователем.
Initial Value [Начальное значение]	Начальное значение, которое вы ввели для переменной в типе данных, определенном пользователем, если вы не хотите, чтобы программное обеспечение использовало значение по умолчанию. Когда вы сохраняете блок данных в первый раз, начальное значение используется как текущее значение, если вы не определили явно текущие значения для переменных.
Comment [Комментарий]	Комментарий, введенный в описание переменных типа данных, определенного пользователем, для документирования элемента данных.

Замечание

В блоках данных, поставленных в соответствие типу данных, определенному пользователем, вы можете редактировать только текущие значения переменных. Для ввода текущих значений переменных вы должны находиться в режиме отображения данных блоков данных.

10.4.5 Редактирование данных в отображении данных

Редактирование текущих данных возможно только в режиме отображения данных блоков данных.

1. Если необходимо, переключитесь в табличное представление в отображении данных с помощью команды меню **View > Data View [Вид > Отображение данных]**.
2. Введите требуемые текущие значения для элементов данных в поля столбца "Actual Value [Текущее значение]". Текущие значения должны быть совместимы с типом данных элементов данных.

Любые неверные записи (например, если введенное текущее значение несовместимо с типом данных), сделанные при редактировании, немедленно распознаются и показываются красным цветом. Эти ошибки должны быть исправлены до сохранения блока данных.

Замечание

Любые изменения в значениях данных запоминаются только после сохранения блока данных.

10.4.6 Сброс данных на их начальные значения

Сброс значений данных возможен только в режиме отображения данных блоков данных.

1. Если необходимо, переключитесь в табличное представление в отображении данных с помощью команды меню **View > Data View [Вид > Отображение данных]**.
2. Чтобы сделать сброс, выберите команду меню **Edit > Initialize Data Block [Редактировать > Инициализировать блок данных]**.

Всем переменным вновь присвоены предназначенные для них начальные значения, т. е. текущие значения всех переменных заменены их соответствующими начальными значениями.

Замечание

Любые изменения в значениях данных запоминаются только после сохранения блока данных.

10.4.7 Сохранение блоков данных

Для ввода в базу данных устройства программирования вновь созданных блоков или измененных значений данных в блоках данных вы должны сохранить соответствующий блок. После этого данные записываются на жесткий диск устройства программирования.

Для сохранения блоков на жестком диске устройства программирования:

1. Активизируйте рабочее окно блока, который вы хотите сохранить.
2. Выберите одну из следующих команд меню:
   - **File > Save [Файл > Сохранить]** сохраняет блок под тем же именем.
   - **File > Save As [Файл > Сохранить как...]** сохраняет блок в другой программе пользователя S7 или под другим именем. Введите в появившемся диалоговом окне новый путь или новое имя блока. В случае блоков данных вы не можете использовать имя DB0, так как этот номер зарезервирован для системы.

В обоих случаях блок сохраняется только тогда, когда его синтаксис не содержит ошибок. Синтаксические ошибки определяются немедленно при создании блока и отображаются на экране красным цветом. Эти ошибки должны быть исправлены до того, как блок может быть сохранен.

Замечание

- Вы можете также сохранять блоки или исходные файлы под другими проектами или библиотеками в SIMATIC Manager (например, с помощью буксировки).
- Вы можете сохранять только блоки или полные программы пользователя на плате памяти в SIMATIC Manager.
- Если при сохранении или компиляции больших блоков возникают проблемы, то вам следует реорганизовать проект. Чтобы сделать это, используйте команду меню **File > Reorganize [Файл > Реорганизовать]** в SIMATIC Manager. Затем попытайтесь сохранить или скомпилировать снова.

---

11 Создание исходных файлов на AWL

11.1 Основная информация по программированию исходных файлов на AWL

Вы можете ввести свою программу или ее часть в виде исходного файла на AWL, а затем скомпилировать ее в блоки за один прием. Исходный файл может содержать код для нескольких блоков, которые затем компилируются как блоки за один проход компилятора.

Создание программ с использованием исходного файла имеет следующие преимущества:

- Вы можете создать и редактировать исходный файл с помощью любого редактора ASCII, а затем импортировать его и скомпилировать в блоки с помощью данного приложения. Процесс компиляции создает отдельные блоки и сохраняет их в программе пользователя S7.
- Вы можете программировать несколько блоков в одном исходном файле.
- Вы можете сохранить исходный файл, даже если он содержит ошибки. Это невозможно, если вы создаете логические блоки, используя пошаговый контроль синтаксиса. Однако при компилировании исходного файла синтаксические ошибки только сообщаются.

Исходный файл создается с использованием синтаксиса представления языка программирования в виде списка команд (AWL). Исходному файлу с помощью ключевых слов дается структура блоков, описаний переменных и сегментов.

При создании блоков в исходных файлах на AWL вы должны принять во внимание следующее:

- Руководящие указания по программированию исходных файлов на AWL
- Синтаксис и форматы для блоков в исходных файлах на AWL
- Структуру блоков в исходных файлах на AWL

11.2 Правила программирования исходных файлов на AWL

11.2.1 Правила ввода операторов в исходных файлах на AWL

Исходный файл на AWL состоит в основном из сплошного текста. Чтобы обеспечить компиляцию этого файла в блоки, вы должны соблюдать определенные правила относительно структуры и синтаксиса.

Следующие общие указания применяются при создании программ пользователя в виде исходных файлов на AWL:

Тема	Правило
Синтаксис	Синтаксис операторов AWL такой же, как и в редакторе пошагового ввода для списка команд. Единственным исключением является команда CALL.
CALL	В исходном файле параметры вводятся в круглых скобках. Отдельные параметры разделяются запятой. Пример: вызов FC (одна строка) CALL FC10 (param1 :=I0.0,param2 :=I0.1); Пример: вызов FB (одна строка) CALL FB10, DB100 (para1 :=I0.0,para2 :=I0.1); Пример: вызов FB (более одной строки) CALL FB10, DB100 (para1 :=I0.0, para2 :=I0.1); Замечание: При вызове блока передавайте параметры в редакторе ASCII в определенном порядке. В противном случае комментарии для этих строк могут не соответствовать друг другу в представлениях AWL и исходного файла.
Верхний/нижний регистр	Редактор в этом приложении не чувствителен к регистру, исключение составляют системные атрибуты. При вводе строк (тип данных STRING) вы тоже должны соблюдать верхний и нижний регистр. Ключевые слова показываются в верхнем регистре. При компиляции верхний и нижний регистр не соблюдаются; поэтому вы можете их вводить в верхнем или нижнем регистре или смешивать оба регистра.
Точка с запятой	Обозначайте конец любого оператора AWL и любого описания переменной точкой с запятой (;). Вы можете вводить в строке более одной команды.
Двойной слеш (//)	Начинайте каждый комментарий двойным слешем (//) и заканчивайте комментарий нажатием RETURN (или переводом строки).

11.2.2 Правила описания переменных в исходных файлах на AWL

Для каждого блока в исходном файле вы должны описать необходимые переменные.

Раздел описания переменных предшествует разделу кодов блока.

Переменные (если они используются) должны быть описаны в правильной последовательности для типов описаний. Это значит, что все переменные, относящиеся к одному типу описания, находятся вместе.

Для контактного плана, функционального плана и списка команд вы заполняете таблицу описания переменных, здесь же вы должны работать с соответствующими ключевыми словами.

Ключевые слова для описания переменных

Тип описания	Ключевые слова	Действительно для…
Входные параметры	"VAR_INPUT" Список описаний "END_VAR"	FB, FC
Входные параметры	"VAR_OUTPUT" Список описаний "END_VAR"	FB, FC
Проходные параметры	"VAR_IN_OUT" Список описаний "END_VAR"	FB, FC
Статические переменные	"VAR" Список описаний "END_VAR"	FB
Временные переменные	"VAR_TEMP" Список описаний END_VAR	OB, FB, FC

Ключевое слово END_VAR отмечает конец списка описаний.

Список описаний – это список переменных, относящихся к одному типу описания, в котором переменным могут быть присвоены значения по умолчанию (исключение: VAR_TEMP). Следующий пример показывает структуру записи в списке описаний:

Duration_Motor1	**:**	**S5TIME**	**:=**	**S5T#1H_30M ;**
Переменная		**Тип данных**		**Значение по умолчанию**

Замечание

Символ переменной должен начинаться с буквы. Вы не можете назначать для переменной символическое имя, совпадающее с одним из зарезервированных ключевых слов.

Если символы переменных одинаковы в локальных описаниях и в таблице символов, то вы можете закодировать локальные переменные, поместив перед именем #, и записывая переменные из таблицы символов в кавычках. В противном случае блок интерпретирует переменную как локальную.

11.2.3 Правила размещения блоков в исходных файлах на AWL

Вызываемые блоки предшествуют вызывающим. Это значит:

- OB1, используемый в большинстве случаев, который вызывает другие блоки, должен быть последним. Блоки, вызываемые из OB1, должны предшествовать ему.
- Типы данных, определенные пользователем (UDT), предшествуют блокам, в которых они используются.
- Блоки данных, связанные с типом данных, определенным пользователем (UDT), следуют за этим типом данных, определенным пользователем.
- Совместно используемые блоки данных предшествуют всем блокам, из которых они вызываются.
- Экземплярные блоки следуют за соответствующим функциональным блоком.
- DB0 зарезервирован. Вы не можете создавать блок данных с этим именем.

11.2.4 Правила установки системных атрибутов в исходных файлах на AWL

Системные атрибуты могут быть назначены блокам и параметрам. Они управляют конфигурацией сообщений и конфигурацией соединений, функциями взаимодействия с оператором и конфигурацией управления процессом.

При вводе системных атрибутов в исходном файле имеет силу следующее:

- Ключевые слова для системных атрибутов всегда начинаются с S7_.
- Системные атрибуты помещаются в фигурных скобках.
- Синтаксис: {S7_idenifier := 'string'}
  несколько идентификаторов разделяются точкой с запятой ";".
- Системные атрибуты для блоков находятся перед свойствами блока и после ключевых слов ORGANIZATION_ и TITLE.
- Системные атрибуты для параметров включаются в описание параметра, т. е. перед двоеточием для описания данных.
- Символы верхнего и нижнего регистра считаются различными. Это значит, что важно правильно использовать символы верхнего и нижнего регистра при вводе системных атрибутов.

Системные атрибуты для блоков могут быть проверены или изменены в режиме пошагового ввода с помощью команды меню **File > Properties [Файл > Свойства]** в закладке "System Attributes [Системные атрибуты]".

Системные атрибуты для параметров могут быть проверены или изменены в режиме пошагового ввода с помощью команды меню **Edit > Object Properties [Редактировать > Свойства объекта]**. Курсор должен быть помещен в поле имени описания параметра.

11.2.5 Правила установки свойств блоков в исходных файлах на AWL

Вы сможете более легко идентифицировать созданные вами блоки, если вы воспользуетесь свойствами блоков и сможете также защитить эти блоки от несанкционированных изменений.

Свойства блоков могут быть проверены или изменены в режиме пошагового ввода с помощью команды меню **File > Properties [Файл > Свойства]** в закладках "General - Part 1 [Общие свойства –Часть 1]" и "General - Part 2 [Общие свойства –Часть 2].

Другие свойства блоков могут быть введены только в исходном файле.

В исходных файлах применяются следующие правила:

- Свойства блока предшествуют разделу описания переменных.
- Каждое свойство блока находится в собственной строке.
- Строка заканчивается точкой с запятой.
- Свойства блоков определяются с помощью ключевых слов.
- Если вы вводите свойства блока, то они должны появляться в последовательности, показанной в Таблице свойств блоков.
- Свойства блоков, имеющие силу для каждого типа блоков, перечислены в разделе Соответствие: свойство блока – тип блока.

Замечание

Свойства отображаются также в SIMATIC Manager в свойствах объекта для блока. Там могут также редактироваться свойства AUTHOR [автор], FAMILY [семейство], NAME [имя] и VERSION [версия].

Свойства боков и порядок блоков

При вводе свойств блока вы должны соблюдать последовательность ввода, показанную в следующей таблице:

Порядок	Ключевое слово/Свойство	Значение	Пример
1.	[KNOW_HOW_PROTECT]	Защита блока; блок, скомпилированный с этим параметром, не позволяет просматривать свой раздел кодов.	KNOW_HOW_PROTECT
2.	[AUTHOR:]	Имя автора: название компании, отдела или другое имя (не более 8 символов без пробелов)	AUTHOR : Siemens, но не ключевое слово
3.	[FAMILY:]	Имя семейства блоков: например, controllers (не более 8 символов без пробелов)	FAMILY : controllers, но не ключевое слово
4.	[NAME:]	Имя блока (не более 8 символов)	NAME : PID, но не ключевое слово
5.	[VERSION: int1 . int2]	Номер версии блока (оба числа между 0 и 15, т. е. от 0.0 до 15.15)	VERSION : 3.10
6.	[CODE_VERSION1]	Идентификатор того, может ли функциональный блок содержать описания мультиэкземпляров или нет. Если вы хотите описывать мультиэкземпляры, функциональный блок не должен иметь этого свойства	CODE_VERSION1

Порядок	Ключевое слово/Свойство	Значение	Пример
7.	[UNLINKED] только для DB	Блок данных со свойством UNLINKED не связан с программой.	
8.	[READ_ONLY] только для DB	Защита от записи для блоков данных; их данные могут быть прочитаны, но не могут быть изменены.	FAMILY= Examples VERSION= 3.10 READ_ONLY

11.2.6 Разрешенные свойства для каждого типа блоков

Следующая таблица показывает, какие свойства могут быть описаны для каждого типа блоков:

Свойство	OB	FB	FC	DB	UDT
KNOW_HOW_PROTECT	•	•	•	•	–
AUTHOR	•	•	•	•	–
FAMILY	•	•	•	•	–
NAME	•	•	•	•	–
VERSION	•	•	•	•	–
UNLINKED	–	–	–	•	–
READ_ONLY	–	–	–	•	–

Установка защиты блока с помощью KNOW_HOW_PROTECT

Вы можете защитить свои блоки от неавторизованных пользователей путем установки защиты блока с помощью ключевого слова KNOW_HOW_PROTECT, когда вы программируете блок в исходном файле на AWL.

Такая защита блока имеет следующие последствия:

- Если вы захотите посмотреть скомпилированный блок позднее в редакторе пошагового ввода AWL, FUP или KOP, то раздел кодов блока нельзя будет отобразить на экране.
- Таблица описания переменных блока отображает только переменные типов var_in, var_out и var_in_out. Переменные типов var_stat и var_temp остаются скрытыми.
- Ключевое слово KNOW_HOW_PROTECT вводится перед другими свойствами блока.

Установка защиты от записи для блоков данных с помощью READ_ONLY

Для блоков данных вы можете установить защиту от записи, так чтобы блок не переписывался во время обработки. Чтобы сделать это, блок данных должен существовать в форме исходного файла на AWL.

Для установки защиты от записи используйте в исходном файле ключевое слово READ_ONLY. Это ключевое слово должно находиться непосредственно перед описаниями переменных в своей собственной строке.

11.3 Структура блоков в исходных файлах на AWL

11.3.1 Структура блоков в исходных файлах на AWL

Блоки в исходных файлах на AWL структурируются с помощью ключевых слов. В зависимости от типа блока имеются различия в структуре:

- логических блоков
- блоков данных
- типов данных, определенных пользователем (UDT)

11.3.2 Структура логических блоков в исходных файлах на AWL

Логический блок состоит из следующих разделов, каждый из которых идентифицируется соответствующим ключевым словом:

- Начало блока, идентифицируемое ключевым словом и номером блока или именем блока, например:
  "ORGANIZATION_BLOCK OB1" для организационного блока,
  "FUNCTION_BLOCK FB6" для функционального блока или
  "FUNCTION FC1 : INT" для функции.
  У функций указывается также тип функции. Это может быть элементарный или составной тип данных, который определяет тип данных возвращаемого значения (RET_VAL). Если никакое значение не возвращается, то дается ключевое слово VOID.
- Необязательный заголовок блока, вводимый ключевым словом "TITLE [заголовок]" (макс. длина заголовка: 64 символа).
- Дополнительные комментарии, начинающиеся двойным слешем // в начале строки
- Свойства блока (не обязательны)
- Раздел описания переменных
- Раздел кодов, начинающийся с "BEGIN [начни]". Раздел кодов состоит из одного или нескольких сегментов, идентифицируемых ключевым словом "NETWORK". Вы не можете вводить номер сегмента.
- Необязательный заголовок сегмента для каждого используемого сегмента, вводимый ключевым словом "TITLE =" (макс. длина заголовка: 64 символа)

- Дополнительные комментарии для каждого сегмента, начинающиеся двойным слешем // в начале строки
- Конец блока, идентифицируемый ключевым словом END_ORGANIZATION_BLOCK, END_FUNCTION_BLOCK или END_FUNCTION

Между типом блока и его номером должен быть пробел. Символическое имя блока может быть идентифицировано с помощью кавычек, чтобы обеспечить уникальность символических имен локальных переменных и имен из таблицы символов.

11.3.3 Структура блоков данных в исходных файлах на AWL

Блок данных состоит из следующих областей, вводимых соответствующими ключевыми словами:

- Начало блока, идентифицируемое ключевым словом и номером блока или именем блока, например, DATA_BLOCK DB26
- Ссылка на соответствующий UDT или функциональный блок (не обязательна)
- Необязательный заголовок блока, вводимый ключевым словом TITLE = (записи длиннее 64 символов обрезаются)
- Необязательный комментарий к блоку, начинающийся двойным слешем //
- Свойства блока (не обязательны)
- Раздел описания переменных (не обязателен)
- Раздел присваиваний со значениями по умолчанию, начинающийся ключевым словом BEGIN [начни] (не обязателен)
- Конец блока, идентифицируемый ключевым словом END_DATA_BLOCK

Имеется три типа блоков данных:

- Блоки данных, определенные пользователем
- Блоки данных с соответствующим типом данных, определенным пользователем (UDT)
- Блоки данных с соответствующим функциональным блоком (известные как "экземплярные" блоки данных)

11.3.4 Структура типов данных, определенных пользователем в исходных файлах на AWL

Тип данных, определенный пользователем, состоит из следующих областей, вводимых соответствующими ключевыми словами:

- Начало блока, идентифицируемое ключевым словом TYPE [тип] и номером или именем, например, TYPE UDT20
- Структурный тип данных
- Конец блока, идентифицируемый ключевым словом END_TYPE

При вводе типа данных, определенного пользователем, вы должны обеспечить, чтобы этот тип данных предшествовал блоку, в котором он используется.

11.4 Синтаксис и форматы для блоков в исходных файлах на AWL

11.4.1 Синтаксис и форматы для блоков в исходных файлах на AWL

Таблицы форматов показывают синтаксис и форматы. Которые вы должны соблюдать при программировании исходных файлов на AWL. Синтаксис представляется следующим образом:

- Каждый элемент описывается в правом столбце.
- Любые элементы, которые должны быть введены, показаны в кавычках.
- Квадратные скобки [...] означают, что содержимое этих скобок не обязательно.
- Ключевые слова даются буквами в верхнем регистре.

11.4.2 Таблица форматов организационных блоков

В следующей таблице представлен краткий список форматов для организационных блоков в исходном файле на AWL:

Структура	Описание
"ORGANIZATION_BLOCK" ob_no или ob_name	ob_no – это номер блока, например: OB1; ob_name – это символическое имя блока, определенное в таблице символов;
[TITLE=]	Заголовок блока (записи длиннее 64 символов обрезаются)
[Комментарий к блоку]	Комментарии могут вводиться после "//"
[Системные атрибуты для блоков]	Системные атрибуты для блоков
[Свойства блока]	Свойства блока
Раздел описания переменных	Описание временных переменных
"BEGIN"	Ключевое слово для отделения раздела описания переменных от списка команд AWL
NETWORK	Начало сегмента
[TITLE=]	Заголовок сегмента (не более 64 символов)
[Комментарий к сегменту]	Комментарии могут вводиться после "//"
Список команд AWL	Команды блока
"END_ORGANIZATION_BLOCK"	Ключевое слово, завершающее организационный блок

11.4.3 Таблица форматов функциональных блоков

В следующей таблице представлен краткий список форматов для функциональных блоков в исходном файле на AWL:

Структура	Описание
"FUNCTION_BLOCK" fb_no или fb_name	fb_no – это номер блока, например: FB6; fb_name – это символическое имя блока, определенное в таблице символов
[TITLE=]	Заголовок блока (записи длиннее 64 символов обрезаются)
[Комментарий к блоку]	Комментарии могут вводиться после "//"
[Системные атрибуты для блоков]	Системные атрибуты для блоков
[Свойства блока]	Свойства блока
Раздел описания переменных	Описание входных, выходных и проходных параметров и временных или статических переменных Описание параметров может также содержать описания системных атрибутов для параметров.
"BEGIN"	Ключевое слово для отделения раздела описания переменных от списка команд AWL
NETWORK	Начало сегмента
[TITLE=]	Заголовок сегмента (не более 64 символов)
[Комментарий к сегменту]	Комментарии могут вводиться после "//"
Список команд AWL	Команды блока
"END_FUNCTION_BLOCK	Ключевое слово, завершающее функциональный блок

11.4.4 Таблица форматов функций

В следующей таблице представлен краткий список форматов для функций в исходном файле на AWL:

Структура	Описание
"FUNCTION" fc_no : fc_type или fc_name : fc_type	fc_no – это номер блока, например: FC5; fc_name – это символическое имя блока, определенное в таблице символов; fc_type – это тип данных возвращаемого значения (RET_VAL) функции. Это может быть элементарный или составной тип данных или VOID. Если вы хотите использовать системные атрибуты для возвращаемого значения (RET_VAL), то вы должны ввести системные атрибуты для параметров перед столбцом для описания данных.
[TITLE=]	Заголовок блока (записи длиннее 64 символов обрезаются)
[Комментарий к блоку]	Комментарии могут вводиться после "//"
[Системные атрибуты для блоков]	Системные атрибуты для блоков
[Свойства блока]	Свойства блока
Раздел описания переменных	Описание входных, выходных и проходных параметров и временных переменных
"BEGIN"	Ключевое слово для отделения раздела описания переменных от списка команд AWL
NETWORK	Начало сегмента
[TITLE=]	Заголовок сегмента (не более 64 символов)
[Комментарий к сегменту]	Комментарии могут вводиться после "//"
Список команд AWL	Команды блока
"END_FUNCTION"	Ключевое слово, завершающее функцию

11.4.5 Таблица форматов блоков данных

В следующей таблице представлен краткий список форматов для блоков данных в исходном файле на AWL:

Структура	Описание
"DATA_BLOCK" db_no или db_name	db_no – это номер блока, например: DB5; db_name – это символическое имя блока, определенное в таблице символов;
[TITLE=]	Заголовок блока (записи длиннее 64 символов обрезаются)
[Комментарий к блоку]	Комментарии могут вводиться после "//"
[Системные атрибуты для блоков]	Системные атрибуты для блоков
[Свойства блока]	Свойства блока
Раздел описаний	Описание того, связан ли блок с UDT или FB, представленным номером блока или символическим именем из таблицы символов или составным типом данных
"BEGIN"	Ключевое слово для отделения раздела описаний от списка присвоенных значений
[Присваивание начальных значений]	Переменные могут иметь конкретные заданные начальные значения. Отдельным переменным присваиваются постоянные значения или делается ссылка на другие блоки.
"END_DATA_BLOCK"	Ключевое слово, завершающее блок данных

11.5 Сохранение и компиляция исходных файлов на AWL и проверка непротиворечивости

11.5.1 Сохранение исходных файлов на AWL

Вы можете сохранить исходный файл на AWL в любое время в его текущем состоянии. Программа не компилируется и проверка синтаксиса не производится, т. е. все ошибки тоже сохраняются.

Синтаксические ошибки обнаруживаются и сообщаются только при компиляции исходного файла или при проверке непротиворечивости.

Для сохранения исходного файла под тем же именем:

1. Активизируйте окно для исходного файла, который вы хотите сохранить.
2. Выберите команду меню **File > Save [Файл > Сохранить]**.

Для сохранения исходного файла под новым именем/в другом проекте:

1. Активизируйте окно для исходного файла, который вы хотите сохранить.
2. Выберите команду меню **File > Save As [Файл > Сохранить как...]**.
3. В диалоговом окне выберите папку с исходными файлами, в которой вы хотите сохранить данный исходный файл, и введите его новое имя.

11.5.2 Проверка непротиворечивости в исходных файлах на AWL

С помощью команды меню **File > Consistency Check [Файл > Проверка непротиворечивости]** вы можете вывести на экран все синтаксические ошибки, имеющиеся в исходном файле на AWL. В отличие от компиляции, блоки при этом не генерируются.

Когда проверка непротиворечивости завершается, открывается диалоговое окно, показывающее общее количество найденных ошибок.

Все найденные ошибки перечисляются по отдельности в нижней части этого окна со ссылкой на строку. Исправьте эти ошибки до компиляции исходного файла, чтобы все блоки могли быть созданы.

11.5.3 Поиск ошибок в исходных файлах на AWL

Активное окно для исходных файлов разделено на две части. В нижней половине перечисляются следующие ошибки:

- Ошибки, найденные после запуска компиляции командой меню **File > Compile [Файл > Компилировать]**.
- Ошибки, найденные после запуска проверки непротиворечивости с помощью команды меню **File > Consistency Check [Файл > Проверка непротиворечивости]**.

Чтобы найти местоположение ошибки в исходном файле, поместите курсор на соответствующее сообщение об ошибке в нижней части окна. Тогда в верхней части окна автоматически выделяется текстовая строка, содержащая ошибку. Сообщение об ошибке появляется также в строке состояния.

11.5.4 Компиляция исходных фалов на AWL

Требования

Чтобы иметь возможность скомпилировать созданную вами в исходном файле программу в блоки, должны быть выполнены следующие требования:

- Компилироваться могут только исходные файлы, хранящиеся в папке "Source Files [Исходные файлы]" под программой S7.
- Папка "Blocks [Блоки], в которой могут быть сохранены блоки, созданные во время компиляции, как и папка "Source Files [Исходные файлы]", должна тоже находиться под программой S7. Блоки, запрограммированные в исходном файле, создаются только в том случае, если исходный файл был скомпилирован без ошибок. Если в исходном файле запрограммировано несколько блоков, создаются только те, которые не содержат ошибок. После этого вы можете открыть эти блоки, отредактировать их, загрузить их в CPU и отладить их по отдельности.

Последовательность действий в редакторе

1. Откройте исходный файл, который вы хотите скомпилировать. Этот исходный файл должен находиться в папке исходных файлов программы S7, в пользовательской программе S7 которой должны быть сохранены скомпилированные блоки.
2. Выберите команду меню **View > Display > Symbolic Representation [Вид > Отобразить > Символическое представление]**, чтобы впоследствии в скомпилированных блоках могли быть отображены символические имена.
3. Выберите команду меню **File > Compile [Файл > Компилировать]**.
4. На экране появляется диалоговое окно "Compiler Report [Отчет компилятора]", показывающее количество скомпилированных строк и найденные синтаксические ошибки.

Указанные в файле блоки создаются только тогда, когда исходный файл был скомпилирован без ошибок. Если в исходном файле запрограммировано несколько блоков, то создаются только те, в которых нет ошибок.
Предупреждения не препятствуют созданию блоков.
Все синтаксические ошибки, обнаруженные при компиляции, отображаются в нижней части рабочего окна и должны быть исправлены до того, как могут быть созданы соответствующие блоки.

Последовательность действий в SIMATIC Manager

1. Откройте нужную папку "Source Files [Исходные файлы]", дважды щелкнув на ней.
2. Выберите один или несколько исходных файлов, которые вы хотите компилировать. Вы не можете запустить процесс компиляции для закрытой папки с исходными файлами, чтобы скомпилировать все находящиеся в ней исходные файлы.
3. Для запуска компиляции выберите команду меню **File > Compile [Файл > Компилировать]**. Для выбранного вами исходного файла вызывается нужный компилятор. Успешно скомпилированные блоки сохраняются затем в папке блоков под программой S7.
   Все синтаксические ошибки, обнаруженные во время компиляции, отображаются в диалоговом окне и должны быть исправлены, чтобы блоки, в которых были обнаружены ошибки, тоже могли быть созданы.

11.6 Примеры исходных файлов на AWL

11.6.1 Примеры описания переменных в исходных файлах на AWL

Переменные, относящиеся к элементарному типу данных

```
                    // Комментарии отделяются от раздела описаний двойным слешем.
VAR_INPUT           // Ключевое слово для входной переменной.
   in1 : INT;       // Имя переменной и ее тип разделяются двоеточием ":"
   in3 : DWORD;     // Описание каждой переменной завершается точкой с запятой.
   in2 : INT := 10; // Необязательная установка начального значения в описании
END_VAR             // Завершение описания переменных, относящихся к одному типу описания
VAR_OUTPUT          // Ключевое слово для выходной переменной
   out1 : WORD;
END_VAR
VAR_TEMP            // Ключевое слово для временной переменной
   temp1 : INT;
END_VAR
```

Переменная типа Array [массив]

```
VAR_INPUT                                  // Входная переменная
   array1 : ARRAY [1..20] of INT;          // array1 – это одномерный массив
   array2 : ARRAY [1..20, 1..40] of DWORD; // array2 – это двумерный массив
END_VAR
```

Переменные типа "Структура"

```
VAR_OUT                                  // Выходная переменная
OUTPUT1:    STRUCT                       // OUTPUT1 имеет тип данных STRUCT
                  var1 : BOOL;           // 1-й элемент структуры
                  var2 : DWORD;          // 2-й элемент структуры
            END_STRUCT;                  // Конец структуры
END_VAR
```

11.6.2 Пример организационных блоков в исходных файлах на AWL

```
ORGANIZATION_BLOCK OB1
TITLE = Пример OB1 с различными вызовами блоков
//Три сегмента иллюстрируют вызовы блоков
//с параметрами и без параметров

{S7_m_c := 'true'}                       //Системный атрибут для блоков
AUTHOR              Siemens
FAMILY              Пример
NAME                Test_OB
VERSION             1.1
VAR_TEMP
Interim value : INT;                     // Промежуточное значение, буфер
END_VAR

BEGIN

NETWORK
TITLE = Вызов функции с передачей параметров
// Передача параметров в одной строке
CALL FC1 (param1 :=I0.0,param2 :=I0.1);

NETWORK
TITLE = Вызов функционального блока
// с передачей параметров
// Передача параметров в более чем одной строке
CALL Traffic light control , DB6 (       // Имя FB (Управление светофором), экземплярный
                                         // блок данных
dur_g_p         := S5T#10S,              // Присваивание фактических значений параметрам
del_r_p         := S5T#30S,
starter         := TRUE,
t_dur_y_car     := T 2,
t_dur_g_ped     := T 3,
t_delay_y_car   := T 4,
```

```
t_dur_r_car      := T 5,
t_next_red_car   := T 6,
r_car            := "re_main",          // Кавычки показывают символические
y_car            := "ye_main",          // имена, введенные в таблице символов
g_car            := "gr_main",
r_ped            := "re_int",
g_ped            := "gr_int");


NETWORK
TITLE = Вызов функционального блока
// с передачей параметров
// Передача параметров в одной строке
CALL FB10, DB100 (para1 :=I0.0,para2 :=I0.1);

END_ORGANIZATION_BLOCK
```

11.6.3 Пример функций в исходных файлах на AWL

```
FUNCTION FC1: VOID
// Только должна быть вызвана
VAR_INPUT
  param1 : bool;
  param2 : bool;
END_VAR
begin
end_function



FUNCTION FC2 : INT
TITLE = Увеличение количества деталей
// Пока передаваемое значение < 1000, эта функция
// увеличивает передаваемое значение. Если число деталей
// превышает 1000, через возвращаемое значение для функции
// (RET_VAL) возвращается "-1".
```

```
AUTHOR                  Siemens
FAMILY                  Контроль производительности
NAME            :       INCR_ITEM_NOS
VERSION         :       1.0

VAR_IN_OUT
ITEM_NOS : INT;                 // Текущее количество изготовленных деталей
END_VAR

BEGIN

NETWORK
TITLE = Увеличение количества деталей на 1
// Пока текущее количество предметов меньше 1000,
// счетчик может быть увеличен на 1
L ITEM_NOS; L 1000;                     // Пример более одного
> I; JC ERR;                            // оператора в строке.
L 0; T RET_VAL;
L ITEM_NOS; INC 1; T ITEM_NOS; BEU;
ERR: L -1;
T RET_VAL;
END_FUNCTION


FUNCTION FC3 {S7_m_c := 'true'} : INT
TITLE = Увеличение количества деталей
// Пока передаваемое количество < 1000, эта функция
//увеличивает передаваемое значение. Если количество деталей
//превышает 1000, через возвращаемое значение для функции
// (RET_VAL) возвращается "-1".
//
//RET_VAL здесь имеет системный атрибут для параметров

AUTHOR          :       Siemens
FAMILY          :       Контроль производительности
NAME            :       INCR_ITEM_NOS
VERSION         :       1.0

VAR_IN_OUT
ITEM_NOS {S7_visible := 'true'}: INT;   // Текущее количество произведенных деталей
//Системные атрибуты для параметров
```

```
END_VAR

BEGIN

NETWORK
TITLE = Увеличение количества деталей на 1
// Пока текущее количество деталей меньше 1000,
// счетчик может быть увеличен на  1
L ITEM_NOS; L 1000;                    // Пример более чем одного
> I; JC ERR;                           // оператора в строке.
L 0; T RET_VAL;
L ITEM_NOS; INC 1; T ITEM_NOS; BEU;
ERR: L -1;
T RET_VAL;

END_FUNCTION
```

11.6.4 Пример функциональных блоков в исходных файлах на AWL

```
FUNCTION_BLOCK FB6
TITLE = Простое переключение светофора
// Управление светофором на пешеходном переходе
// на главной улице

{S7_m_c := 'true'}        //Системный атрибут для блоков
AUTHOR          :         Siemens
FAMILY          :         Светофор
NAME            :         Светофор01
VERSION         :         1.3

VAR_INPUT

starter         :         BOOL    :=      FALSE;  // Запрос на переход от пешехода
t_dur_y_car     :         TIMER;          // Длительность зеленого для пешеходов
t_next_r_car    :         TIMER;          // Интервал между красными фазами для автомобилей
t_dur_r_car     :         TIMER;

END_VAR
VAR_OUTPUT
```

```
g_car           :       BOOL            :=      FALSE;  // ЗЕЛЕНЫЙ для автомобилей
number          {S7_server := 'alarm_archiv'; S7_a_type := 'alarm_8'} :DWORD;
// Количество автомобилей
// number имеет системные  атрибуты для параметров

END_VAR
VAR
condition       :       BOOL    :=      FALSE;  // Условие красного для автомобилей
END_VAR

BEGIN
NETWORK
TITLE = Условие красного для движения по главной улице
// По истечении минимального интервала, запрос на зеленый свет на
// пешеходном переходе образует условие включения красного сигнала
// для движения по главной улице
                A(;
                A       #starter;       // Запрос на зеленый на пешеходном переходе и
                A       #t_next_r_car;  // время между красными фазами превышено
                O       #condition;     // или условие для красного
                );
                AN      #t_dur_y_car;   // И в настоящее время свет не красный
                =       #condition;     // Условие для красного
NETWORK
TITLE = Зеленый свет для движения по главной улице
                AN      #condition;     // Нет условия для красного для движения по главной
                                        // улице
                =       #g_car;         // ЗЕЛЕНЫЙ для движения по главной улице
NETWORK
TITLE = Длительность желтой фазы для автомобилей
                // Дополнительная программа, необходимая для управления
                // светофорами

END_FUNCTION_BLOCK

FUNCTION_BLOCK FB10
VAR_INPUT
  para1 : bool;
  para2: bool;
end_var
begin
```

```
end_function_block

data_block db10
FB10
begin
end_data_block

data_block db6
FB6
begin
end_data_block
```

11.6.5 Пример блоков данных в исходных файлах на AWL

Блок данных:

```
DATA_BLOCK DB10
TITLE = DB Пример 10
STRUCT
                aa : BOOL;              // Переменная aa типа BOOL
                bb : INT;               // Переменная bb типа INT
                cc : WORD;
END_STRUCT;
BEGIN                                   // Присваивание начальных значений
                aa := TRUE;
                bb := 1500;
END_DATA_BLOCK
```

Блок данных с соответствующим типом данных, определенным пользователем:

```
DATA_BLOCK DB20
TITLE = DB (UDT) Пример
UDT 20                                  // Соответствующий тип данных, определенный
                                        // пользователем
BEGIN
        start := TRUE;                  // Присваивание начальных значений
        setp. := 10;
END_DATA_BLOCK
```

Замечание

Используемый UDT должен находиться в исходном файле до блока данных.

Блок данных с соответствующим функциональным блоком:

```
DATA_BLOCK DB30
TITLE = DB (FB) Пример
FB30                                    // Соответствующий функциональный блок
BEGIN
        start := TRUE;                  // Присваивание начальных значений
        setp. := 10;
END_DATA_BLOCK
```

Замечание

Соответствующий функциональный блок должен находиться в исходном файле до блока данных.

11.6.6 Пример типов данных, определенных пользователем, в исходных файлах на AWL

```
TYPE  UDT20
STRUCT
        start : BOOL;           //Переменная типа BOOL
        setp. : INT;            // Переменная типа INT
        value : WORD;           // Переменная типа WORD
END_STRUCT;
END_TYPE
```

12 Обзор имеющихся в распоряжении справочных данных

12.1 Обзор имеющихся в распоряжении справочных данных

12.1.1 Обзор имеющихся в распоряжении справочных данных

Вы можете создавать и анализировать справочные данные, чтобы облегчить отладку и модификацию своей пользовательской программы. Справочные данные используются в следующих целях:

- Как обзор вашей пользовательской программы в целом
- Как основа для изменений и тестирования
- Как дополнение к вашей программной документации

Следующая таблица показывает, какую информацию вы можете извлечь из отдельных видов справочных данных:

Вид	Назначение
Список перекрестных ссылок	Обзор адресов в областях памяти I, Q, M, P, T, C и DB, используемых в программе пользователя. Используя команду меню **View > Cross References for Address [Вид > Перекрестные ссылки для адресов]**, вы можете отобразить все перекрестные ссылки, включая перекрывающийся доступ к выбранным адресам.
Список назначений для входов, выходов и меркеров (I,Q,M) Список назначений для таймеров и счетчиков (T/C)	Обзор того, какие биты адресов в областях памяти I, Q и M и какие таймеры и счетчики (T и C) уже заняты в программе пользователя; образует важную основу для поиска ошибок или изменений в программе пользователя
Структура программы	Иерархия вызовов блоков внутри программы пользователя и обзор используемых блоков и их уровней вложенности
Неиспользуемые символы	Обзор всех символов, определенных в таблице символов, но не использованных в разделах программы пользователя, для которых доступны справочные данные
Адреса, не имеющие символов	Обзор всех абсолютных адресов, которые используются в разделах программы пользователя, для которых доступны справочные данные, но для которых не были определены символы в таблице символов

Справочные данные для выбранной программы пользователя включают в себя все списки, перечисленные в таблице. Имеется возможность создавать

и выводить на экран один или более списков для одной программы пользователя или для нескольких программ пользователя.

Отображение нескольких видов справочных данных одновременно

Вывод на экран других списков в дополнительных окнах дает вам, например, возможность:

- Сравнивать одноименные списки для различных программ пользователя S7.
- Выводить на экран различные представления списка, например, списка перекрестных ссылок, отображаемых по-разному и помещенных на экране рядом друг с другом. Например, в одном из списков перекрестных ссылок вы можете отобразить только входы программы пользователя S7, а в другом списке только выходы.
- Открывать одновременно несколько списков для одной программы пользователя S7, например, структуру программы и список перекрестных ссылок.

12.1.2 Список перекрестных ссылок

Список перекрестных ссылок дает обзор использования адресов внутри программы пользователя S7.

При выводе на экран списка перекрестных ссылок вы получаете список элементов областей памяти входов (I), выходов (Q), меркеров (M), таймеров (T), счетчиков (C), функциональных блоков (FB), функций (FC), системных функциональных блоков (SFB), системных функций (SFC), периферийных входов/выходов (P) и блоков данных (DB), используемых в программе пользователя S7 вместе с их адресами (абсолютным адресом или символом) и использованием. Он выводится на экран в активном окне. Строка заголовка рабочего окна показывает имя программы пользователя, которой принадлежит список перекрестных ссылок.

Каждая строка в окне соответствует записи в списке перекрестных ссылок. Функция поиска облегчает нахождение конкретных адресов и символов.

Список перекрестных ссылок отображается по умолчанию при выводе на экран справочных данных. Вы можете изменить это умолчание.

Структура

Запись списка перекрестных ссылок состоит из следующих столбцов:

Столбец	Содержимое/Значение
Address [Адрес]	Абсолютный адрес
Symbol [Символ]	Символическое имя адреса
Block [Блок]	Блок, в котором используется адрес
Type [Тип]	Используется ли доступ к адресу на чтение (R) и/или на запись (W)
Language/Details [Язык / Подробности]	Информация о языке программирования, использованном при создании блока

Столбцы Symbol [Символ], Block [Блок], Type [Тип] и Language/Details [Язык/ Подробности] отображается только в том случае, если для списка перекрестных ссылок были выбраны соответствующие параметры. Информация о языке и подробностях отображается в одном столбце, и только весь столбец может быть активизирован или деактивизирован. Эта информация о блоке варьируется в зависимости от языка программирования, на котором блок был написан.

С помощью мыши вы можете устанавливать на экране требуемую ширину столбца в списке перекрестных ссылок.

Сортировка

По умолчанию список перекрестных ссылок сортируется по областям памяти. Щелкнув мышью на заголовке столбца, вы можете рассортировать записи этого столбца по критериям сортировки, установленным по умолчанию.

Пример компоновки списка перекрестных ссылок

Адрес	Символ	блок	Тип	Язык	Подробности
I1.0	Двигатель_вкл	OB2	R	STL	Nw 2 Inst 33 /0
M1.2	Меркер	FC2	R	LAD	Nw 33
C2	Счетчик2	FB2		FBD	Nw2

Здесь: Nw – сегмент, Inst – команда

12.1.3 Структура программы

Структура программы описывает иерархию вызовов блоков внутри программы пользователя S7. Вам также дается обзор используемых блоков, их зависимостей и потребностей в локальных данных.

С помощью команды меню **View > Filter [Вид > Фильтр]** в окне "Generating Reference Data [Генерирование справочных данных] вы открываете диалоговое окно с закладками. В закладке "Program Structure [Структура программы] вы можете установить, в каком виде вы хотите отобразить структуру программы.

Вы можете выбирать между:

- древовидной структурой и
- структуры в виде родительских и дочерних записей (табличная форма)

Вы можете указать, хотите ли вы отобразить все блоки или иерархия должна начинаться с конкретного начального блока.

Символы в структуре программы

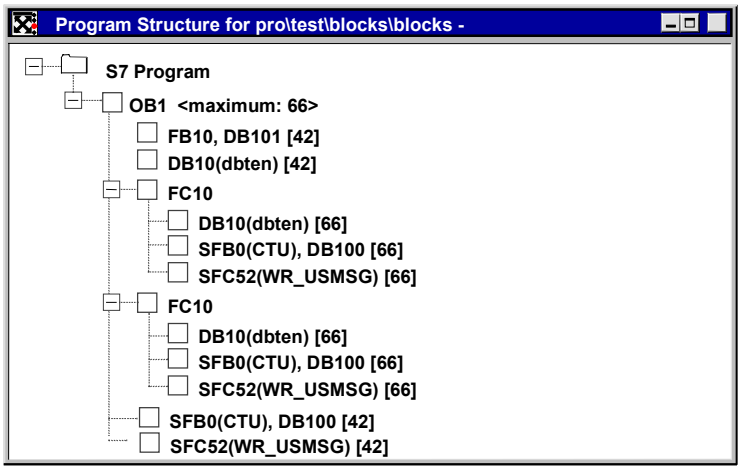
Символ	Значение
	Блок, вызываемый стандартно (CALL FB10)
	Блок, вызываемый безусловно (UC FB10)
	Блок, вызываемый условно (CC FB10)
	Блок данных
	Рекурсия
	Рекурсия, вызываемая условно
	Рекурсия, вызываемая безусловно
	Блок не вызывается

- Рекурсии в вызове в древесной структуре распознаются и индицируются визуально .
- Рекурсии внутри иерархии вызовов отмечаются разными символами.
- Стандартно вызываемые блоки (CALL), условно вызываемые блоки (CC) и безусловно вызываемые блоки (UC) отмечаются разными символами.
- Блоки, которые не вызываются, показываются внизу древовидной структуры и помечаются черным крестом. Другие разрывы структуры вызовов, кроме разрывов, обусловленных не вызванными блоками, отсутствуют.

Древовидная структура

Показывается вся иерархия вызовов, начиная с конкретного блока.

Структура каждой программы обладает ровно одним блоком, являющимся ее корнем. Это может быть блок OB1 или любой другой блок, предварительно установленный пользователем в качестве начального.



Если структура программы должна быть создана для всех организационных блоков (OB), OB1 отсутствует в программе пользователя S7, или если был указан начальный блок, отсутствующий в программе, то для вас автоматически выводится приглашение указать другой блок в качестве корня структуры программы.

Отображение множественных вызовов может быть деактивизировано при настройке параметров как для древовидной структуры, так и для структуры в виде родительских и дочерних записей.

Отображение максимальной потребности в локальных данных в древовидной структуре

Чтобы дать быстрый обзор потребности в локальных данных организационных блоков в отображаемой программе пользователя, на экран в древовидной структуре может быть выведена следующая информация:

- максимальная потребность в локальных данных на OB и
- потребность в локальных данных на путь

Вы можете активизировать и деактивизировать отображение этой информации в закладке ”Program Structure [Структура программы]".

Чтобы вывести на экран потребность в локальных данных выделенного блока, щелкните правой кнопкой мыши и выберите в контекстно-чувствительном меню команду **”Block Information [Информация о блоке]"**.

Если имеются в наличии OB синхронных ошибок (OB121, OB122), то после числового значения для максимальной потребности в локальных данных отображается знак плюс и дополнительная потребность для OB синхронных ошибок.

Структура в виде родительских и дочерних записей

Показываются вызывающий и вызываемый блок. Такое попарное представление дается для каждого блока в программе пользователя S7.

Отображение удаленных блоков

Строки, соответствующие удаленным блокам, отображаются красным цветом, а после этих блоков отображается строка символов "?????".

12.1.4 Список назначений для входов, выходов и меркеров (I/Q/M)

Список назначений показывает, какие адреса уже назначены внутри программы пользователя. Это отображение является важной основой для поиска ошибок и выполнения изменений в программе пользователя.

Отображение списка назначений I/Q/M дает обзор того, какой бит, в каком байте областей памяти входов (I), выходов (Q) и меркеров (M) используется. Список назначений I/Q/M отображается в рабочем окне. Строка заголовка рабочего окна показывает имя программы пользователя S7, к которой относится список назначений.

Каждая строка содержит один байт области памяти, в которой закодированы восемь битов в соответствии с доступом к ним. Она также показывает, происходит ли обращение к байту, слову или двойному слову.

Коды в списке назначений I/Q/M

. к этому адресу нет обращения, следовательно, он не назначен

o обращение к этому адресу происходит непосредственно

x обращение к этому адресу происходит косвенно (обращение к байту, слову или двойному слову)

Столбцы в списке назначений I/Q/M

Столбец	Содержимое/Значение
7 6 5 4 3 2 1 0	Номер бита соответствующего байта
B	Байт занят обращением к одному байту
W	Байт занят обращением к одному слову
D	Байт занят обращением к двойному слову

Пример компоновки списка назначений (I/Q/M)

Следующий пример показывает типичную компоновку списка назначений для входов, выходов и меркеров (I/Q/M).

Адрес	7	6	5	4	3	2	1	0		B	W	D
QB0	O	X	X	O	X	X	X	X		O	.	.
QB1	.	O	.	.	O	.	O	.		.	.	.
IB0	O	O	O	.	O	.	O	.		.	.	.
IB1	.	.	.	.	.	.	.	.		.	.	.
MB0	X	X	X	X	X	X	X	X		.	O	.
MB1	X	X	X	X	X	X	O	X		.	.	.

Первая строка дает назначения для выходного байта QB0. Обращение к адресу QB0 происходит побайтно. Однако, в то же время имеется битовый доступ к выходным битам Q 0.4 и Q 0.7. Поэтому в столбцах "4" и "7" стоит "O". В столбцах "0", "1", "2", "3", "5" и "6" стоит "X", что указывает на байтовый доступ. "O" появляется в столбце B, так как имеется байтовый доступ к адресу QB0.

12.1.5 Список назначений для таймеров и счетчиков (T/C)

Список назначений показывает, какие адреса уже назначены внутри программы пользователя. Это отображение является важной основой для поиска ошибок и выполнения изменений в программе пользователя.

Отображение списка назначений T/C дает обзор того, какие таймеры (T) и счетчик (C) используются.

Список назначений T/C отображается в рабочем окне. Строка заголовка рабочего окна показывает имя программы пользователя S7, к которой относится список назначений. Каждая строка содержит 10 таймеров или счетчиков.

Коды в списке назначений T/C

. не используется

x используется

Пример компоновки списка назначений (T/C)

	0	1	2	3	4	5	6	7	8	9
T 00-09	.	X	.	.	.		X	.	.	.
T 10-19	.	.	X	.	.	.	.	X	.	X
T 20-29	.	.	.	.	X	.	.	.	.	.
C 00-09	.	.	X	.	.	.	.	X	.	.
C 10-19	.	.	.	.	.	.	.	.	.	X
C 20-29	.	.	.	.	.	.	.	.	.	.
C 30-39	.	.	.	.	X	.	.	.	.	.

В этом примере назначены таймеры T1, T6, T12, T17, T19, T24 и счетчики C2, C7, C19, C34.

Эти списки сортируются по алфавиту. Вы можете сортировать записи, щелкнув на заголовке столбца.

12.1.6 Неиспользованные символы

Вам дается обзор всех символов со следующими характеристиками:

- Символы, определенные в таблице символов.
- Символы, не используемые в тех частях программы пользователя, для которых существуют справочные данные.

Они отображаются в активном окне. Строка заголовка рабочего окна показывает имя программы пользователя S7, к которой относится список.

Каждая строка, показанная в этом окне, соответствует записи из списка. Строка состоит из адреса, символа, типа данных и комментария.

Столбец	Содержимое/Значение
Symbol [Символ]	Символическое имя
Address [Адрес]	Абсолютный адрес
Data Type [Тип данных]	Тип данных адреса
Comment [Комментарий]	Комментарий к адресу из таблицы символов

Пример компоновки списка неиспользуемых символов

Symbol [Символ]	Address [Адрес]	Data Type [Тип данных]	Comment [Комментарий]
MCB1	I 103.6	BOOL	Автоматический выключатель двигателя 1
MCB2	I 120.5	BOOL	Автоматический выключатель двигателя 2
MCB3	I 121.3	BOOL	Автоматический выключатель двигателя 3

Вы можете отсортировать записи, щелкнув на заголовке столбца.

12.1.7 Адреса без символов

Когда вы выводите на экран список адресов, не имеющих символов, вы получаете список элементов, используемых в программе пользователя S7, которые не определены в таблице символов. Они отображаются в активном окне. Строка заголовка рабочего окна показывает имя программы пользователя S7, к которой относится список.

Строка состоит из адреса и количества раз, которое этот адрес используется в программе пользователя.

Пример:

Address [Адрес]	Number [Количество]
Q 2.5	4
I 23.6	3
M 34.1	20

Список рассортирован по адресам.

12.1.8 Отображение информации о блоках для KOP, FUP и AWL

Информация о блоках отображается для контактного плана, функционального плана и списка команд в списке перекрестных ссылок и в структуре программы. Эта информация состоит из языка блока и подробностей.

В представлении структуры программы информация о блоках отображается с помощью команды меню **View > Block Information [Вид > Информация о блоке]** или через правую кнопку мыши. Это отображение зависит от того, было ли при настройке фильтра в закладке "Program Structure [Структура программы] выбрано представление "Parent/Child Structure [Структура в виде родительских и дочерних записей]" или представление "Tree Structure [Древовидная структура]".

В отображении "Cross References [Перекрестные ссылки]" информация о блоках может включаться и выключаться с помощью команды меню **View > Filter [Вид > Фильтр]**.

- Активизируйте триггерную кнопку "Block language and details [Язык блока и подробности]" в закладке "Cross References [Перекрестные ссылки]" диалогового окна "Filter [Фильтр]", чтобы отобразить информацию о блоках.

Информация о блоке варьируется в зависимости от языка, на котором был написан блок и отображается с использованием следующих сокращений:

Язык	Сегмент	Оператор	Команда
STL (или AWL)	Nw	Inst	/
LAD (или KOP)	Nw		
FBD (или FUP)	Nw		

Nw и Inst указывают, в каком сегменте и в каком операторе используется адрес (список перекрестных ссылок) или вызывается блок (структура программы).

Отображение информации о блоках для дополнительных языков программирования

Доступ к темам оперативной помощи относительно получения информации о блоках может быть получен, если установлен соответствующий дополнительный пакет.

12.2 Работа со справочными данными

12.2.1 Способы отображения справочных данных

Для отображения справочных данных имеются в распоряжении следующие способы:

Отображение из SIMATIC Manager

1. В окне проекта в отображении компонентов в режиме offline выберите папку "Blocks [Блоки]".
2. Выберите команду меню **Options > Reference Data > Display [Параметры > Справочные данные > Отобразить]**.

Отображение из окна редактора

1. Откройте блок в папке "Blocks [Блоки]".
2. В окне редактора языка программирования выберите команду меню **Options > Reference Data [Параметры > Справочные данные]**.

Запускается приложение для вывода на экран справочных данных, и отображается список перекрестных ссылок для выбранной программы пользователя (отображение по умолчанию для первого отображения справочных данных). Если справочные данные не полны, открывается диалоговое окно, из которого вы можете запустить обновление справочных данных.

Отображение непосредственно из скомпилированного блока

Вы можете вывести на экран справочные данные для скомпилированного блока непосредственно из языкового редактора, чтобы получить текущий обзор своей пользовательской программы.

12.2.2 Отображение списков дополнительных рабочих окон

С помощью команды меню **Window > New Window [Окно > Новое окно]** вы можете открывать дополнительные рабочие окна и выводить на экран другие представления справочных данных (например, Список неиспользованных символов).

Рабочее окно для ранее не выведенных справочных данных открывается с помощью команды меню **Reference Data > Open [Справочные данные > Открыть]**.

Вы можете перейти к другому отображению справочных данных, выбрав одну из команд в меню "View [Вид]" или нажав соответствующую кнопку на панели инструментов.

Отображение справочных данных	Соответствующая команда меню
Адреса без символов	View > Addresses Without Symbols [Вид > Адреса без символов]
Неиспользованные символы	View > Unused Symbols [Вид > Неиспользованные символы]
Список назначений I/Q/M	View > Assignment > Inputs, Outputs, and Bit Memory [Вид > Назначение > Входы, выходы и меркеры]
Список назначений T/C	View > Assignment > Timers and Counters [Вид > Назначение > Таймеры и счетчики]
Структура программы	View > Program Structure [Вид > Структура программы]
Список перекрестных ссылок	View > Cross References [Вид > Список перекрестных ссылок]

12.2.3 Генерирование и отображение справочных данных

Генерирование справочных данных:

1. В SIMATIC Manager выберите папку блоков, для которых вы хотите сгенерировать справочные данные.
2. Выберите в SIMATIC Manager команду меню **Options > Reference Data > Generate [Параметры > Справочные данные > Генерировать]**.

Перед генерированием справочных данных компьютер проверяет, доступны ли справочные данные и если да, то являются ли они текущими.

- Если справочные данные доступны, то они генерируются.
- Если доступные справочные данные не являются текущими, то вы можете выбрать, обновить ли эти справочные данные или сгенерировать их полностью снова.

Отображение справочных данных:

Справочные данные можно отобразить с помощью команды меню **Options > Reference Data > Display [Параметры > Справочные данные > Отобразить]**.

Перед отображением справочных данных выполняется проверка, чтобы убедиться, существуют ли те или иные справочные данные и являются ли текущими существующие справочные данные.

- Если справочные данные не существуют, то они генерируются.
- Если существуют неполные справочные данные, то открывается диалоговое окно с предупреждением о нарушении целостности справочных данных. После этого вы можете принять решение, обновить ли справочные данные и до какой степени. Далее у вас есть следующие возможности:

Выбор	Значение
Только для измененных блоков	Справочные данные обновляются для всех измененных или новых блоков; информация о любых удаленных блоках удаляется из справочной базы данных.
Для всех блоков	Справочные данные генерируются снова с начальной позиции для всех блоков.
Не обновлять	Справочные данные не обновляются.

Чтобы обновить справочные данные, блоки перекомпилируются. Вызывается соответствующий компилятор для компиляции каждого блока. С помощью команды меню **View > Update [Вид > Обновить]** вы можете обновить отображение справочных данных, уже выведенное в активном окне.

12.2.4 Быстрый поиск расположения адреса в программе

Вы можете использовать справочные данные при программировании, чтобы поместить курсор в различные места нахождения адреса в программе. Чтобы сделать это, вы должны иметь новейшие справочные данные. Однако вам нет необходимости запускать приложение для отображения справочных данных.

Основная последовательность действий

1. Выберите в SIMATIC Manager команду меню **Options > Reference Data > Generate [Параметры > Справочные данные > Генерировать]**, чтобы сгенерировать текущие справочные данные. Этот шаг необходим только в том случае, если справочные данные отсутствуют или у вас имеются старые справочные данные.
2. Выберите адрес в открытом блоке.
3. Выберите команду меню **Edit > Go To > Location [Редактировать > Перейти к > Местоположение]**.
   Теперь появляется диалоговое окно, содержащее список с местами нахождения этого адреса в программе.
4. Выберите параметр "Overlapping access to memory areas [Пересекающиеся обращения к областям памяти]", если вы также хотите отобразить местонахождение операндов, физические адреса которых или области адресов пересекаются с адресом вызываемого операнда. К таблице добавляется столбец "Address [Адрес]".
5. Выберите место в списке и щелкните на кнопке "Go To [Перейти к]".

Если справочные данные устарели к моменту открытия диалогового окна, то об этом появится сообщение. После этого вы можете обновить справочные данные.

Список местоположений адресов

Список местоположений адресов содержит в диалоговом окне следующую информацию:

- Блок, в котором используется адрес
- Символическое имя этого блока, если оно существует
- Подробности, например, информацию о местоположении и, если необходимо, команду, что зависит от первоначального языка программирования блока или исходного файла (SCL)
- Информацию, зависящую от языка
- Тип доступа к адресу: только чтение (R), только запись (W), чтение и запись (RW), неизвестно (?).
- Язык блока

Вы можете отфильтровать отображение местоположений адресов и таким способом просмотреть, например, только доступ к адресам на запись. Оперативная помощь для этого диалогового окна предоставит вам более подробную информацию о том, что следует вводить в поля, и другую отображаемую информацию.

Замечание

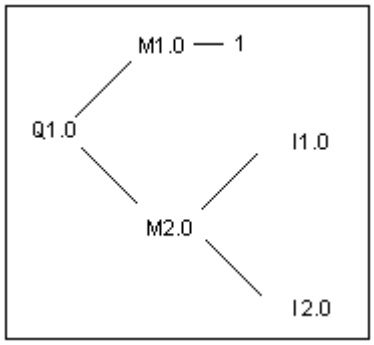
Справочные данные существуют только в режиме offline. Следовательно, эта функция работает только с перекрестными ссылками блоков, отображаемых offline, даже если вы вызываете эту функцию в блоке, открытом online.

12.2.5 Пример работы с местоположениями адресов

Вы хотите определить, в каких местах установлен выход Q1.0 (непосредственно или косвенно). В качестве примера в OB1 используется следующий код AWL:

```
Network 1: .....
A Q 1.0  // в этом примере
= Q 1.1  // не имеет значения

Network 2:
A M1.0
A M2.0
= Q 1.0  // присваивание

Network 3:
//только строка комментария
SET
= M1.0  // присваивание

Network 4:
A I 1.0
A I 2.0
= M2.0  // присваивание
```

Это дает следующее дерево присваиваний для Q1.0:



Далее действуйте следующим образом:

1. Поместите курсор на Q1.0 (NW 1, Inst 1) в OB1 в редакторе KOP/AWL/FUP.
2. Выберите команду меню **Edit > Go To > Location [Редактировать > Перейти к > Местоположение]** или используйте правую кнопку мыши, чтобы выбрать "Go to Location [Перейти к месту нахождения]". Теперь диалоговое окно отобразит все назначения для Q1.0:
```
OB1     Cycle Execution    NW 2  Inst 3  /=    W    STL
[OB1 Циклическое выполнение Сегм.2 Ком.3 /=  запись AWL]
OB1     Cycle Execution    NW 1  Inst 1  /A    R    STL
[OB1 Циклическое выполнение Сегм.1 Ком.3 /A  чтение AWL]
```
3. Перейдите в редакторе к "NW 2 Inst 3 [Сегмент 2 команда 3]" с помощью кнопки "Go To [Перейти к]" в диалоговом окне:
```
Network 2:
A M1.0
A M2.0
= Q 1.0
```
4. Назначения для M1.0 и M2.0 теперь не должны проверяться. Сначала поместите курсор на M1.0 в редакторе KOP/AWL/FUP.
5. Выберите команду меню **Edit > Go To > Location [Редактировать > Перейти к > Местоположение]** или используйте правую кнопку мыши, чтобы выбрать "Go to Location [Перейти к месту нахождения]". Теперь диалоговое окно отображает все назначения для M1.0:
```
OB1     Cycle Execution    NW 3  Inst 2  /=    W    STL
[OB1 Циклическое выполнение Сегм.3 Ком.2 /=  запись AWL]
OB1     Cycle Execution    NW 2  Inst 1  /A    R    STL
[OB1 Циклическое выполнение Сегм.2 Ком.1 /A  чтение AWL]
```
6. Перейдите в редакторе к "NW 3 Inst 2 [Сегмент 3 команда 2]" с помощью кнопки "Go To [Перейти к]" в диалоговом окне.
7. В редакторе KOP/AWL/FUP в сегменте 3 вы увидите, что присваивание M1.0 не имеет значения (т. к. он всегда TRUE) и что вместо этого следует рассмотреть присваивание M2.0.

В STEP 7 версий, более ранних, чем V5, вам теперь бы пришлось снова пройти всю цепочку присваиваний. Кнопки ">>" и "<<" делают это значительно проще:

8. Переместите на передний план диалоговое окно "Go to Location [Перейти к месту нахождения]" или вызовите функцию "Go to Location [Перейти к месту нахождения]" в редакторе KOP/AWL/FUP из своей текущей позиции.
9. Щелкните на кнопке "<<" один или два раза, пока не отобразятся все местоположения Q1.0 are displayed; последнее место перехода "NW 2 Inst 3" выбирается.
10. Перейдите из диалогового окна местоположений адресов к "NW 2 Inst 3" в редакторе с помощью кнопки "Go To [Перейти к]" (как в пункте 3):
```
Network 2:
A M1.0
A M2.0
= Q 1.0
```
11. В пункте 4 было проверено присваивание M1.0. Теперь вам нужно проверить все присваивания (прямые или косвенные) меркеру M2.0. Поместите в редакторе курсор на M2.0 и вызовите функцию "Go to Location [Перейти к месту нахождения]": Отображаются все назначения M2.0:
```
OB1     Cycle Execution    NW 4  Inst 3  /=    W    STL
[OB1 Циклическое выполнение Сегм.4 Ком.3 /=  запись AWL]
OB1     Cycle Execution    NW 2  Inst 2  /A    R    STL
[OB1 Циклическое выполнение Сегм.2 Ком.2 /A  чтение AWL]
```
12. Перейдите к "NW 4 Inst 3 [Сегмент 4 команда 3]" в редакторе KOP/AWL/FUP с помощью кнопки "Go To [Перейти к]":
```
Network 4:
A I 1.0
A I 2.0
= M2.0
```
13. Теперь вам нужно проверить присваивания I1.0 и I2.0. Этот процесс не описан в данном примере, так как вы будете действовать так же, как и раньше (пункт 4 и далее).

Переключаясь между редактором KOP/AWL/FUP и диалоговым окном местоположения адресов, вы можете найти и проверить все имеющие значение места нахождения адресов в своей программе.

13 Метка времени как свойство блока и конфликты меток времени

13.1 Метка времени как свойство блока и конфликты меток времени

Блоки содержат метку времени кода и метку времени интерфейса. Эти метки времени отображаются в диалоговом окне свойств блока. С помощью этих меток вы можете контролировать непротиворечивость программ STEP 7.

STEP 7 отображает конфликт меток времени, если при сравнении меток времени он обнаруживает нарушение правил. Могут произойти следующие нарушения:

- вызываемый блок является более новым, чем вызывающий блок (CALL)
- блок, на который ссылаются, является более новым, чем блок, который его использует

Примеры нарушений второго типа:

- UDT является более новым, чем блок, который его использует, т. е. DB, или другой UDT, или FC, или FB, или OB, который использует UDT в таблице описания переменных.
- FB является более новым, чем соответствующий экземплярный DB.
- FB2 определен как мультиэкземпляр в FB1, и FB2 является более новым, чем FB1.

Замечание

Даже если соотношение между метками времени интерфейсов правильно, могут возникнуть противоречия:

Определение интерфейса для блока, на который производится ссылка, не соответствует определению в том месте, где он используется.

Эти противоречия известны как конфликты интерфейсов. Они могут возникнуть, например, когда блоки копируются из разных программ или когда при компиляции исходного файла в формате ASCII не все блоки в программе оказываются сгенерированными.

13.2 Метки времени в логических блоках

Метка времени кода

Здесь вводятся время и дата создания блока. Эта метка времени обновляется:

- когда изменяется код программы
- когда изменяется описание интерфейса
- когда изменяется комментарий
- когда в первый раз создается и компилируется исходный ASCII-файл
- когда изменяются свойства блока (диалоговое окно "Properties [Свойства]")

Метка времени интерфейса

Эта метка времени обновляется:

- когда изменяется описание интерфейса (изменения типов данных или начальных значений, новые параметры)
- когда в первый раз создается и компилируется исходный ASCII-файл, если интерфейс изменяется структурно

Эта метка времени не обновляется:

- когда изменяются символы
- когда изменяется комментарии в описании переменных
- когда производятся изменения в области TEMP

Правила для вызовов блоков

- Метка времени интерфейса вызываемого блока должна быть старше метки времени вызывающего блока.
- Изменяйте интерфейс блока только в том случае, если не открыт ни один блок, который вызывает данный блок. В противном случае, если вы сохраняете вызывающие блоки позднее, чем измененный блок, вы не сможете распознать это противоречие из метки времени.

Последовательность действий при возникновении конфликта меток времени

Конфликт меток времени отображается, когда открывается вызывающий блок. После выполнения изменений в интерфейсе FB или FC все обращения к этому блоку в вызывающих блоках показываются в расширенной форме.

Если интерфейс блока был изменен, все блоки, вызывающие этот блок, также должны быть скорректированы.

После выполнения изменений в интерфейсе FB должны быть обновлены существующие определения мультиэкземпляров и экземплярные блоки данных.

13.3 Метки времени в совместно используемых блоках данных

Метка времени кода

Эта метка времени обновляется:

- когда исходный ASCII-файл создается в первый раз
- когда исходный ASCII-файл компилируется
- когда выполняются изменения в отображении описаний или в отображении данных блока

Метка времени интерфейса

Эта метка времени обновляется:

- когда изменяется описание интерфейса в отображении описаний (изменения типов данных или начальных значений, новые параметры).

13.4 Метки времени в экземплярных блоках данных

Экземплярный блок данных сохраняет формальные параметры и статические данные для функциональных блоков.

Метка времени кода

Здесь вводятся время и дата создания экземплярных блоков данных. Метка времени обновляется, когда вы вводите текущие значения в отображении данных экземплярного блока данных. Пользователь не может изменить структуру экземплярного блока данных, так как эта структура получается из соответствующего функционального блока (FB) или системного функционального блока (SFB).

Метка времени интерфейса

При создании экземплярного блока данных вводится метка времени интерфейса соответствующего FB или SFB.

Правила открытия без конфликтов

Метки времени интерфейса FB/SFB и соответствующего экземплярного блока данных должны совпадать.

Последовательность действий при возникновении конфликта меток времени

Если вы изменяете интерфейс FB, то метка времени интерфейса этого FB обновляется. Когда вы открываете соответствующий экземплярный блок данных, то появляется сообщение о конфликте меток времени, так как метки времени экземплярного блока данных и FB больше не совпадают. В разделе описаний блока данных интерфейс отображается с символами, сгенерированными компилятором (псевдосимволы). Экземплярный блок данных теперь можно только просматривать.

Для устранения конфликтов меток времени этого типа вы должны создать экземплярный блок данных для измененного FB снова.

13.5 Метки времени в UDT и блоках данных, полученных из UDT

Типы данных, определенные пользователем (UDT), могут быть использованы, например, для создания нескольких блоков данных с одинаковой структурой.

Метка времени кода

Метка времени кода обновляется при каждом изменении.

Метка времени интерфейса

Метка времени интерфейса обновляется, когда изменяется описание интерфейса (изменения типов данных или начальных значений, новые параметры).

Метка времени интерфейса UDT обновляется также при компиляции исходного ASCII-файла.

Правила открытия без конфликтов

- Метка времени интерфейса типа данных, определенного пользователем, должна быть старше, чем метка времени интерфейса в логических блоках, в которых этот тип данных используется.
- Метка времени интерфейса типа данных, определенного пользователем, должна быть идентична метке времени блока данных, полученного из UDT.
- Метка времени интерфейса типа данных, определенного пользователем, должна быть младше, чем метка времени вторичного UDT.

Последовательность действий при возникновении конфликта меток времени

Если вы изменяете определение UDT, который используется в блоке данных, функции, функциональном блоке или в другом определении UDT, STEP 7 сообщает о конфликте меток времени при открытии блока.

Компонент UDT показывается в виде расширенной структуры. Все имена переменных заменяются значениями, предустановленными системой.

14 Проектирование сообщений

14.1 Концепция сообщений

14.1.1 Концепция сообщений

Сообщения дают вам возможность быстро обнаруживать, локализовать и устранять ошибки, возникающие при обработке программ на программируемых контроллерах, существенно сокращая, таким образом, непроизводительные потери времени установки.

Прежде чем сообщения могут быть выведены, они должны быть сначала спроектированы.

С помощью STEP 7 вы можете создавать и редактировать сообщения, связанные с событиями назначенными текстами сообщений и атрибутами. Вы можете также компилировать сообщения и выводить их на устройства отображения.

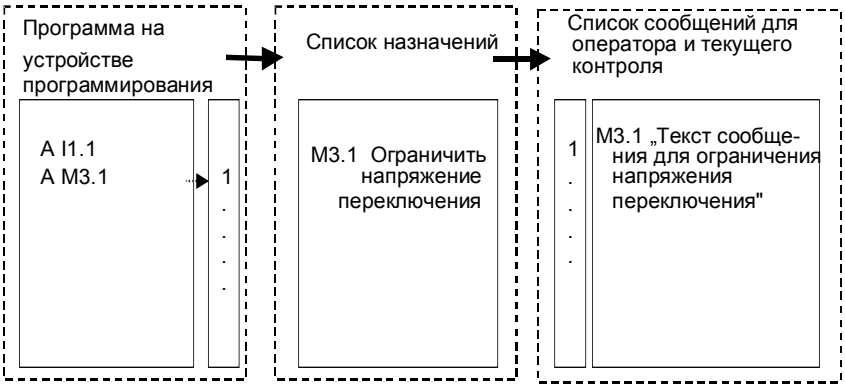
14.1.2 В чем состоят различные методы сообщений?

Имеются различные методы создания сообщений.

Битовый обмен сообщениями

Битовый обмен сообщениями требует от программиста выполнения трех шагов:

- Создать программу пользователя на устройстве программирования и установит требуемый бит.
- Создать список назначений, используя любой текстовый редактор, в котором текст сообщения назначается биту сообщения (например, M 3.1 = Ограничить напряжение переключения).
- Создать список текстов сообщений на панели оператора на основе списка назначений.

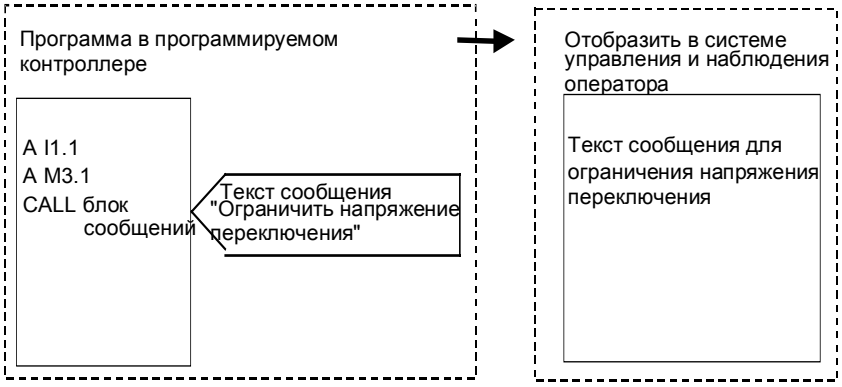


Система взаимодействия с оператором циклически опрашивает программируемый контроллер, изменился бит сообщения или нет. Если программируемый контроллер сообщает об изменении, то соответствующее сообщение отображается. Сообщение получает метку времени от системы взаимодействия с оператором.

Нумерация сообщений

Нумерация сообщений требует от программиста выполнения только одного шага:

- Создать программу пользователя на устройстве программирования, установить требуемый бит и назначить требуемый текст сообщения этому биту непосредственно при программировании.



Циклический опрос программируемого контроллера отсутствует. Когда программируемый контроллер сигнализирует об изменении, в систему взаимодействия с оператором передается номер соответствующего сообщения, и текст соответствующего сообщения отображается. Сообщение получает метку времени из программируемого контроллера и поэтому может более точно отслеживаться, чем в случае битового обмена сообщениями.

14.1.3 Выбор метода сообщений

Обзор

В следующей таблице показаны свойства и требования для различных методов сообщений:

Нумерация сообщений	**Битовый обмен сообщениями**
Управление сообщениями производится в общей базе данных для устройства программирования и панели оператора.	Отсутствует общая база данных для устройства программирования и панели оператора.
Загрузка в шину на низком уровне (программируемый контроллер активен).	Загрузка в шину на высоком уровне (опрос ведет панель оператора).
Сообщения получают метку времени от программируемого контроллера.	Сообщения получают метку времени от панели оператора.

Метод нумерации сообщений распознает следующие три типа сообщений:

Сообщения, связанные с блоками	*Сообщения, связанные с символами*	*Диагностические сообщения, определенные пользователем*
Синхронны по отношению к программе Отображение с помощью WinCC и ProTool (только ALARM_S) Возможны у S7-300/400 Программа использует блоки сообщений: • ALARM • ALARM_8 • ALARM_8P • NOTIFY • ALARM_S(Q) • AR_SEND Передача на панель оператора • для WinCC через конфигурацию соединения ПЛК – станция оператора • для ProTool	Асинхронны по отношению к программе Отображение с помощью WinCC Возможны только у S7-400 Конфигурирование через таблицу символов Загрузка в программируемый контроллер через системные блоки данных (SDB) Передача на панель оператора через конфигурацию соединения ПЛК – станция оператора	Синхронны по отношению к программе Отображение в диагностическом буфере на устройстве программирования Возможны у S7-300/400 Программа использует блок сообщений (системная функция) • WR_USMSG Передача на панель оператора отсутствует

STEP 7 поддерживает более дружественный по отношению к пользователю метод нумерации сообщений, который будет подробно описан ниже.

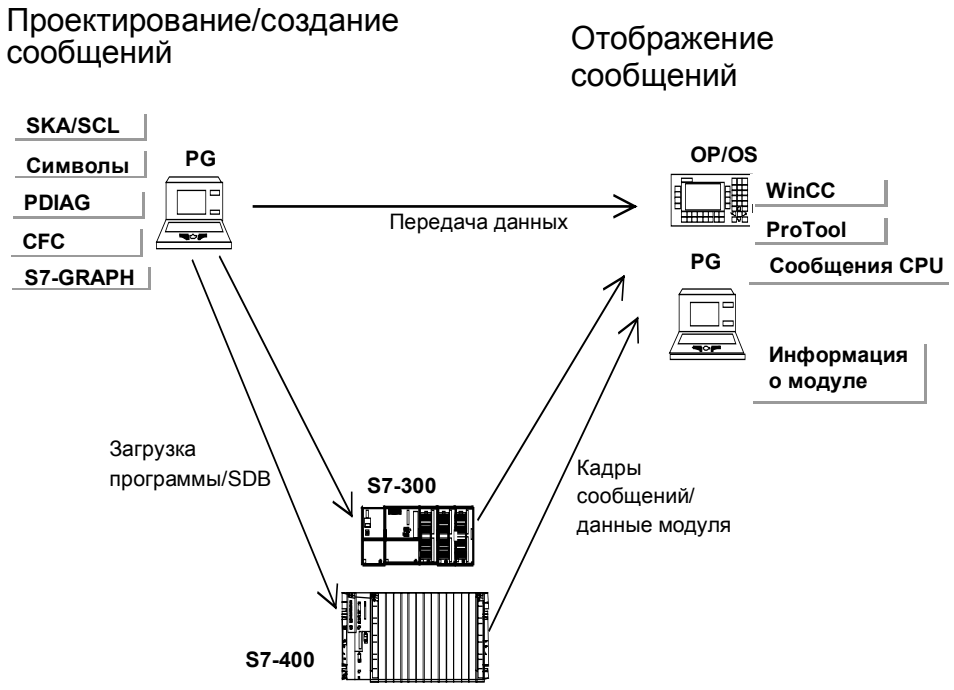
Примеры нумерации сообщений

Метод обмена сообщениями	Применение
Сообщения, связанные с блоками	Используются для сообщения о событиях, синхронных по отношению к программе, например, чтобы показать, что контроллер достиг предельного значения
Сообщения, связанные с символами	Используются для сообщения о событиях, которые не зависят от программы, например, установка контролируемого переключателя
Сообщения, определенные пользователем	Используются для сообщения о диагностических событиях в диагностическом буфере с каждым вызовом SFC

14.1.4 Компоненты SIMATIC

Обзор

На следующем рисунке показан обзор компонентов SIMATIC, вовлеченных в проектирование и отображение сообщений.



14.1.5 Части сообщения

Как сообщение отображается, зависит от метода обмена сообщениями, используемого блока сообщений и устройства отображения.

В следующей таблице перечислены возможные составные части сообщения:

Часть	Описание
Метка времени	Генерируется в программируемом контроллере при возникновении события, которому соответствует сообщение
Состояние сообщения	Возможны следующие состояния: прибытие, убытие, убытие без подтверждения, убытие с подтверждением
Присоединенное значение	Некоторым сообщениям может быть поставлена в соответствие величина, связанная с процессом, которая может быть оценена используемым блоком сообщений
Образ	Если система терпит крах, то возникшие сообщения могут быть последовательно отображены на станции оператора
Номер сообщения	Уникальный для всего проекта номер, который назначается системой и идентифицирует сообщение
Текст сообщения	Проектируется пользователем

Пример

Следующий пример показывает аварийное сообщение на панели оператора.

14.1.6 Присвоение номера сообщения

Сообщения идентифицируются по номеру, уникальному во всем проекте. Чтобы добиться этого, каждой из программ STEP 7 выделяется диапазон номеров внутри всего имеющегося в распоряжении диапазона (от 1 до 2097151). Если при копировании программы возникает конфликт, т. е. если те же самые номера сообщений уже были назначены в целевом диапазоне, то новой программе должен быть выделен новый диапазон номеров. Если такая ситуация возникает, то STEP 7 автоматически открывает диалоговое окно, в котором вы можете указать новый диапазон номеров.

Вы можете также установить или изменить диапазон номеров для программы S7 с помощью команды меню **Edit > Special Object Properties > Message Numbers [Редактировать > Специальные свойства объекта > Номера сообщений]**.

14.2 Назначение и редактирование сообщений, связанных с блоками

14.2.1 Назначение и редактирование сообщений, связанных с блоками

Сообщения, связанные с блоками, назначаются блоку (FB). Для создания сообщения, связанного с блоком, в качестве блоков сообщений можно использовать системные функциональные блоки (SFB) и системные функции (SFC).

14.2.2 Какие блоки сообщений имеются в распоряжении?

У вас есть выбор между следующими блоками сообщений, каждый из которых содержит запрограммированную функцию сообщений:

- SFB33 ”ALARM”
- SFB34 ”ALARM_8”
- SFB35 ”ALARM_8P”
- SFB36 ”NOTIFY”
- SFC18 ”ALARM_S” и SFC17 ”ALARM_SQ”
- SFB37 ”AR_SEND” (для передачи архивов)

Более подробную информацию вы найдете в оперативной помощи по блокам.

Когда какой блок сообщений использовать?

Следующая таблица поможет вам принять решение, какой блок сообщений следует выбрать для вашей конкретной задачи. Выбор блока сообщений зависит от следующего:

- от количества каналов, доступных в блоке и, следовательно, от количества сигналов, наблюдаемых при вызове блока
- должны ли сообщения квитироваться
- от возможности назначения присоединенных значений
- от используемого устройства отображения

Блок сообщений	Каналы	Квитирование	Присоединенные значения	Отображение WinCC	Отображение ProTool	Отображение сообщений CPU /статуса S7	ПЛК	Примечания
ALARM SFB33	1	Возможно	до 10	да	нет	нет	S7-400	Посылает сообщение для каждого приходящего и уходящего фронта
ALARM_8 SFB34	8	Возможно	нет	да	нет	нет	S7-400	Посылает сообщение для каждого приходящего и уходящего фронта одного или более сигналов
ALARM_8P SFB35	8	Возможно	до 10	да	нет	нет	S7-400	Как ALARM_8
NOTIFY SFB36	1	Нет	до 10	да	нет	нет	S7-400	Как ALARM
AR_SEND SFB37	1			да	нет	нет	S7-400	Используется для передачи архива
ALARM_SQ SFC17	1	Возможно	1	да	да	да	S7-300/S7-400	Сообщение генерируется не изменением фронта, а каждым вызовом SFC
ALARM_S SFC18	1	Нет	1	да	да	да	S7-300/S7-400	Как ALARM_SQ

14.2.3 Формальные параметры, системные атрибуты и блоки сообщений

Формальные параметры как входы для номеров сообщений

Для каждого сообщения или группы сообщений вам нужен в вашей программе формальный параметр, который определяется как входная переменная в таблице описания переменных вашей программы. Затем этот формальный параметр используется как вход для номера сообщения и образует основу сообщения.

Как снабдить формальные параметры системными атрибутами

В качестве предпосылки для начала проектирования сообщения вы должны в первую очередь снабдить формальные параметры системными атрибутами следующим образом:

1. Добавьте следующие системные атрибуты для параметров: "S7_server" и "S7_a_type"
2. Присвойте значения системным атрибутам в соответствии с блоками сообщений, которые вы вызвали в своем программном коде. Значение для атрибута "S7_server" всегда "alarm_archiv", значение для атрибута "S7_a_type" соответствует вызываемому блоку сообщений.

Системные атрибуты и соответствующие блоки сообщений

Сами блоки сообщений не отображаются как объекты в администраторе сообщений; вместо этого отображение содержит соответствующие значения системного атрибута "S7_a_type". Эти значения имеют такие же имена, как и блоки сообщений, существующие в виде SFB и SFC (исключение: "alarm_s").

S7_a_type	Блок сообщений	Описание	Свойства
Alarm_8	ALARM_8	SFB34	8 каналов, может быть квитирован, нет присоединенных значений
Alarm_8p	ALARM_8P	SFB35	8 каналов, может быть квитирован, до 10 присоединенных значений на канал
Notify	NOTIFY	SFB36	1 канал, не может быть квитирован, до 10 присоединенных значений
Alarm	ALARM	SFB33	1 канал, может быть квитирован, до 10 присоединенных значений
Alarm_s	ALARM_S	SFC18	1 канал, не может быть квитирован, не более одного присоединенного значения
Alarm_s	ALARM_SQ	SFC17	1 канал, может быть квитирован, не более одного присоединенного значения
ar_send	AR_SEND	SFB37	Используется для передачи архива

Более подробную информацию вы найдете в оперативной помощи по системным атрибутам.

Системные атрибуты назначаются автоматически, если блоками сообщений, которые вы используете в своей программе, являются SFB или FB с

соответствующими системными атрибутами, и они вызываются как мультиэкземпляры.

14.2.4 Шаблоны сообщений и сообщения

Проектирование сообщений позволяет использовать различные процедуры для создания шаблона сообщений или сообщения. Это зависит от блока, вызывающего сообщение, через который вы получаете доступ к проектированию сообщения.

Блок, вызывающий сообщение, может быть функциональным блоком (FB) или экземплярным блоком данных.

- Используя FB, вы можете создать шаблон, с помощью которого могут создаваться сообщения. Все записи, которые вы делаете для шаблона сообщений, вводятся в сообщения автоматически. Если вы ставите в соответствие этому функциональному блоку экземплярный блок данных, то сообщения для экземплярного блока данных генерируются автоматически в соответствии с шаблоном сообщений и назначенными номерами сообщений.
- Для экземплярного блока данных вы можете для конкретного экземпляра изменять сообщения, сгенерированные на основе этого шаблона сообщений.

Очевидная разница здесь состоит в том, что номера сообщений назначаются сообщениям, но не шаблонам сообщений.

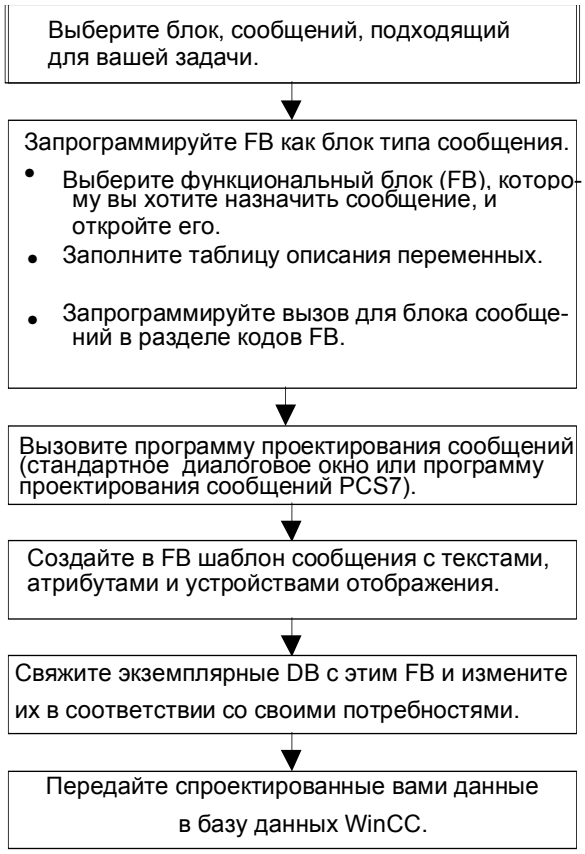
Блокировка данных для шаблона сообщений

Проектирование сообщений дает возможность вводить тексты и атрибуты для сообщений, зависящих от событий. Вы можете также указывать, например, как вы хотите представлять сообщения на конкретных устройствах отображения. Для облегчения генерирования сообщений вы можете работать с шаблонами сообщений.

- При вводе данных (атрибутов и текстов) вы можете указать, должны они быть заблокированы или нет. В случае блокировки атрибутов вслед за окном ввода добавляется символ ключа. Заблокированные тексты имеют метку в столбце "Locked [Заблокирован]".
- В случае шаблона сообщений "locked data [заблокированные данные]" вы не можете производить изменения в сообщениях, относящихся к конкретному экземпляру. Данные только отображаются.
- Если вам нужно произвести изменения, то вы должны вернуться к шаблону сообщений, удалить там блокировку и выполнить изменения там. Изменение не действует на экземпляры, сгенерированные до этого изменения.

14.2.5 Создание сообщений, связанных с блоками

Основная последовательность действий



Программирование блоков, вызывающих сообщения (FB)

1. В SIMATIC Manager выберите функциональный блок (FB), для которого вы хотите сгенерировать сообщение, связанное с блоком, и откройте этот блок двойным щелчком.
   Результат: Выбранный блок открывается и отображается в окне ”LAD/STL/FBD (KOP/AWL/FUP)”.
2. Заполните таблицу описания переменных. Для каждого блока сообщений, вызываемого в данном функциональном блоке, вы должны описать переменные в вызывающем функциональном блоке.

   В таблице описания переменных введите следующие переменные в столбце "Declaration [Описание]":

- Для типа описания ”in” введите символическое имя для входа блока сообщений, например, ”Mess01” (для входа сообщений 01), и тип (должен быть ”DWORD” без начального значения).
- Для типа описания ”stat” введите символическое имя для подлежащего вызову блока сообщений, например, ”alarm”, и соответствующий тип, здесь ”SFB33”.

3. В разделе кодов функционального блока вставьте вызов для выбранного блока сообщений, здесь ”CALL alarm”, и закончите ввод клавишей RETURN.

   Результат: Входные переменные для вызываемого блока сообщений (здесь SFB33) отображаются в разделе кодов функционального блока.
4. Присвойте символическое имя, которое вы назначили на шаге 2 входу блока сообщений, здесь "Mess01", переменной ”EV_ID” и подтвердите, что для проектирования сообщения должны быть использованы системные атрибуты.

   Результат: В столбце "Name [Имя] должна появиться пометка, если этот столбец не выбран. После этого выбранный блок устанавливается как блок, содержащий сообщения. Требуемые системные атрибуты (например, S7_server и S7_a_type) и соответствующие значения назначаются автоматически.
5. Повторите шаги со 2-го по 4-й для всех обращений к блокам сообщений в этом функциональном блоке.
6. Сохраните блок с помощью команды меню **File > Save [Файл > Сохранить]**.

Открытие диалогового окна для проектирования сообщений

- Выберите в SIMATIC Manager команду меню **Edit > Special Object Properties > Message [Редактировать > Специальные свойства объекта > Сообщение]**.
  Результат: Открывается диалоговое окно для проектирования сообщений STEP 7 (стандартное диалоговое окно). Информацию об открытии функции проектирования сообщений PCS7 можно найти в разделе Проектирование сообщений PCS7.

Создание шаблона сообщений

1. Дважды щелкните выведенный на экран блок сообщений и введите требуемые атрибуты сообщения и его текст в закладках ”Attributes [Атрибуты]” и ”Text [Текст]”.
   Если вы выбрали многоканальный блок сообщений (например, ”ALARM_8”), то вы можете назначить свой собственный текст сообщения каждому субномеру. Атрибуты относятся ко всем субномерам.
2. Назначьте шаблону сообщений требуемые устройства отображения, щелкнув на кнопке "New Device [Новое устройство]" и выбрав требуемые устройства отображения в диалоговом окне "Add Display Device [Добавить устройство отображения]".

В следующих страницах с закладками введите требуемые тексты и атрибуты для устройств отображения. Выйдите из диалогового окна с помощью "ОК" и закройте окно ”LAD/STL/FBD (KOP/AWL/FUP)".

Замечание

При редактировании текстов и атрибутов, относящихся к устройству отображения, прочитайте, пожалуйста, документацию, поставляемую с вашим устройством отображения.

Создание экземплярных блоков данных

1. Когда вы создали шаблон сообщений, вы можете связать с ним экземплярные блоки данных и отредактировать для этих блоков данных сообщения, относящиеся к экземплярам. Чтобы сделать это, откройте в SIMATIC Manager блок, который должен вызывать ваш предварительно спроектированный функциональный блок, например, ”OB1”, дважды щелкнув на нем. В открывшемся разделе кодов OB введите вызов (”CALL”), имя и номер подлежащего вызову FB и экземплярного DB, который вы хотите связать с этим FB. Подтвердите ваш ввод нажатием RETURN.

   Пример: Введите ”CALL FB1, DB1”. Если DB1 еще не существует, подтвердите приглашение создать экземплярный DB, нажав ”Yes [Да]”.

   Результат: Экземплярный DB создан. В разделе кодов OB отображаются входные переменные соответствующих FB, здесь, например, ”Mess01”, и номер сообщения, выделенный системой, здесь ”1”.
2. Сохраните OB с помощью команды меню **File > Save [Файл > Сохранить]** и закройте окно ”LAD/STL/FBD (KOP/AWL/FUP)”.

Редактирование сообщений

1. В SIMATIC Manager выделите созданный экземплярный DB, например, ”DB1”, и выберите команду меню **Edit > Special Object Properties > Message [Редактировать > Специальные свойства объекта > Сообщение]**, чтобы открыть диалоговое окно для проектирования сообщений.
   Результат: Открывается диалоговое окно "Message Configuration [Проектирование сообщения]" и отображается выбранный экземплярный DB с номером сообщения, выделенным системой.
2. Введите требуемые изменения для соответствующего экземплярного DB в нужные закладки и, если хотите, добавьте другое устройство отображения. Выйдите из диалогового окна, нажав ”OK”.
   Результат: Проектирование сообщения для выбранного экземплярного DB завершено.

Передача спроектированных данных

- Передайте спроектированные данные в базу данных WinCC (через конфигурацию соединения PLC-OS) или в базу данных ProTool.

14.2.6 Проектирование сообщений PCS7

Для редактирования шаблонов сообщений и сообщений, подлежащих выводу на устройства отображения WinCC, функция проектирования сообщений PCS7 в STEP 7 предоставляет удобный для пользователя метод:

- упрощения конфигурирования устройств отображения (выполняется автоматически)
- упрощения ввода атрибутов и текстов для сообщений
- гарантирования стандартизации сообщений.

Открытие функции проектирования сообщений PCS7

1. В SIMATIC Manager выделите блок (FB или DB), текст сообщения которого вы хотите редактировать, и используйте команду меню **Edit > Object Properties [Редактировать > Свойства объекта]**, чтобы открыть диалоговое окно для ввода системных атрибутов.
2. В появившейся таблице введите следующий системный атрибут:
   - Атрибут: ”S7_alarm_ui” и значение: ”1”.

Замечание
При вводе системных атрибутов производится проверка синтаксиса, и неправильные записи отмечаются красным цветом.

3. Выйдите из диалогового окна с помощью ”OK”.
4. Выберите команду меню **Edit > Special Object Properties > Message [Редактировать > Специальные свойства объекта > Сообщение]**.

   Результат: Открывается диалоговое окно "PCS7 Message Configuration [Проектирование сообщений PCS7]".

Редактирование шаблонов сообщений

1. В SIMATIC Manager выберите FB, тексты сообщений которого вы хотите редактировать, и откройте диалоговое окно для проектирования сообщений PCS7.

   Результат: В диалоговом окне появляется закладка для каждого блока сообщений, для которого вы описали в FB переменную.
2. Заполните текстовые окна для разделов сообщения "Origin [Происхождение]", "OS area [Область OS]" и "Batch ID [Идентификатор пакета]".
3. Введите класс сообщения и текст события для всех событий используемых блоков сообщений и укажите, должно ли каждое событие квитироваться индивидуально.
4. Для разделов сообщения, которые применимы ко всем экземплярам и не должны изменяться, щелкните на триггерной кнопке "Locked [Заблокирован]".

Редактирование сообщений

1. В SIMATIC Manager выберите экземплярный DB, текст сообщения которого вы хотите редактировать, и откройте диалоговое окно для проектирования сообщений PCS7.
2. Не изменяйте относящиеся к экземпляру незаблокированные разделы сообщения.

14.3 Назначение и редактирование сообщений, связанных с символами

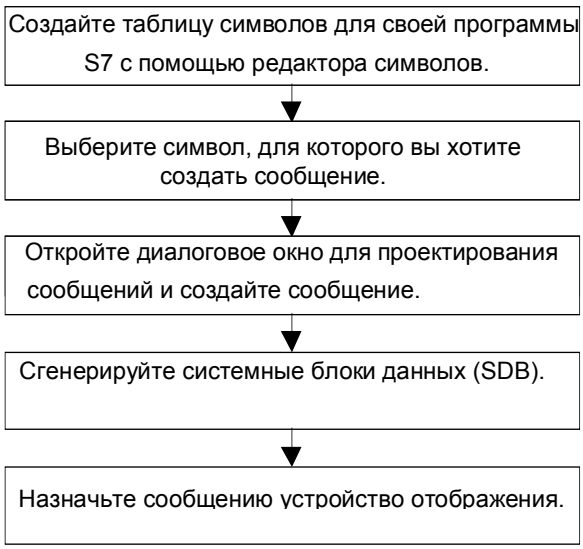
14.3.1 Назначение и редактирование сообщений, связанных с символами

Сообщения, связанные с символами (SCAN), назначаются непосредственно сигналу в таблице символов. Разрешенными сигналами являются все булевы операнды: входы (I), выходы (Q) и меркеры (M). С помощью функции проектирования сообщений вы можете назначать этим сигналам различные атрибуты, тексты сообщений и до 10 присоединенных значений. Отбор сигналов в таблице символов можно облегчить установкой фильтров.

С помощью сообщений, связанных с символами, вы можете просматривать состояние сигнала через заранее определенные промежутки времени, чтобы определить, изменился он или нет.

Замечание
Промежуток времени зависит от используемого CPU.

Основная последовательность действий



Во время обработки сигналы, для которых вы спроектировали сообщения, проверяются асинхронно по отношению к вашей программе. Контроль происходит через запроектированные промежутки времени. Сообщения выводятся на назначенных устройствах отображения.

14.4 Создание и редактирование диагностических сообщений, определенных пользователем

14.4.1 Создание и редактирование диагностических сообщений, определенных пользователем

С помощью этой функции вы можете сделать пользовательскую запись в диагностический буфер и послать соответствующее сообщение, которое создается в приложении для проектирования сообщений. Диагностические сообщения, определенные пользователем, создаются с помощью системной функции SFC52 (WR_USMSG), которая используется как блок сообщений. Вы должны вставить вызов SFC52 в свою пользовательскую программу и выделить ей идентификатор события.

В отличие от сообщений, связанных с блоками или с символами, диагностические сообщения, определенные пользователем, могут отображаться только на устройстве программирования. Поэтому вы не можете назначить этим сообщениям устройство отображения в приложении для проектирования сообщений.

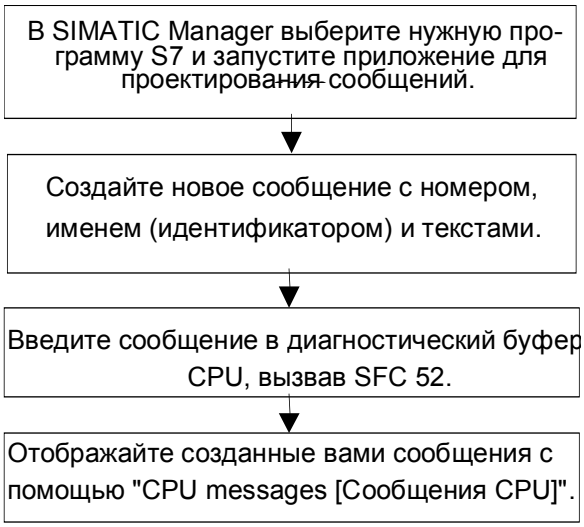
Предпосылки

Перед созданием диагностического сообщения, определенного пользователем, вы должны:

- создать проект в SIMATIC Manager
- создать в этом проекте программу S7, которой вы хотите назначить сообщение

Основная последовательность действий

Для создания и отображения диагностического сообщения, определенного пользователем, действуйте следующим образом:



14.5 Перевод и редактирование текстов пользователя

14.5.1 Перевод и редактирование текстов пользователя

Обзор

Тексты, выводимые на устройство отображения во время процесса редактирования, обычно вводились на том же языке, который использовался при программировании решения задачи автоматизации.

Часто может оказаться так, что оператор, который должен реагировать на сообщения, выводимые на устройство отображения, не говорит на этом языке. Этому пользователю нужны тексты на его родном языке, чтобы обеспечить спокойную, беспроблемную обработку и быструю реакцию на сообщения, выводимые системой.

STEP 7 позволяет переводить все пользовательские тексты на любой требуемый язык. Единственной предпосылкой для этого является предварительная установка языка в вашем проекте (команда меню: **Options > Language for Display Devices [Параметры > Язык для устройств отображения]** в SIMATIC Manager). Количество доступных языков определяется при установке Windows 95/98/NT (свойство системы).

Таким образом, вы можете быть уверены, что любой пользователь, столкнувшийся впоследствии с таким сообщением, получит его отображение на подходящем языке. Эта системная характеристика существенно увеличивает безопасность и правильность обработки.

Списки текстов пользователя

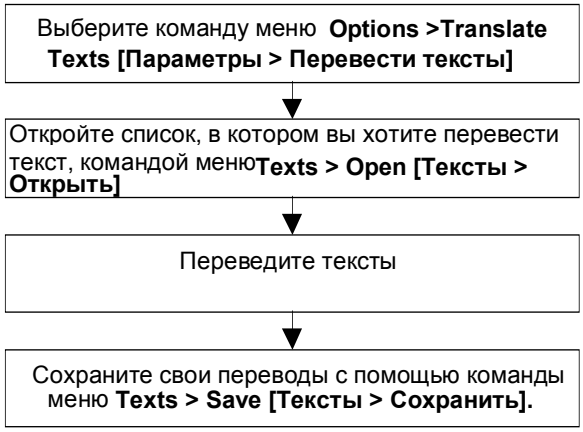
Списки текстов пользователя могут быть созданы для всего проекта, для программ S7, для папки блоков или отдельных блоков или для таблицы символов, если сообщения проектируются в этих объектах. Они содержат все тексты и сообщения для проекта, которые, например, могут быть выведены на устройства отображения. Может быть несколько списков пользовательских текстов для проекта, которые вы можете перевести на требуемые языки.

Вы можете выбирать языки, доступные в проекте (команда меню: **Options > Language for Display Devices [Параметры > Язык для устройств отображения]**). Вы можете также добавлять или удалять языки на последующих этапах.

При открытии списка текстов пользователя (команда меню: **Options > Translate Texts [Параметры > Перевести тексты]**) на экран выводится таблица, в которой каждому языку выделен один столбец. Первый столбец всегда отображает язык, установленный по умолчанию.

Основная последовательность действий

Убедитесь, что вы установили языки, на которые вы хотите перевести тексты пользователя, в SIMATIC Manager с помощью команды меню **Options > Language for Display Devices [Параметры > Язык для устройств отображения]**.



Экспорт и импорт текстов пользователя

Вы можете также переводить или редактировать пользовательские тексты, созданные в STEP 7, вне STEP 7. Для этого экспортируйте выведенный на экран список тестов пользователя в текстовый файл, который вы можете редактировать с помощью редактора ASCII или табличного редактора, такого как Microsoft Excel. У вас есть выбор из форматов файлов *.TXT и *.CSV. Затем тексты импортируются обратно в STEP 7.

Тексты пользователя могут импортироваться только в ту часть проекта, из которой они были экспортированы.

Замечания

При редактировании экспортированных текстов позаботьтесь о том, чтобы не переписать какую-нибудь другую информацию (идентификаторы языков или имена путей).

Редактируйте редактором ASCII только строки, начинающиеся с "T-ID=", и не удаляйте никаких точек с запятой.

Табличным редактором никогда не редактируйте первый столбец или первые две строки.

Пример редактирования с помощью редактора ASCII:
(редактироваться могут только тексты, изображенные курсивом)

INTERN;GERMAN (GERMANY);ENGLISH (USA);ITALIAN (ITALY)
;1031;1033;1040

T-COUNTER=15;;;

D:\SIEMENS\STEP7\S7proj\Project1;;;

TL-ID=9;;;
Project1 \ S7-Program(1) \ S7-Program(1) \ S7-Program(1);;;
T-ID=1;*Test point 21*;*Test point 21*;*Test point 21*
T-ID=2;*Test loop end*;*Test loop end*;*Test loop end*

Пример редактирования с помощью табличного редактора:
(редактироваться могут только тексты, изображенные курсивом)

INTERN	GERMAN (GERMANY)	ENGLISH (USA)
	1031	1033
T-COUNTER=15		
D:\SIEMENS\STEP7\S7proj\Project1		
TL-ID=9		
Project1 \ S7-Program(1) \ S7-Program(1) \ S7-Program(1)		
T-ID=1	*Test point 21*	*Test point 21*
T-ID=2	*Test loop end*	*Test loop end*

14.6 Передача данных проектирования сообщений в программируемый контроллер

14.6.1 Передача данных проектирования в программируемый контроллер

Обзор

С помощью программы передачи PLC-OS Engineering (Проектирование соединения ПЛК - станция оператора) сгенерированные данные, полученные при проектировании сообщений, передаются в базу данных WinCC.

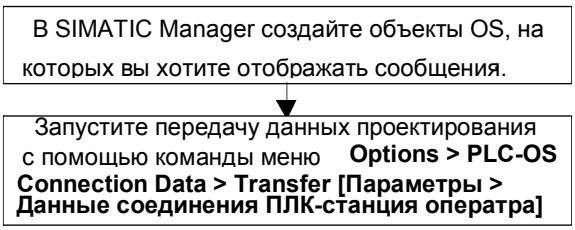
У вас есть выбор из ряда параметров передачи. Например, вы можете выбрать сравнение адреса и текста, чтобы гарантировать, что текущие данные переданы.

Предпосылки

Перед началом передачи должны быть выполнены следующие требования:

- Вы инсталлировали программу установки инструментального средства для проектирования соединения ПЛК - станция оператора
- Вы сгенерировали проектные данные для создания сообщений.

Основная последовательность действий



14.7 Отображение сообщений CPU и диагностических сообщений, определенных пользователем

14.7.1 Отображение сообщений CPU и диагностических сообщений, определенных пользователем

С помощью функции ”CPU Messages [Сообщения CPU]" вы можете отображать асинхронные сообщения о событиях, связанных с системными ошибками, и диагностические сообщения, определенные пользователем.

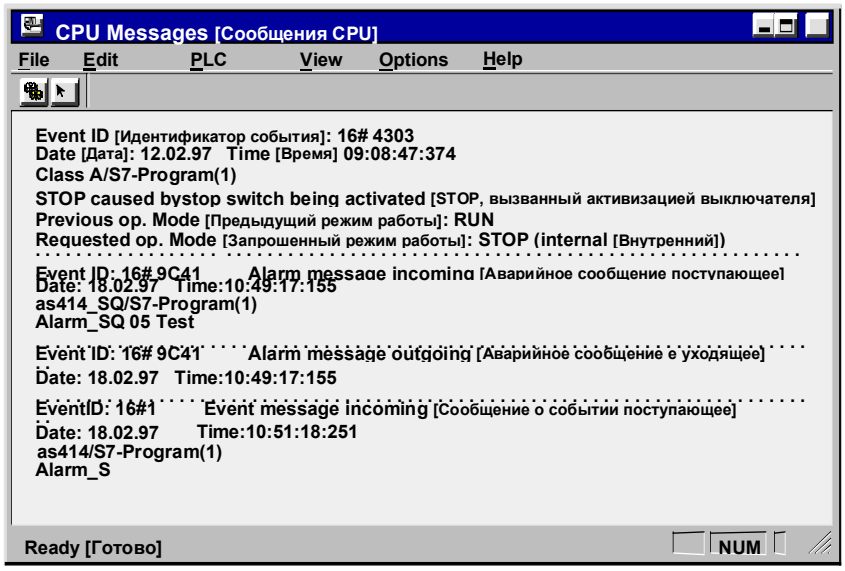
Вы можете также запускать приложение для проектирования сообщений из приложения CPU Messages [Сообщения CPU] с помощью команды меню **Options > Configure Messages [Параметры > Проектирование сообщений]** и создавать диагностические сообщения, определенные пользователем. Предпосылкой для этого является предварительный запуск приложения CPU Messages [Сообщения CPU] через проект, открытый online.

Отображение параметров

С помощью функции "CPU Messages [Сообщения CPU]" вы можете принять решение, отображать ли и, если отображать, то каким образом, оперативные (online) сообщения для выбранных CPU.

- "**Top [Сверху]**": Окно, содержащее сообщения CPU, появляется на переднем плане. Оно всплывает каждый раз, как поступает новое сообщение.
- ”**Background [Задний план]**": Сообщения CPU получаются в фоновом режиме. При получении новых сообщений окно остается на заднем плане и может перемещаться на передний план в случае необходимости.
- ”**Ignore [Игнорировать]**": Сообщения CPU **не** отображаются и, в отличие от двух других методов, **не** архивируются.

В окне ”CPU Messages [Сообщения CPU]" вы можете просматривать сообщения, находящиеся в архиве. Некоторые примеры показаны на следующем рисунке:



Пояснения к рисунку: File – файл; Edit – редактировать; PLC – ПЛК; View – вид; Options – параметры; Help – помощь

Квитируемые сообщения (ALARM_SQ) отображаются полужирным шрифтом и могут быть квитированы с помощью команды меню **Edit > Acknowledge CPU Message [Редактировать > Квитировать сообщение CPU]**.

Функция архивирования

Имеется архив для создания резервных копий сообщений, в котором может храниться от 40 до 2000 сообщений CPU. Если установленный размер архива превышен, то из него удаляется самое старое сообщение, чтобы освободить место для нового сообщения.

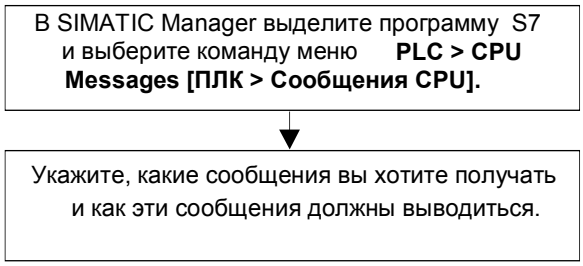
Обновление сообщений ALARM_S/ALARM_SQ

При обновлении сообщений ALARM_S/ALARM_SQ все непосланные и/или неквитированные сообщения снова вводятся в архив. Сообщения обновляются:

- если производится перезапуск в модуле, к которому относятся эти сообщения (не холодный рестарт)
- если вы щелкнете на опции "A" для ALARM_S/ALARM_SQ в диалоговом окне "Customize [Настройка]" при настройке сообщений CPU.

Основная последовательность действий

Для настройки сообщений CPU для выбранных модулей:



14.7.2 Настройка сообщений CPU

Чтобы настроить сообщения CPU для выбранных модулей, действуйте следующим образом:

1. В SIMATIC Manager через проект, открытый online, запустите приложение CPU Messages [Сообщения CPU]. Для этого выберите программу S7 в режиме online и вызовите для выбранного CPU приложение CPU Messages [Сообщения CPU] с помощью команды меню **PLC > CPU Messages [ПЛК > Сообщения CPU]**.
   Результат: Появляется диалоговое окно "Customize [Настройка]" со списком зарегистрированных CPU.
2. Вы можете расширить список зарегистрированных CPU, повторяя шаг 1 для других программ и интерфейсов.
3. Щелкните на триггерной кнопке перед записями списка и укажите, какие сообщения следует принимать для модуля:

   A: активизирует сообщения о событиях и неисправностях (ALARM_SQ (SFC 17) и ALARM_S (SFC 18))

   W: активизирует пользовательские и системные диагностические сообщения.

- Установите режим, в котором вы хотите отображать поступающие сообщения:

4. Установите размер архива.
5. Завершив настройку, закройте диалоговое окно.

   Результат: Как только происходят вышеуказанные сообщения, они записываются в архив сообщений и отображаются в выбранной вами форме.

Замечание
CPU, для которых вы вызывали в SIMATIC Manager команду меню **PLC > CPU Messages [ПЛК > Сообщения CPU]**, вносятся в список зарегистрированных модулей в диалоговом окне "Customize [Настройка]". Записи в списке сохраняются до тех пор, пока они не будут удалены в диалоговом окне "Customize [Настройка]".

14.7.3 Отображение сохраненных сообщений CPU

Сообщения CPU всегда записываются в архив, если только вы не установили в диалоговом окне "Customize [Настройка]" параметр "Ignore [Игнорировать]". Все заархивированные сообщения всегда отображаются.

15 Управление и наблюдение за переменными

15.1 Проектирование переменных для управления и наблюдения со стороны оператора

Обзор

STEP 7 предоставляет удобный для пользователя метод управления и наблюдения за переменными в вашем процессе или программируемом контроллере с использованием WinCC.

Преимущество этого метода перед ранее использовавшимися состоит в том, что вам более не нужно проектировать данные отдельно для каждой станции оператора (OS), вы просто проектируете их один раз в STEP 7. При проектировании с помощью STEP 7 вы можете передать сгенерированные данные в базу данных WinCC с помощью программы передачи PLC-OS Engineering [Проектирование связи ПЛК - станция оператора] (часть пакета программ "Process Control System PCS7 [Система управления процессами PCS7]"). При этом проверяется непротиворечивость данных и их совместимость с системой отображения. WinCC использует эти данные в блоках переменных и в графических объектах.

С помощью STEP 7 вы можете проектировать или изменять атрибуты управления и наблюдения со стороны оператора для следующих переменных:

- входные, выходные и проходные параметры в функциональных блоках
- меркеры и сигналы ввода/вывода
- параметры для блоков CFC в схемах CFC

Основная последовательность действий

Последовательность действий при проектировании переменных для управления и наблюдения со стороны оператора зависит от выбора языка программирования/проектирования и типа переменных, подлежащих управлению и наблюдению. Однако основная последовательность действий всегда включает в себя следующие шаги:

1. Назначьте системные атрибуты для управления и наблюдения со стороны оператора параметрам функционального блока или символам в таблице символов.
   Этот шаг не требуется в CFC, так как вы берете уже готовые блоки из библиотеки.
2. В диалоговом окне снабдите переменные, подлежащие управлению и наблюдению, необходимыми атрибутами, такими как предельные значения, заменяющие значения и свойства протоколирования.
3. Передайте данные, сгенерированные с помощью STEP 7, на свою систему отображения (WinCC) с помощью инструментального средства для проектирования соединений ПЛК – станция оператора PLC-OS Engineering.

Соглашения о наименованиях

Чтобы данные проектирования для WinCC могли быть сохранены и переданы, они хранятся под уникальным именем, автоматически назначаемым STEP 7. Имена переменных для управления и наблюдения со стороны оператора, схем CFC и программ S7 образуют часть этого имени и поэтому являются предметом определенных соглашений:

- Имена программ S7 в проекте S7 должны быть уникальными (различные станции не могут содержать программы S7 с одинаковыми именами).
- Имена переменных, программ S7 и схем CFC не могут содержать символов подчеркивания, пробелов и следующих специальных символов: [‘] [.] [%] [-] [/] [*] [+].

15.2 Установление атрибутов для управления и наблюдения со стороны оператора в случае списка команд, контактного плана и функционального плана

15.2.1 Установление атрибутов для управления и наблюдения со стороны оператора в случае списка команд, контактного плана и функционального плана

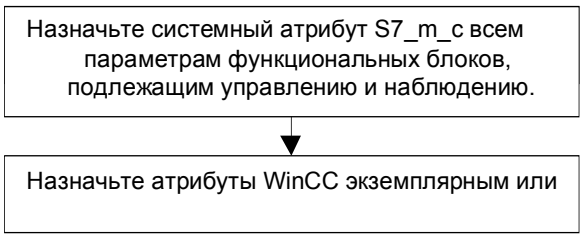
Обзор

Используя описанную ниже процедуру, вы можете в своей пользовательской программе формировать параметры функционального блока, пригодные для управления и наблюдения со стороны оператора, и назначать требуемые атрибуты O, C и M соответствующим экземплярным или совместно используемым DB.

Предпосылки

Вы должны были создать проект STEP 7, программу S7 и функциональный блок.

Основная последовательность действий



15.3 Установление атрибутов для управления и наблюдения со стороны оператора через таблицу символов

15.3.1 Установление атрибутов для управления и наблюдения со стороны оператора через таблицу символов

Обзор

С помощью описанной ниже процедуры вы можете проектировать, независимо от используемого языка программирования, следующие переменные:

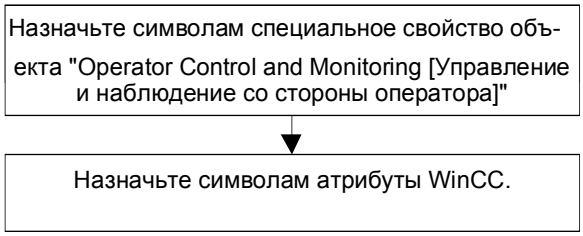
- меркеры
- сигналы ввода/вывода

Предпосылки

До того как вы начнете, должны быть выполнены следующие требования:

- Вы создали проект в SIMATIC Manager.
- В этом проекте должна существовать программа S7 с таблицей символов.
- Таблица символов должна быть открыта.

Основная последовательность действий



15.4 Изменение атрибутов управления и наблюдения со стороны оператора в случае CFC

15.4.1 Изменение атрибутов управления и наблюдения со стороны оператора в случае CFC

Обзор

При использовании CFC вы создаете свою пользовательскую программу, выбирая блоки, уже обладающие свойством управления и наблюдения со стороны оператора, из библиотеки и помещая и связывая их между собой в схеме.

Предпосылки

Вы вставили программу S7 в проект STEP 7, создали схему CFC и разместили в ней блоки.

Основная последовательность действий

Отредактируйте свойства объекта в блоках.

Замечание

Если вы используете блоки, которые вы создали сами и которым вы назначили системный атрибут S7_m_c, вы можете придать этим блокам свойства управления и наблюдения со стороны оператора, активизировав триггерную кнопку "Operator Control and Monitoring [Управление и наблюдение со стороны оператора]" в диалоговом окне с таким же названием (команда меню **Edit > Object Properties [Редактировать > Свойства объекта]**).

15.5 Передача данных проектирования в программируемый контроллер интерфейса с оператором

15.5.1 Передача данных проектирования в программируемый контроллер интерфейса с оператором

Обзор

С помощью программы передачи инструментального средства для проектирования соединений ПЛК – панель оператора PLC-OS Engineering сгенерированные данные для управления и наблюдения со стороны оператора передаются в базу данных WinCC.

У вас есть выбор из ряда различных параметров передачи. Например, вы можете выбрать сравнение адресов и текстов, чтобы гарантировать передачу текущих атрибутов WinCC.

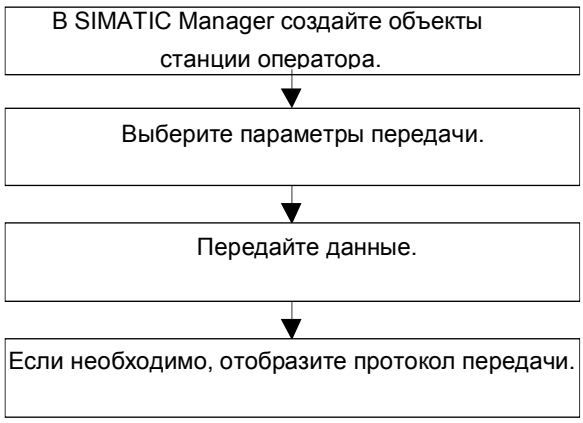
Предпосылки

До начала передачи должны быть выполнены следующие требования:

- Вы инсталлировали программу установки инструментального средства для проектирования соединения ПЛК - станция оператора (PLC-OS Engineering).
- Вы сгенерировали проектные данные для управления и наблюдения со стороны оператора.

Основная последовательность действий

Чтобы передать проектные данные для управления и наблюдения со стороны оператора в базу данных WinCC, действуйте следующим образом:



16 Установление соединения online и настройка CPU

16.1 Установление соединения online

16.1.1 Установление соединения online

Соединение online между устройством программирования и программируемым логическим контроллером необходимо для загрузки пользовательских программ S7 или блоков, считывания блоков из программируемого контроллера S7 в устройство программирования и для других операций:

- отладка программ пользователя
- отображение и изменение режима работы CPU
- отображение и установка времени и даты CPU
- отображение информации о модуле
- сравнение блоков online и offline
- диагностика аппаратуры

Для установления соединения online устройство программирования и программируемый логический контроллер должны быть соединены через надлежащий интерфейс (например, многоточечный интерфейс (MPI)). После этого вы можете получить доступ в контроллер через окно online в проекте или окно "Accessible Nodes [Доступные узлы]".

16.1.2 Установление соединения online через окно "Accessible Nodes [Доступные узлы]"

Этот тип доступа дает вам возможность обратиться к программируемому логическому контроллеру быстро, например, для тестирования. Вы можете получить доступ ко всем программируемым модулям в сети. Выбирайте этот метод, если на вашем устройстве программирования отсутствуют проектные данные о программируемых контроллерах.

Окно "Accessible Nodes [Доступные узлы]" открывается с помощью команды меню **PLC > Display Accessible Nodes [ПЛК > Отобразить доступные узлы]**. В объекте "Accessible Nodes [Доступные узлы]" все программируемые модули, доступные в сети, отображаются вместе с их адресами.

Узлы, которые не могут программироваться с помощью STEP 7 (например, устройства программирования или панели оператора), тоже могут быть отображены.

16.1.3 Установление соединения online через окно online проекта

Выберите этот метод, если вы сконфигурировали программируемый контроллер в проекте на своем устройстве программирования/PC. Вы можете открыть окно online в SIMATIC Manager с помощью команды меню **View > Online [Вид > Online]**. Она отображает данные проекта в программируемом контроллере (в отличие от окна offline, отображающего данные проекта в устройстве программирования/PC). Окно online показывает данные в программируемом контроллере как для программы S7, так и для программы M7.

Это представление проекта используется для функций, включающих в себя обращение к программируемому контроллеру. Некоторые функции в меню "PLC [ПЛК]" в SIMATIC Manager могут быть активизированы только в окне online, но не в окне offline.

Имеются следующие два типа доступа:

- **Доступ с конфигурированием аппаратуры**
  Это значит, что вы можете обращаться только к модулям, которые были сконфигурированы в режиме offline. К каким модулям вы имеете доступ online, определяется адресом MPI, установленным при конфигурировании программируемых модулей.
- **Доступ без конфигурирования аппаратуры**
  Предпосылкой для этого является существование программы S7 или M7, созданной независимо от аппаратуры (т.е. она находится непосредственно проектом). К каким модулям вы можете получить доступ в режиме online, определяется здесь указанием соответствующего адреса MPI в свойствах объекта программы S7/M7.

Доступ через окно online объединяет данные в программируемой системе управления с соответствующими данными в устройстве программирования. Если, например, вы открываете блок S7 под проектом online, то отображение формируется следующим образом:

- раздел кодов блока из CPU в программируемом логическом контроллере S7, а
- комментарии и символы из базы данных в устройстве программирования (при условии, что они существуют offline). Когда вы открываете блоки непосредственно в подключенном CPU при отсутствии структуры проекта, они отображаются так, как они найдены в CPU, т. е. без символов и комментариев.

16.1.4 Защита паролем для доступа к программируемым контроллерам

Используя пароли, вы можете:

- защитить программу пользователя в CPU и ее данные от несанкционированного изменения (защита от записи)
- защитить программные ноу-хау в своей пользовательской программе (защита от чтения)
- воспрепятствовать в режиме online действиям, которые могли бы помешать реализации процесса

Вы можете защитить модуль с помощью пароля только в том случае, если модуль поддерживает эту функцию.

Если вы хотите защитить модуль паролем, то вы должны определить уровень защиты и установить пароль в процессе назначения параметров модуля, а затем загрузить измененные параметры в модуль.

Если вам нужно ввести пароль, чтобы выполнить какую-либо функцию в режиме online, то на экране появляется диалоговое окно "Enter Password [Введите пароль]". Если вы вводите правильный пароль, то вы получаете права доступа к модулям, для которых был установлен конкретный уровень защиты во время назначения параметров. После этого вы можете установить в режиме online соединение с защищенным модулем и выполнить функцию, относящуюся к этому уровню защиты.

С помощью команды меню **PLC > Access Rights [ПЛК > Права доступа]** вы можете вызвать диалоговое окно "Enter Password [Введите пароль]" непосредственно. В этом диалоговом окне вы можете указать, что введенный пароль должен также использоваться для любого будущего доступа к защищенным модулям. Тогда диалоговое окно будет появляться лишь при вводе неправильного пароля.

Параметр CPU	Примечания
Test operation/process operation [Режим тестирования/режим обработки]	Может быть установлен в закладке "Protection [Защита]". В режиме обработки (process operation) тестовые функции, такие как статус программы или управление/наблюдение переменных, ограничены так, что установленное увеличение допустимого времени цикла опроса не превышается. Это значит, например, что в статусе программы не разрешается установка условий вызова, а отображение статуса программного цикла прерывается в точке возврата. В этом режиме не может быть применено тестирование с использованием контрольных точек и пошагового выполнения программы. В режиме тестирования (test operation) все тестовые функции, выполняемые через устройство программирования/PC, могут использоваться без ограничений, даже если они вызывают существенное увеличение времени цикла опроса.
Protection level [Уровень защиты]	Может быть установлен в закладке "Protection [Защита]". Вы можете сделать доступ к CPU на запись или чтение/запись зависящим от знания правильного пароля. Пароль устанавливается в этой закладке.

16.1.5 Обновление содержимого окна

Вам следует иметь в виду следующее:

- Изменения в окне проекта online, как результат действий пользователя (например, загрузка или удаление блоков), не отображаются автоматически во всех открытых окнах "Accessible Nodes [Доступные узлы]".
- Любые такие изменения в окне "Accessible Nodes [Доступные узлы]" не отображаются автоматически во всех открытых окнах проекта online.

Чтобы обновить отображение в параллельно открытом окне, вы должны сделать это явно (с помощью команды меню или функциональной клавиши F5).

16.2 Отображение и изменение режима работы

16.2.1 Отображение и изменение режима работы

С помощью этой функции вы можете, например, переключить CPU снова в RUN после исправления ошибки.

Отображение режима работы

1. Откройте свой проект и выделите программу S7/M7 или откройте окно "Accessible Nodes [Доступные узлы]" с помощью команды меню **PLC > Display Accessible Nodes [ПЛК > Отобразить доступные узлы]** и выберите узел ("MPI=...").
2. Выберите команду меню **PLC > Operating Mode [ПЛК > Режим работы]**.

Это диалоговое окно отображает текущий и последний режим работы и текущее положение переключателя режимов на модуле. Для модулей, у которых текущее положение переключателя не может быть отображено, выводится текст "Undefined [Не определено]".

Изменение режима работы

Вы можете изменить режим работы CPU с помощью кнопок. Активны только те кнопки, которые могут быть выбраны в текущем режиме работы.

16.3 Отображение и установка времени и даты

16.3.1 Отображение и установка времени и даты

Действуйте следующим образом:

1. Откройте свой проект и выделите программу S7/M7 или откройте окно "Accessible Nodes [Доступные узлы]" с помощью команды меню **PLC > Display Accessible Nodes [ПЛК > Отобразить доступные узлы]** и выберите узел ("MPI=...").
2. Выберите команду меню **PLC > Set Date and Time [ПЛК > Установить дату и время]**.
   Эта команда меню может быть выбрана только в том случае, если в окне проекта выделена программа S7/M7 (представление online) или выделен узел ("MPI=...") в окне "Accessible Nodes [Доступные узлы]".
3. В открывшемся диалоговом окне вы можете прочитать текущее время и дату на выбранном модуле.
4. Если необходимо, вы можете ввести новые значения в поля "Date [Дата]" и "Time [Время]" или вы можете использовать опцию умолчания, чтобы принять дату и время, установленные на вашем устройстве программирования/PC.

Замечание

Если в модуле нет часов реального времени, то диалоговое окно показывает "00:00:00" для времени и "00.00.00" для даты.

---

17 Загрузка и считывание

17.1 Загрузка из PG/PC в программируемый контроллер

17.1.1 Предпосылки для загрузки

Предпосылки для загрузки в программируемый контроллер

- Должна быть установлена связь между вашим устройством программирования и CPU в программируемом контроллере (например, через многоточечный интерфейс).
- Должен быть возможен доступ к программируемому контроллеру.
- Загружаемая вами программа скомпилирована без ошибок.
- CPU должен находиться в рабочем режиме, для которого загрузка разрешена (STOP или RUN-P).
  Имейте в виду, что в режиме RUN-P программа будет загружаться поблочно. Если, делая это, вы переписываете старую программу CPU, то могут возникнуть конфликты, например, если изменились параметры блока. Тогда при обработке цикла CPU переходит в состояние STOP. Поэтому мы рекомендуем вам перед загрузкой переводить CPU в STOP.
- Если вы открыли блок offline и хотите загрузить его, то CPU должен быть связан с пользовательской программой online в SIMATIC Manager.
- Перед загрузкой своей пользовательской программы вам следует сбросить CPU, чтобы обеспечить отсутствие "старых" блоков в CPU.

Режим STOP

Переключайте режим работы из RUN в STOP перед следующими операциями:

- Загрузка всей программы пользователя или ее частей в CPU
- Выполнение сброса памяти на CPU
- Сжатие памяти пользователя

Теплый рестарт (переход в режим RUN)

Если вы выполняете теплый рестарт в режиме "STOP", то программа перезапускается и сначала обрабатывает программу запуска (в блоке OB100) в режиме STARTUP [запуск]. Если запуск успешен, то CPU переходит в режим RUN. Теплый рестарт требуется после:

- сброса CPU
- загрузки программы пользователя в режиме STOP

17.1.2 Различия между сохранением и загрузкой блоков

Вы всегда должны различать сохранение и загрузку блоков.

	Сохранение	**Загрузка**
Команды меню	**File > Save [Файл > Сохранить]** **File > Save As [Файл > Сохранить как...]**	**PLC > Download [ПЛК >Загрузить]**
Действие	Текущее состояние блока в редакторе сохраняется на жестком диске устройства программирования.	Текущее состояние блока в редакторе только загружается в CPU.
Проверка синтаксиса	Производится проверка синтаксиса. Обо всех ошибках сообщается в диалоговых окнах. Показываются также причины ошибок и их местонахождение. Вы должны исправить эти ошибки, прежде чем сохранить или загрузить блок. Если ошибки в синтаксисе не обнаружены, то блок компилируется в машинный код, а затем сохраняется или загружается.	Производится проверка синтаксиса. Обо всех ошибках сообщается в диалоговых окнах. Показываются также причины ошибок и их местонахождение. Вы должны исправить эти ошибки, прежде чем сохранить или загрузить блок. Если ошибки в синтаксисе не обнаружены, то блок компилируется в машинный код, а затем сохраняется или загружается.

Эта таблица имеет силу независимо от того, открыли ли вы блок online или offline.

Совет по изменениям блоков – сначала сохранить, затем загрузить

Чтобы ввести вновь созданные блоки или изменения в раздел кодов логических блоков или в таблицы описаний, или ввести новые или измененные значения в блоки данных, вы должны сохранить соответствующий блок. Любые изменения, которые вы выполняете в редакторе и передаете в CPU с помощью команды меню **PLC > Download [ПЛК > Загрузить]**, например, для тестирования небольших изменений, должны быть также в каждом случае сохранены на жестком диске устройства программирования до выхода из редактора. Иначе у вас будут разные версии вашей пользовательской программы в CPU и в устройстве программирования. В общем случае рекомендуется сначала сохранять все изменения, а затем их загружать.

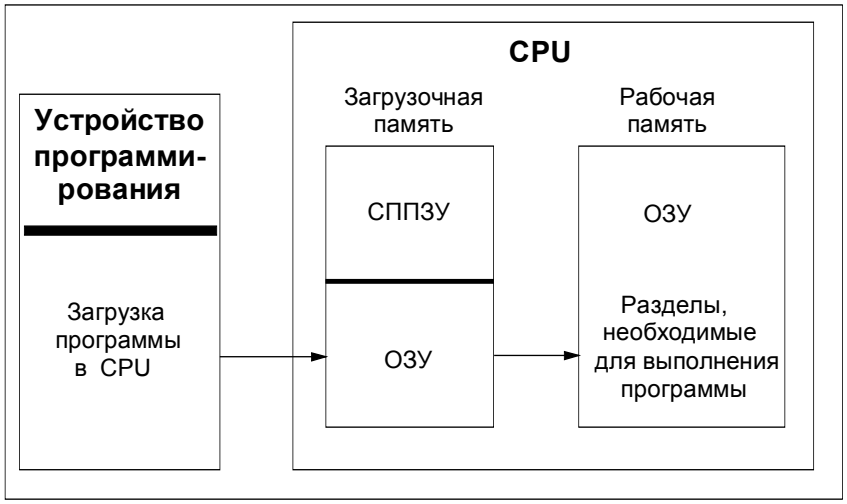
17.1.3 Загрузочная и рабочая память в CPU

После завершения конфигурирования, назначения параметров, создания программы и установления связи online вы можете загружать программы пользователя полностью или отдельные блоки в программируемый контроллер. Чтобы тестировать отдельные блоки, вы должны загрузить, по крайней мере, один организационный блок (OB) и функциональные блоки (FB) и функции (FC), вызываемые в этом OB, и используемые блоки данных (DB). Чтобы загрузить в программируемый контроллер системные данные, созданные при конфигурировании аппаратуры, сетей и формировании таблицы соединений, вы загружаете объект ”System Data [Системные данные]".

Пользовательские программы загружаются в программируемый контроллер с помощью SIMATIC Manager, например, на последнем этапе тестирования программы или для исполнения завершенной программы пользователя.

Связь между загрузочной и рабочей памятью

Вся программа пользователя загружается в загрузочную память; разделы, необходимые для выполнения программы, загружаются также в рабочую память .



Загрузочная память CPU

- Загрузочная память используется для хранения программы пользователя без таблицы символов и комментариев (они остаются в памяти устройства программирования).
- Боки, не помеченные как необходимые для запуска, будут храниться только в загрузочной памяти.
- Загрузочная память может быть ОЗУ, ПЗУ или СППЗУ в зависимости от программируемого контроллера.
- Загрузочная память может иметь также встроенный раздел ЭСППЗУ, а также встроенный раздел ОЗУ (например, CPU 312 IFM и CPU 314 IFM).
- В S7-400 настоятельно требуется использование платы памяти (ОЗУ или ЭСППЗУ) для расширения загрузочной памяти.

Рабочая память CPU

Рабочая память (встроенное ОЗУ) используется для хранения разделов программы пользователя, необходимых для обработки программы.

Возможные процедуры загрузки

Функция загрузки используется для передачи программы пользователя или загружаемых объектов (например, блоков) в программируемый контроллер. Если блок уже существует в ОЗУ CPU, вы получите подсказку, чтобы подтвердить, хотите вы или нет, чтобы блок был переписан.

- Вы можете выделить загружаемые объекты в окне проекта, а затем загрузить их из SIMATIC Manager (команда меню: **PLC > Download [ПЛК > Загрузить]**).
- При программировании блоков и конфигурировании аппаратуры и сетей вы можете непосредственно загрузить объект, который вы в данный момент редактируете, используя меню в главном окне приложения, с которым вы работаете (команда меню: **PLC > Download [ПЛК > Загрузить]**).
- Еще одна возможность состоит в том, чтобы открыть окно online с отображением программируемого контроллера (например, с помощью **View > Online [Вид > Online]** или **PLC > Display Accessible Nodes [ПЛК > Отобразить доступные узлы]**) и скопировать объект, который вы хотите загрузить, в окно online.

В качестве альтернативы вы можете считать текущее содержимое блоков из ОЗУ загрузочной памяти CPU в свое устройство программирования с помощью функции считывания.

См. также:

Загрузочная и рабочая память

17.1.4 Методы загрузки, зависящие от загрузочной памяти

Разделение загрузочной памяти CPU на области ОЗУ и ЭСППЗУ определяет методы, доступные для загрузки вашей пользовательской программы или блоков в вашей пользовательской программе. Для загрузки данных в CPU возможно использование следующих методов:

Загрузочная память	Метод загрузки	Тип связи между PG и ПЛК
ОЗУ	Загрузка и удаление отдельных блоков	Связь PG – ПЛК online
	Загрузка и удаление всей программы пользователя	Связь PG – ПЛК online
	Перезагрузка отдельных блоков	Связь PG – ПЛК online
Встроенное (только в S7-300) или вставляемое СППЗУ	Загрузка программ пользователя целиком	Связь PG – ПЛК online
Вставляемое СППЗУ	Загрузка программ пользователя целиком	Внешняя загрузка СППЗУ и вставка платы памяти

Загрузка в ОЗУ через соединение online

Данные в программируемом контроллере теряются, если происходит сбой по питанию, а ОЗУ не резервируется. Данные в ОЗУ в этом случае будут потеряны.

Сохранение на плате памяти СППЗУ

Блоки или программа пользователя сохраняются на плате памяти СППЗУ, которая затем вставляется в гнездо на CPU.

Платы памяти – это съемные носители данных. Они записываются с помощью устройства программирования, а затем вставляются в соответствующее гнездо на CPU.

Хранящиеся на них данные не теряются после потери питания и при сбросе CPU. Содержимое СППЗУ снова копируется в область ОЗУ памяти CPU, когда питание восстанавливается после сброса памяти CPU и выключения питания, если ОЗУ не резервируется.

Сохранение во встроенном СППЗУ

Для CPU 312 вы можете также сохранить содержимое ОЗУ во встроенном СППЗУ. Данные во встроенном СППЗУ при выключенном питании сохраняются. Содержимое встроенного СППЗУ снова копируется в область ОЗУ памяти CPU, когда питание восстанавливается после выключения питания и сброса памяти CPU, если ОЗУ не резервируется

17.2 Загрузка из программируемого контроллера в PG/PC

13.1.1 Загрузка из программируемого контроллера в PG/PC

Эта помогает вам при выполнении следующих действий:

- сохранение информации из программируемого контроллера (например, для целей обслуживания)
- быстрое конфигурирование и редактирование станции, если компоненты аппаратуры доступны перед началом конфигурирования.

Сохранение информации из программируемого контроллера

Это мероприятие может оказаться необходимым, если, например, данные проекта offline версии, выполняемой на CPU, недоступны или доступны только частично. В этом случае вы можете, по крайней мере, восстановить данные проекта, доступные в режиме online, и загрузить их в свое устройство программирования.

Быстрое конфигурирование

Ввод конфигурации станции облегчается, если вы считываете конфигурационные данные из программируемого контроллера в свое устройство программирования после того, как вы сконфигурировали аппаратуру и перезапустили станцию. Это предоставляет в ваше распоряжение конфигурацию станции и типы отдельных модулей. Затем все, что вам нужно сделать, - это более подробно определить эти модули (заказной номер) и назначить им параметры.

В устройство программирования считывается следующая информация:

- S7-300: конфигурация для центральной стойки и всех стоек расширения
- S7-400: конфигурация для центральной стойки с CPU и сигнальных модулей без стоек расширения
- Конфигурационные данные для децентрализованной периферии не могут быть считаны в устройство программирования.

Эта информация считывается, если в программируемом контроллере отсутствует конфигурационная информация; например, если в системе был выполнен сброс памяти. В противном случае функция **Upload [Считать]** дает значительно лучшие результаты.

Для систем S7-300 без децентрализованной периферии все, что вам нужно сделать, - это более подробно определить эти модули (заказной номер) и назначить им параметры.

Замечание

При считывнии данных из контроллера (если только у вас уже нет конфигурации offline) STEP 7 не может определить все заказные номера компонентов.

Вы можете ввести "неполные" заказные номера при конфигурировании аппаратуры с помощью команды меню **Options > Specify Module [Параметры > Определить модуль]**. Таким способом вы можете назначить параметры модулям, которые STEP 7 не распознает (т. е. модули, которые не появляются в окне "Hardware Catalog [Каталог аппаратуры]"); однако STEP 7 потом не проверяет, придерживаетесь ли вы правил назначения параметров.

Ограничения при считывании из программируемого контроллера

При считывании данных из программируемого контроллера в устройство программирования действуют следующие ограничения:

- Блоки не содержат символических имен для параметров, переменных и меток
- Блоки не содержат комментариев
- Вся программа считывается со всеми системными данными, при этом система может продолжать обрабатывать только системные данные, принадлежащие приложению "Configuring Hardware [Конфигурирование аппаратуры]"
- В дальнейшем не могут обрабатываться данные для связи с помощью глобальных данных (GD) и для проектирования сообщений, связанных с символами
- Принудительные задания не считываются в устройство программирования вместе с другими данными. Они должны быть сохранены отдельно в виде таблицы переменных (VAT)
- Комментарии в диалоговых окнах модулей не считываются.
- Имена модулей отображаются только в том случае, если эта опция была выбрана при конфигурировании (HW Config: опция "Save object names in the programmable logic controller [Сохранить имена объектов в программируемом логическом контроллере]" в диалоговом окне, вызываемом через **Options > Customize [Параметры > Настройка]**).

17.2.2 Загрузка станции в устройство программирования

С помощью команды меню **PLC > Upload Station [ПЛК > Загрузить станцию в устройство программирования]** вы можете считать текущую конфигурацию и все блоки из выбранного вами программируемого контроллера в устройство программирования.

Чтобы сделать это, STEP 7 создает новую станцию в текущем проекте, в которой будет сохранена эта конфигурация. Вы можете изменить предустановленное имя новой станции (например, "SIMATIC 300-Station(1)"). Вставленная станция отображается как в представлении online, так и в представлении offline.

Эта команда меню может быть выбрана, когда проект открыт. Выделение объекта в окне проекта или в представлении online или offline не оказывает влияния на эту команду меню.

Вы можете использовать эту функцию для облегчения конфигурирования.

- Для программируемых контроллеров S7-300 фактическая конфигурация аппаратуры считывается с включением стоек расширения, но без децентрализованной периферии (DP).
- Для программируемых контроллеров S7-400 конфигурация считывается без стоек расширения и без децентрализованной периферии.

У систем S7-300 без децентрализованной периферии все, что вам нужно сделать, - это более подробно определить модули (заказной номер) и назначить им параметры.

Ограничения при загрузке станций в устройство программирования

Для данных, считываемых в устройство программирования, действуют следующие ограничения:

- Блоки не содержат символических имен для параметров, переменных и меток
- Блоки не содержат комментариев
- Вся программа считывается со всеми системными данными, при этом не все данные могут обрабатываться в дальнейшем
- В дальнейшем не могут обрабатываться данные для связи с помощью глобальных данных (GD), для проектирования сообщений, связанных с символами, и для конфигурирования сетей
- Принудительные задания не могут быть считаны в устройство программирования, а затем загружены обратно в программируемый контроллер.

17.2.3 Загрузка блоков из CPU S7

Вы можете загрузить блоки S7 из CPU на жесткий диск устройства программирования с помощью SIMATIC Manager. Загрузка блоков в устройство программирования полезна в следующих ситуациях:

- Создание резервной копии текущей программы пользователя, загруженной в CPU. Эта копия затем может быть загружена снова, например, после обслуживания или сброса памяти CPU обслуживающим персоналом.
- Вы можете загрузить программу пользователя из CPU в устройство программирования и редактировать ее там, например, для поиска ошибок. В этом случае у вас нет доступа к символам или комментариям для документирования программы. Поэтому мы рекомендуем использовать эту процедуру только в целях обслуживания.

17.2.4 Редактирование загруженных блоков в PG/PC

Возможность загружать блоки из CPU в устройство программирования имеет следующие преимущества:

- На этапе тестирования вы можете корректировать блок непосредственно в CPU и задокументировать результат.
- Вы можете загружать текущее содержимое блоков из загрузочной памяти ОЗУ CPU на свое устройство программирования с помощью функции загрузки.

Замечание

Конфликты меток времени при работе online и offline

Следующие процедуры ведут к конфликтам меток времени и поэтому должны избегаться.

Конфликты меток времени возникают, когда вы открываете блок online, если:

Изменения, сделанные online, не были сохранены в программе пользователя S7, открытой offline

Изменения, сделанные offline, не были загружены в CPU

Конфликты меток времени возникают, когда вы открываете блок offline, если:

Блок online с конфликтом меток времени копируется в программу пользователя S7 offline, и блок затем открывается offline.

Два различных случая

При загрузке блоков из CPU в устройство программирования следует помните, что имеются две различных ситуации:

1. Программа пользователя, которой принадлежат блоки, расположена на устройстве программирования.
2. Программа пользователя, которой принадлежат блоки, не находится на устройстве программирования.

Это значит, что перечисленные ниже разделы программы, которые не могут быть загружены в CPU, недоступны. Этими компонентами являются:

- Таблица символов с символическими именами адресов и комментариями
- Комментарии к сегментам программ, представленных в виде контактного или функционального плана
- Комментарии к строкам программы, представленной в виде списка команд
- Типы данных, определенные пользователем

17.2.5 Удаление в программируемом контроллере

17.2.5.1 Очистка загрузочной/рабочей памяти и сброс CPU

Перед загрузкой своей пользовательской программы в программируемый контроллер S7 вам следует выполнить сброс памяти на CPU, чтобы обеспечить отсутствие "старых" блоков в CPU.

Предпосылки для сброса памяти

Чтобы выполнить сброс памяти, CPU должен находиться в состоянии STOP (переключатель режимов установлен в STOP, или он установлен в RUN-P, и режим изменен на STOP с помощью команды меню **PLC > Operating Mode [ПЛК > Режим работы]**).

Выполнение сброса памяти на CPU S7

При выполнении сброса памяти на CPU S7 происходит следующее:

- CPU сбрасывается.
- Все данные пользователя удаляются (блоки и системные блоки данных (SDB) за исключением параметров MPI).
- CPU разрывает все существующие связи.
- Если на СППЗУ имеются данные (плата памяти или встроенное СППЗУ), то CPU после сброса памяти копирует содержимое СППЗУ обратно в область ОЗУ.

Содержимое диагностического буфера и параметры MPI сохраняется.

Выполнение сброса памяти на CPU/FM M7

Когда сброс памяти выполняется на M7 CPU/FM, то происходит следующее:

- Восстанавливается исходное состояние.
- Системные блоки данных (SDB) за исключением параметров MPI удаляются.
- CPU/FM разрывает все существующие связи. Пользовательские программы сохраняются и продолжат работу после переключения CPU из STOP в RUN.

С помощью функции "memory reset [сброс памяти]" вы можете восстановить первоначальное состояние CPU или FM M7 после серьезных ошибок путем удаления текущих системных блоков данных (SDB) в ОЗУ. В некоторых случаях потребуется теплый рестарт операционной системы. Чтобы сделать это, вы очищаете M7 с помощью переключателя режимов работы (ключ в положение MRES). Сброс с помощью переключателя режимов работы на CPU или FM SIMATIC M7 возможен только в том случае, если на CPU/FM используется операционная система RMOS32.

17.2.5.2 Удаление блоков S7 в программируемом контроллере

Удаление отдельных блоков на CPU может оказаться необходимым на этапе тестирования программы CPU. Блоки хранятся в памяти пользователя CPU в СППЗУ или в ОЗУ (в зависимости от CPU и процедуры загрузки).

- Блоки в ОЗУ могут быть удалены непосредственно. Занятое пространство в загрузочной или рабочей памяти освобождается и может быть снова использовано.
- Блоки во встроенном СППЗУ после сброса памяти CPU всегда копируются в область ОЗУ. Эти копии в ОЗУ могут быть удалены непосредственно. После этого удаленные блоки помечаются в СППЗУ как недействительные вплоть до следующего сброса памяти или выключения питания без резервирования ОЗУ. После сброса памяти или выключения питания без резервирования ОЗУ "удаленные" блоки копируются из СППЗУ в ОЗУ и становятся активными. Блоки во встроенном СППЗУ (например, в CPU 312) удаляются путем переписывания их новым содержимым ОЗУ.
- Платы памяти СППЗУ должны быть стерты в устройстве программирования.

17.2.6 Сжатие памяти пользователя (ОЗУ)

17.2.6.1 Пропуски в памяти пользователя (ОЗУ)

После удаления и перезагрузки блоков в памяти пользователя (загрузочной и рабочей) могут возникнуть пропуски, сокращая тем самым доступную для использования область памяти. С помощью функции сжатия существующие блоки переставляются в памяти пользователя без пропусков, и создается непрерывная свободная память.

На следующем рисунке показана диаграмма того, как занятые блоки памяти сдвигаются вместе функцией сжатия.

Всегда старайтесь сжимать память в режиме STOP

Все пропуски закрываются только в том случае. Если сжатие памяти производится в режиме "STOP". В режиме RUN-P (положение переключателя режимов работы) блоки, обрабатываемые в данный момент времени, не могут быть сдвинуты, так как они открыты. Функция сжатия не действует в режиме RUN (положение переключателя режимов работы) (защита от записи!).

18 Тестирование с помощью таблицы переменных

18.1 Введение в тестирование с помощью таблицы переменных

При тестировании с помощью таблицы переменных доступны следующие функции:

- **Наблюдение переменных**
  Эта функция дает вам возможность отображать на устройстве программирования/PC текущие значения отдельных переменных в программе пользователя или CPU.
- **Изменение переменных**
  Вы можете использовать эту функцию для назначения фиксированных значений отдельным переменным программы пользователя или CPU. Непосредственное изменение переменных возможно также при тестировании с использованием статуса программы.
- **Разблокировка периферийного выхода** и **Активизация изменения значений**
  Эти две функции позволяют присваивать фиксированные значения отдельным периферийным выходам программы пользователя или CPU в режиме STOP.
- **Принудительное присваивание значений переменным**
  Вы можете использовать эту функцию для назначения отдельным переменным программы пользователя или CPU фиксированных значений, которые не могут быть переписаны программой пользователя.

Вы можете присваивать или отображать значения для следующих переменных:

- входы, выходы, меркеры, таймеры и счетчики
- содержимое блоков данных
- периферия

Переменные, значения которых вы хотите отобразить или изменить, вводятся в таблицы переменных.

Вы можете определить, когда и как часто переменные наблюдаются или им присваиваются новые значения, путем назначения точки запуска и частоты запуска.

18.2 Основная последовательность действий при наблюдении и изменении переменных с помощью таблицы переменных

Для использования функций **Monitor [Наблюдение]** и **Modify [Изменение]** действуйте следующим образом:

1. Создайте новую или откройте существующую таблицу переменных.
2. Отредактируйте или проверьте содержимое таблицы переменных.
3. Установите связь online между текущей таблицей переменных и нужным CPU с помощью команды меню **PLC > Connect To [ПЛК > Соединить с]**.
4. С помощью команды меню **Variable > Trigger [Переменная > Запустить]**, выберите подходящую точку запуска и установите частоту запуска.
5. Команды меню **Variable > Monitor [Переменная > Наблюдать]** и **Variable > Modify [Переменная > Изменить]** включают и выключают функции наблюдения и изменения переменных.
6. Сохраните завершенную таблицу переменных с помощью команды меню **Table > Save [Таблица > Сохранить]** или **Table > Save As [Таблица > Сохранить как...]**, так что вы сможете вызвать ее снова в любое время.

18.3 Редактирование и сохранение таблиц переменных

18.3.1 Создание и открытие таблицы переменных

Прежде чем вы сможете наблюдать и изменять переменные, вы должны создать таблицу переменных (VAT) и ввести требуемые переменные. Для создания таблицы переменных вы можете выбрать один из следующих двух методов:

В SIMATIC Manager:

- Выделите папку "Blocks [Блоки]" и выберите команду меню **Insert > S7 Block > Variable Table [Вставить > Блок S7 > Таблица переменных]**. В диалоговом окне вы можете дать таблице имя. Вы можете открыть таблицу переменных, дважды щелкнув на этом объекте.
- Выберите соединение или, в представлении online, программу S7 или M7 из списка доступных узлов. Вы создадите таблицу переменных, не имеющую имени, с помощью команды меню **PLC > Monitor/Modify Variables [ПЛК > Наблюдение/изменение переменных]**.

В "Monitor/Modify Variables [Наблюдение/изменение переменных]":

- Для создания новой таблицы переменных, которая еще не назначена ни одной из программ S7 или M7, вы можете использовать команду меню **Table > New [Таблица > Новая]**. Существующие таблицы вы можете открывать с помощью команды **Table > Open [Таблица > Открыть]**.
- Для создания или открытия таблиц переменных вы можете использовать соответствующие символы на панели инструментов.

Создав однажды таблицу переменных, вы можете ее сохранить, распечатать и использовать ее снова и снова для наблюдения и изменения значений.

18.3.2 Сохранение таблицы переменных

Сохраненные таблицы переменных вы можете снова использовать для наблюдения и изменения переменных при тестировании программы.

1. Сохраните таблицу переменных с помощью команды меню **Table > Save [Таблица > Сохранить]**.
2. Если вы работали с таблицей переменных, не имеющей имени, то вы должны теперь ввести номер, чтобы дать таблице имя, например, VAT4711.

Если ваша таблица переменных открыта более одного раза, то команды меню **Table > Properties [Таблица > Свойства]** и **Table > Save [Таблица > Сохранить]** не могут быть выбраны. Если вы сделали изменения в одном из окон, которые вы хотите сохранить, то это возможно только с помощью команды меню **Table > Save As [Таблица > Сохранить как...]**.

При сохранении таблицы переменных (VAT) сохраняются также текущие настройки и формат таблицы. Это значит, что настройки, сделанные в пункте меню "Trigger [Запуск]", сохраняются.

18.4 Ввод переменных в таблицу переменных

18.4.1 Вставка адресов или символов в таблицу переменных

Выберите переменные, значения которых вы хотите изменять или наблюдать, и введите их в таблицу переменных. Начинайте "снаружи" и продвигайтесь "внутрь"; это значит, что сначала вам следует выбрать входы, затем переменные, на которые эти входы влияют и которые влияют на выходы, и, наконец, выходы.

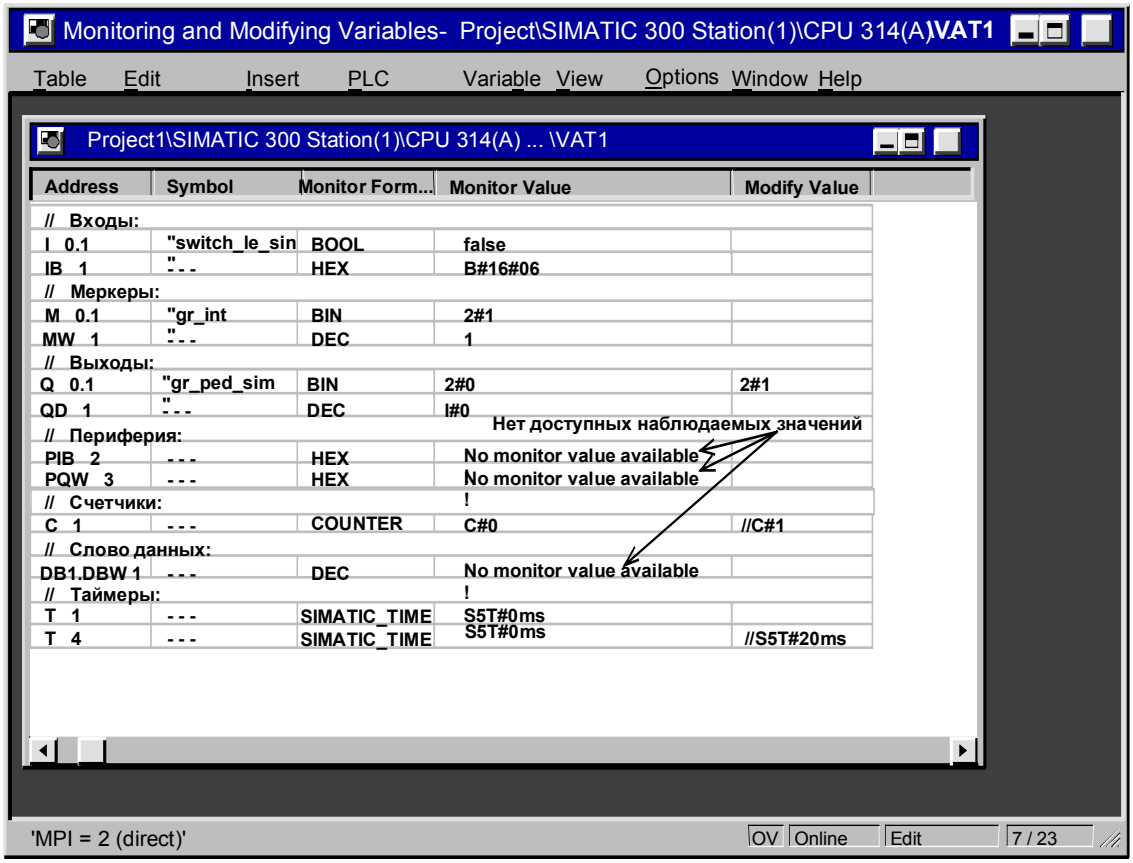
Если, например, вы хотите наблюдать входной бит 1.0, меркерное слово 5 и выходной байт 0, то введите следующее в столбец "Address [Адрес]":

Пример:

I 1.0
MW5
QB0

Пример завершенной таблицы переменных

На следующем рисунке показана таблица переменных со следующими видимыми столбцами: Address [Адрес], Symbol [Символ], Monitor Format [Формат наблюдения], Monitor Value [Наблюдаемое значение] и Modify Value [Назначаемое значение].



Замечания по вставке символов

- Переменная, которую вы о хотите изменить, вводится с помощью адреса или в виде символа. Вы можете вводить символы и адреса или в столбце "Symbol [Символ]" или в столбце "Address [Адрес]". Этот ввод затем автоматически записывается в правильный столбец.
  Если соответствующий символ определен в таблице символов, то столбец символов или столбец адресов заполняется автоматически.
- Вы можете вводить только символы, которые уже были определены в таблице символов.
- Символ должен быть введен точно так же, как он был определен в таблице символов.
- Символические имена, включающие в себя специальные знаки, должны быть заключены в кавычки (например, "Motor.Off", "Motor+Off", "Motor-Off").
- Для определения новых символов в таблице символов выберите команду меню **Options > Symbol Table [Параметры > Таблица символов]**. Символ может быть также скопирован из таблицы символов и вставлен в таблицу переменных.

Проверка синтаксиса

При вводе переменных в таблицу переменных в конце каждой строки производится проверка синтаксиса. Любой неправильный ввод отмечается красным цветом. Поместив курсор в строку, отмеченную красным цветом, вы можете прочитать причину ошибки в строке состояний. Указания по исправлению ошибки могут быть получены нажатием F1.

Максимальный размер

В таблице переменных допускается не более 255 знаков на строку. Использование возврата каретки для перехода в другую строку не допускается. Таблица переменных может иметь не более 1024 строк. Это ее максимальный размер.

18.4.2 Вставка назначаемых значений

Задаваемая величина как комментарий

Если вы поместите метку комментария ("//") в столбце "Modify Value [Задаваемая величина]" перед значением переменной, которую вы хотите изменить, то это делает данное значение недействительным. Когда метка комментария удаляется, то значение снова становится действительным и может быть изменено.

Вы можете также использовать команду меню **Variable > Modify Value as Comment [Переменная > Задаваемая величина как комментарий]**, чтобы подтвердить или отменить действие задаваемой величины.

18.4.3 Верхние границы для ввода таймеров

Примите во внимание следующие верхние границы для ввода таймеров:

Пример: W#16#3999 (максимальное значение в формате BCD)

Примеры:

Адрес	Формат наблюдения	Ввод	Отображение задаваемой величины	Объяснение
T 1	SIMATIC_TIME	137	S5TIME#130MS	Преобразование в миллисекунды
MW4	SIMATIC_TIME	137	S5TIME#890MS	Представление в формате BCD возможно
MW4	HEX	137	W#16#0089	Представление в формате BCD возможно
MW6	HEX	157	W#16#009D	Представление в формате BCD невозможно, поэтому формат наблюдения SIMATIC_TIME не может быть выбран

Указание

Вы можете вводить таймеры миллисекундными шагами, но введенное значение адаптируется к выделенному кванту времени. Размер кванта времени зависит от введенного значения времени (137 превращается в 130 мс; 7 мс были округлены в меньшую сторону).

Задаваемые значения для адресов, относящихся к типу данных WORD, например, IW1, преобразуются в формат BCD. Однако не каждое битовое представление является допустимым BCD-числом. Если введенная величина не может быть представлена как SIMATIC_TIME для адреса, имеющего тип данных WORD, то приложение автоматически возвращается к формату по умолчанию (здесь: HEX, см. Выбор формата наблюдения, команда по умолчанию (меню View [Вид])), так чтобы введенная величина могла быть отображена.

Формат BCD для переменных в формате SIMATIC_TIME

Значения переменных в формате SIMATIC_TIME вводятся в формате BCD. 16 битов имеют следующие значения:

| 0 0 x x | c c c c | д д д д | е е е е |

Биты 15 и 14 всегда равны 0.
Биты 13 и 12 (помеченные xx) устанавливают множитель для битов с 0 по 11:
- 00 => множитель 10 миллисекунд
- 01 => множитель 100 миллисекунд
- 10 => множитель 1 секунда
- 11 => множитель 10 секунд

Биты с 11 по 8 сотни (cccc)
Биты с 7 по 4 десятки (дддд)
Биты с 3 по 0 единицы (eeee)

18.4.4 Верхние границы для ввода счетчиков

Обратите внимание на следующие верхние границы для ввода счетчиков:

Верхняя граница для счетчиков: C#999
W#16#0999 (максимальное значение в формате BCD)

Примеры:

Адрес	Формат наблюдения	Ввод	Отображение задаваемой величины	Объяснение
C1	COUNTER	137	C#137	Преобразование
MW4	COUNTER	137	C#89	Представление в формате BCD возможно
MW4	HEX	137	W#16#0089	Представление в формате BCD возможно
MW6	HEX	157	W#16#009D	Представление в формате BCD невозможно, поэтому формат наблюдения COUNTER не может быть выбран

Указание

Если вы вводите для счетчика десятичное число и не помечаете это значение с помощью C#, то это значение автоматически преобразуется в формат BCD (137 превращается в C#137).

Задаваемые значения для адресов, относящихся к типу данных WORD, например, IW1, преобразуются в формат BCD. Однако не каждое битовое представление является допустимым BCD-числом. Если введенная величина не может быть представлена как COUNTER для адреса, имеющего тип данных WORD, то приложение автоматически возвращается к формату по умолчанию (здесь: HEX, см. Выбор формата наблюдения, команда по умолчанию (меню View [Вид])), так чтобы введенная величина могла быть отображена.

18.4.5 Вставка строк комментария

Строки комментария вводятся с помощью метки комментария ”//".

С помощью команды меню **Edit > Comment Line [Редактировать > Строка комментария]** или соответствующей кнопки на панели инструментов вы можете временно отобразить строку таблицы как строку комментариев.

18.5 Пример ввода в таблицы переменных

18.5.1 Пример ввода адресов в таблицы переменных

Допустимые адреса:	Тип данных:	Пример (международная мнемоника):
Вход | Выход | Меркер	BOOL	I 1.0 | Q 1.7 | M 10.1
Вход | Выход | Меркер	BYTE	IB 1 | QB 10 | MB 100
Вход | Выход | Меркер	WORD	IW 1 | QW 10 | MW 100
Вход | Выход | Меркер	DWORD	ID 1 | QD 10 | MD 100
Периферия (Вход | Выход)	BYTE	PIB 0 | PQB 1
Периферия (Вход | Выход)	WORD	PIW 0 | PQW 1
Периферия (Вход | Выход)	DWORD	PID 0 | PQD 1
Таймеры	TIMER	T 1
Счетчики	COUNTER	C 1
Блок данных	BOOL	DB1.DBX 1.0
Блок данных	BYTE	DB1.DBB 1
Блок данных	WORD	DB1.DBW 1
Блок данных	DWORD	DB1.DBD 1

Замечание

Ввод "DB0. .." запрещен, так как он уже используется внутри системы.

В окне Force Values [Принудительно задаваемые значения]:

У модулей S7-300 принудительно могут задаваться только входы, выходы и периферийные выходы.

У модулей S7-400 принудительно могут задаваться только входы, выходы, меркеры и периферия (входы/выходы).

18.5.2 Пример ввода непрерывной области адресов

Откройте таблицу переменных и вызовите диалоговое окно "Insert Block [Вставить блок]" командой меню **Insert > Block [Вставить > Блок]**.

Для вводов этого диалогового окна в таблицу переменных вставляются следующие строки для области меркеров:

От адреса: M 3.0

Количество: 10

Формат наблюдения: BIN

Адрес	Формат наблюдения
M 3.0	BIN
M 3.1	BIN
M 3.2	BIN
M 3.3	BIN
M 3.4	BIN
M 3.5	BIN
M 3.6	BIN
M 3.7	BIN
M 4.0	BIN
M 4.1	BIN

Обратите внимание, что в этом примере обозначение в столбце "Address [Адрес]" изменяется через восемь записей.

18.5.3 Примеры ввода для задаваемых и принудительно задаваемых значений

Битовые адреса

Возможные битовые адреса	Допустимые задаваемые и принудительно задаваемые значения
I1.0	true [истина]
M1.7	false [ложь]
Q10.7	0
DB1.DBX1.1	1
I1.1	2#0
M1.6	2#1

Байтовые адреса

Возможные байтовые адреса	Допустимые задаваемые и принудительно задаваемые значения
IB 1	2#00110011
MB 12	b#16#1F
MB 14	1F
QB 10	'a'
DB1.DBB 1	10
PQB 2	-12

Адреса слов

Возможные адреса слов	Допустимые задаваемые и принудительно задаваемые значения
IW 1	2#0011001100110011
MW12	w#16#ABCD
MW14	ABCD
QW 10	b#(12,34)
DB1.DBW 1	'ab'
PQW 2	-12345
MW3	12345
MW5	s5t#12s340ms
MW7	0.3s or 0,3s
MW9	c#123
MW11	d#1990-12-31

Адреса двойных слов

Возможные адреса двойных слов	Допустимые задаваемые и принудительно задаваемые значения
ID 1	2#00110011001100110011001100110011
MD 0	1.23e4
MD 4	1.2
QD 10	dw#16#abcdef10
QD 12	ABCDEF10
DB1.DBD 1	b#(12,34,56,78)
PQD 2	'abcd'
MD 8	l# -12
MD 12	l#12
MD 16	-123456789
MD 20	123456789
MD 24	t#12s345ms
MD 28	tod#1:2:34.567
MD 32	p#e0.0

Таймер

Возможные адреса типа "Timer"	Допустимые задаваемые и принудительно задаваемые значения	Объяснение
T 1	0	Преобразование в миллисекунды (мс)
T 12	20	Преобразование в мс
T 14	12345	Преобразование в мс
T 16	s5t#12s340ms	
T 18	1.3	Преобразование в 1с 300 мс
T 20	1.3s	Преобразование в 1с 300 мс

Изменение таймера влияет только на значение, но не на состояние. Это значит, что таймеру T1 может быть задано значение 0, не изменяя при этом результата логической операции для A T1.

Буквенные константы s5t, s5time могут быть записаны в верхнем или в нижнем регистре.

Счетчик (COUNTER)

Возможные адреса типа "Counter"	Допустимые задаваемые и принудительно задаваемые значения
C 1	0
C 14	20
C 16	c#123

Изменение счетчика оказывает влияние только на значение, но не на состояние. Это значит, что счетчику C1 может быть задано значение 0 без изменения результата логической операции A C1.

18.6 Установление связи с CPU

18.6.1 Установление связи с CPU

Чтобы иметь возможность наблюдать и изменять переменные, которые вы ввели в свою текущую таблицу переменных (VAT), вы должны установить связь с соответствующим CPU. Каждую таблицу переменных можно связывать с различными CPU.

Отображение связи online

Если связь online существует, то в строке состояний для окна появляется слово ”Online".

Установление связи online с CPU

Если связь online с нужным CPU не существует, то для возможности наблюдения и изменения переменных воспользуйтесь командой меню **PLC > Connect To > ... [ПЛК > Соединить с > …]**, чтобы определить связь с требуемым CPU. В качестве альтернативы вы можете также щелкнуть на соответствующих кнопках на панели инструментов.

Связь со сконфигурированным CPU

Связь с непосредственно подключеченным CPU (например, MPI=2 (direct [непосредственно]))

Разрыв связи Online с CPU

Связь между таблицей переменных и CPU разрывается с помощью команды меню **PLC > Disconnect [ПЛК > Отсоединить]**.

Замечание

Если вы создали таблицу переменных без имени с помощью команды меню **Table > New [Таблица > Новая]**, то вы можете установить связь с последним сконфигурированным CPU, если он определен.

18.7 Наблюдение переменных

18.7.1 Введение в наблюдение переменных

Для наблюдения переменных в вашем распоряжении имеются следующие методы:

- Активизируйте функцию наблюдения с помощью команды меню **Variable > Monitor [Переменная > Наблюдать]**. Значения выбранных переменных отображаются в таблице переменных в соответствии с установленными точкой и частотой запуска. Если вы установили частоту запуска ”Every cycle [Каждый цикл]", то снова отключить функцию наблюдения с помощью команды меню **Variable > Monitor [Переменная > Наблюдать]**.
- Вы можете обновить значения выбранных переменных немедленно с помощью команды меню **Variable > Update Monitor Values [Переменная > Обновить наблюдаемые значения]**. В таблице переменных отображаются текущие значения выбранных переменных.

Отмена наблюдения с помощью ESC

Если при активной функции наблюдения вы нажмете клавишу ESC, то функция завершает работу без запроса.

18.7.2 Определение запуска для наблюдения переменных

Вы можете отобразить на устройстве программирования текущие значения отдельных переменных в программе пользователя в конкретной точке обработки программы (точке запуска), чтобы наблюдать за ними.

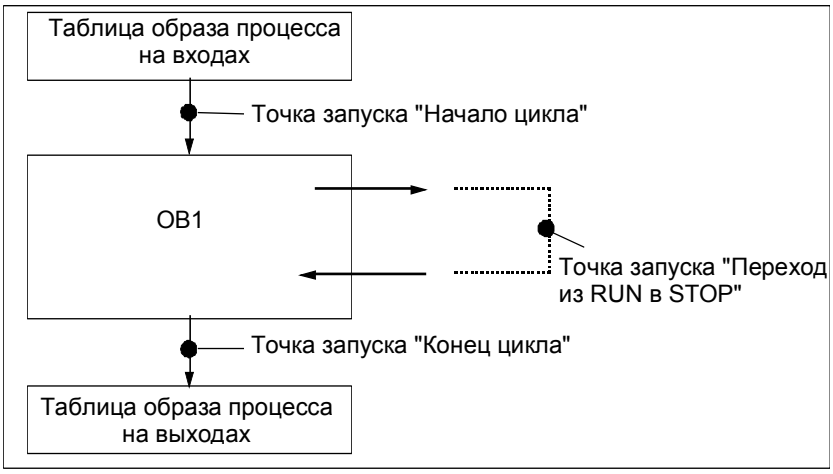
Выбирая точку запуска, вы определяете момент времени, в который будут отображаться наблюдаемые значения переменных.

Точка запуска и частота запуска устанавливаются с помощью команды меню **Variable > Trigger [Переменная > Запуск]**.

Запуск	Возможные установки
Trigger point [Точка запуска]	Start of cycle [Начало цикла] End of cycle [Конец цикла] Transition from RUN to STOP [Переход из RUN в STOP]
Trigger frequency [Частота запуска]	Once [Один раз] Every cycle [Каждый цикл]

Точка запуска

На следующем рисунке показано положение точек запуска.



Если вы установили одну и ту же точку запуска для наблюдения и изменения, то наблюдаемое значение отображается до его изменения, так как функция наблюдения исполняется **раньше**, чем функция изменения. Для отображения измененного значения вам следует установить в качестве точки запуска для наблюдения "Начало цикла", а в качестве точки запуска для изменения "Конец цикла".

Немедленный запуск

Значения выбранных переменных можно обновить с помощью команды меню **Variable > Update Monitor Values [Переменная > Обновить наблюдаемые значения]**. Эта команда подразумевает "немедленный запуск" и исполняется так быстро, насколько это возможно, безотносительно к какой-либо точке в программе пользователя. Эти функции используются, главным образом, для наблюдения и изменения переменных в состоянии STOP.

Частота запуска

Следующая таблица показывает влияние частоты запуска на наблюдение переменных:

	Частота запуска: один раз	**Частота запуска: каждый цикл**
Наблюдаемые переменные	*Обновляются один раз* Зависит от точки запуска	*Наблюдение с определенной точкой запуска* При тестировании блока вы можете точно проследить ход процесса обработки.

18.8 Изменение переменных

18.8.1 Введение в изменение переменных

Для изменения переменных в вашем распоряжении имеются следующие методы:

- Активизируйте функцию изменения переменных с помощью команды меню **Variable > Modify [Переменная > Изменить]**. Программа пользователя применяет заданные значения к выбранным переменным из таблицы переменных в соответствии с точкой и частотой запуска. Если вы установили частоту запуска ”Every cycle [Каждый цикл]", то вы снова можете отключить функцию изменения переменных командой меню **Variable > Modify [Переменная > Изменить]**.
- Вы можете обновить значения выбранных переменных немедленно с помощью команды меню **Variable > Activate Modify Values [Переменная > Активизировать изменение значений]**.

Функции Force [Принудительное задание значений] и Enable Peripheral Output (PQ) [Деблокировка периферийного выхода] предоставляют другие возможности.

При изменении значений переменных примите во внимание:

- Изменяются только те адреса, которые были видны в таблице переменных, когда вы начали изменение.
  Если вы уменьшили размер видимой области таблицы переменных сразу после того, как вы запустили функцию изменения, могут быть изменены адреса, которые более не видны.
  Если видимая область таблицы переменных увеличивается, то возможны видимые адреса, которые не изменяются.
- Изменение не может быть отменено (например, с помощью **Edit > Undo[Редактировать > Отменить]**).
- Если вы выбрали изменение в каждом цикле, то вы не сможете продвигать информацию на экране.

Опасность
Изменение значений переменных при выполнении процесса может привести к нанесению серьезного ущерба имуществу или здоровью персонала при возникновении ошибок в действиях или в программе. Перед выполнением функции изменения переменных убедитесь в невозможности возникновения опасных ситуаций.

Отмена функции изменения переменных с помощью ESC

Если при работе функции изменения переменных вы нажмете ESC, то функция прекращает работу без запроса.

18.8.2 Определение запуска для изменения переменных

Вы можете назначить фиксированные значения отдельным переменным программы пользователя (один раз или каждый цикл) в конкретной точке обработки программы (точке запуска).

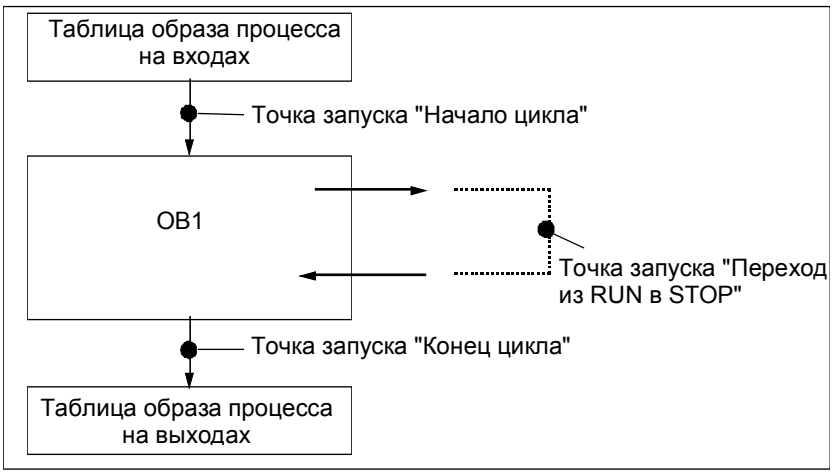
Выбирая точку запуска, вы определяете момент времени, в который измененные значения присваиваются переменным.

Точку запуска и частоту запуска вы можете установить с помощью команды меню **Variable > Trigger [Переменная > Запуск]**.

Запуск	Возможные установки
Trigger point [Точка запуска]	Start of cycle [Начало цикла] End of cycle [Конец цикла] Transition from RUN to STOP [Переход из RUN в STOP]
Trigger frequency [Частота запуска]	Once [Один раз] Every cycle [Каждый цикл]

Точка запуска

На следующем рисунке показано положение точек запуска.



Если вы установили одну и ту же точку запуска для наблюдения и изменения, то наблюдаемое значение отображается до его изменения, так как функция наблюдения исполняется **раньше**, чем функция изменения. Для отображения измененного значения вам следует установить в качестве точки запуска для наблюдения "Начало цикла", а в качестве точки запуска для изменения "Конец цикла".

Для точек запуска при изменении переменных имеет силу следующее:

- Если в качестве частоты запуска вы установили ”Once [Один раз]", то при невозможности изменения выбранных переменных появляется соответствующее сообщение.
- При частоте запуска ”Every cycle [Каждый цикл]" никаких сообщений не появляется.

Немедленный запуск

Значения выбранных переменных можно изменить с помощью команды меню **Variable > Activate Modify Values [Переменная > Активизировать изменение значений]**. Эта команда подразумевает "немедленный запуск" и исполняется так быстро, насколько это возможно, безотносительно к какой-либо точке в программе пользователя. Эти функции используются, главным образом, для наблюдения и изменения переменных в состоянии STOP.

Частота запуска

Следующая таблица показывает влияние установленных условий запуска на изменение значений переменных:

	Частота запуска: один раз	**Частота запуска: каждый цикл**
Изменение переменных	*Активизируется один раз* Вы можете присвоить значения переменным один раз, независимо от точки запуска.	*Изменение с определенной точкой запуска* Назначая фиксированные значения, вы можете имитировать определенные ситуации для своей пользовательской программы и использовать это для отладки функций, которые вы запрограммировали.

18.9 Принудительное присваивание значений переменным

18.9.1 Введение в принудительное присваивание значений переменным

Вы можете присвоить отдельным переменным программы пользователя фиксированные значения так, что они не могут быть изменены или переписаны даже пользовательской программой, исполняющейся в CPU. Предпосылкой для этого является то, что CPU поддерживает эту функцию (например, CPU S7-400). Присваивая фиксированные значения переменным, вы можете установить конкретные ситуации для своей пользовательской программы и использовать это для тестирования запрограммированных функций.

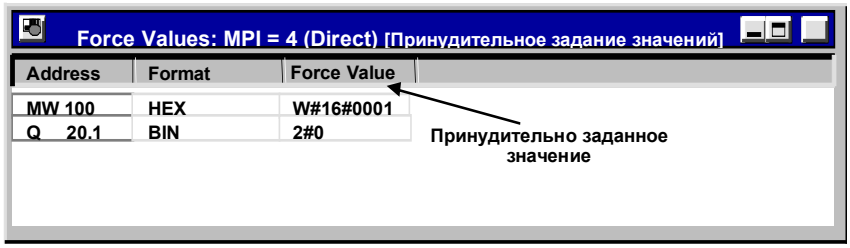
Окно "Force Values [Задать значения принудительно]"

Команды для принудительного задания значений могут быть выбраны только в том случае, если активизировано окно "Force Values [Задать значения принудительно]".

Чтобы отобразить это окно, выберите команду меню **[Переменная > Отобразить принудительное задание значений]**.

Для CPU вы должны открыть только одно окно "Force Values [Задать значения принудительно]". В этом окне отображаются переменные вместе с соответствующими принудительно заданными значениями для активного задания принудительных значений.

Пример окна принудительного задания значений



Имя текущей связи online показано в **строке заголовка**.

Дата и время, когда задание на принудительное присваивание значений было считано из CPU, показаны в **строке состояний**.

Когда отсутствуют активные задания на принудительное присваивание значений, окно пусто.

Различные методы **отображения переменных** в окне "Force Values [Принудительное задание значений]" имеют следующий смысл:

Отображение	Значение
Полужирное:	Переменные, которым уже назначено фиксированное значение в CPU.
Нормальное:	Редактируемые переменные.
Серого цвета:	Переменные модуля, который отсутствует / не вставлен в стойку или Переменные с ошибочным адресом; выводится сообщение об ошибке.

Использование принудительно назначаемых адресов из таблицы переменных

При открытии окна "Force Values [Принудительное задание значений]" выводится сообщение, и вы можете решить, хотите ли вы использовать в окне "Force Values" принудительно задаваемые адреса из таблицы переменных. Принудительно задаваемые адреса – это адреса, которым могут быть присвоены фиксированные значения. Вы можете ввести не более 512 принудительно задаваемых адресов.

Использование задания на принудительное присваивание значений из CPU или создание нового задания

Если окно "Force Values [Принудительное задание значений]" открыто и активно, то выводится еще одно сообщение:

- Если вы его подтверждаете, то изменения в этом окне переписываются заданием на принудительное присваивание значений, существующим в CPU. Вы можете восстановить предыдущее содержимое окна с помощью команды меню **Edit > Undo [Редактировать > Отменить]**.
- Если вы его отменяете, то текущее содержимое окна сохраняется. Затем вы можете сохранить содержимое окна "Force Values [Принудительное задание значений]" в виде таблицы переменных с помощью команды меню **Table > Save As [Таблица > Сохранить как...]** или выбрать команду меню **Variable > Force [Переменная > Принудительно присвоить]**: она записывает текущее содержимое окна в CPU как новое задание на принудительное присваивание значений.

Наблюдение и изменение переменных возможно только в таблице переменных, а не в окне "Force Values [Принудительное задание значений]".

Сохранение окна принудительно задаваемых значений

Вы можете сохранить содержимое окна принудительно задаваемых значений в таблице переменных. С помощью команды меню **Insert > Variable Table [Вставить > Таблица переменных]** вы можете повторно вставить это сохраненное содержимое в окно принудительно задаваемых значений.

Замечания о символах в окне принудительно задаваемых значений

Символы вводятся в последнее активное окно за исключением случая, если вы открыли приложение "Monitoring and Modifying Variables [Наблюдение и изменение переменных]" из другого приложения, в котором нет символов.

Если вы не можете ввести символические имена, то столбец "Symbol [Символ]" остается скрытым. Команда меню **Options > Symbol Table [Параметры > Таблица символов]** в этом случае деактивирована.

18.9.2 Соблюдайте меры безопасности при принудительном задании значений переменных

Остерегайтесь нанесения вреда персоналу и повреждения имущества

Имейте в виду, что при использовании функции принудительного задания значений любое неправильное действие может:

- подвергнуть опасности жизнь или здоровье персонала или
- вызвать повреждение отдельных механизмов или всего оборудования.

Предостережение

- Перед запуском функции принудительного задания значений вы должны проверить, что никто не выполняет эту функцию на том же CPU в то же самое время.
- Задание на принудительное присваивание значений может быть удалено или завершено только с помощью команды меню **Variable > Stop Forcing [Переменная > Прекратить принудительное задание значений]**. Закрытие окна для принудительного задания значений или выход из приложения ”Monitoring and Modifying Variables [Наблюдение и изменение переменных]" не удаляет задание на принудительное присваивание значений.
- Принудительное присваивание значений не может быть отменено (например, с помощью **Edit > Undo [Редактировать > Отменить]**).
- Прочтите информацию о различиях между принудительным заданием и изменением значений переменных.
- Если CPU не поддерживает функцию принудительного присваивания значений, то все команды в меню Variable [Переменная], связанные с принудительным заданием значений, деактивированы.

Если деактивирована блокировка выходов с помощью команды меню **Variable > Enable Peripheral Outputs [Переменная > Деблокировать периферийные выходы]**, то все модули вывода, к которым применена функция принудительного задания значений, выдают свои принудительно заданные значения.

18.9.3 Различия между принудительным заданием и изменением значений переменных

Следующая таблица подводит итог различиям между принудительным заданием и изменением значений переменных:

Свойство/ Функция	Принудительное задание у S7-400	Принудительное задание у S7-300	Изменение
Меркеры (M)	●	–	●
Таймеры и счетчики (T, C)	–	–	●
Блоки данных (DB)	–	–	●
Периферийные входы (PIB, PIW, PID)	●	–	–
Периферийные выходы (PQB, PQW, PQD)	●	–	●
Входы и выходы (I, Q)	●	●	●
Установка запуска	всегда немедленный запуск	всегда немедленный запуск	один раз или каждый цикл
Функция действует только на переменные в видимой области активного окна	действует на все принудительно задаваемые значения	действует на все принудительно задаваемые значения	●
Программа пользователя может переписать измененные/принудительно заданные значения	–	●	●
Замена принудительно заданного значения эффективна без прерывания	●	●	–
Переменные сохраняют свои значения при завершении приложения	●	●	–
Переменные сохраняют свои значения после обрыва связи с CPU	●	●	–
Адресация ошибок разрешена: напр. IW1 измененное/принудительно заданное значение: 1 IW1 измененное/принудительно заданное значение: 0	–	–	Последнее становится эффектив-ным

Замечание

Если периферийные выходы деблокированы с помощью "Enable Peripheral Outputs", то принудительно заданные значения для периферийных выходов, к которым применена функция принудительного задания, становятся эффективными на соответствующих модулях вывода; однако это не относится к функции изменения значений для периферийных выходов.

В случае принудительного задания значений переменная всегда имеет заданное значение. Это значение считывается при каждом обращении на чтение к программе пользователя. Все формы обращения для записи не действуют.

При непрерывном изменении обращение на чтение к программе эффективно и остается таким до следующей точки запуска.

19 Тестирование с использованием статуса программы

19.1 Тестирование с использованием статуса программы

Вы можете тестировать свою программу, выводя на экран статус программы (RLO (или VKE), бит состояния) или содержимое соответствующих регистров для каждой команды. Вы можете определить объем отображаемой информации в закладке "LAD/FBD (KOP/FUP)" диалогового окна "Customize [Настройка]". Это диалоговое окно открывается с помощью команды меню **Options > Customize [Параметры > Настройка]** в окне "LAD/STL/FBD: Programming Blocks [KOP/AWL/FUP: Программирование блоков]".

Предупреждение
Тестирование программы при действующем процессе может привести в случае возникновения ошибок в действиях или в программе к нанесению существенного ущерба имуществу или здоровью персонала.

Перед выполнением этой функции убедитесь в невозможности возникновения опасных ситуаций.

Предпосылки

Для отображения статуса программы должны быть выполнены следующие требования:

- Вы должны были сохранить блок без ошибок, а затем загрузить его в CPU.
- CPU должен быть в рабочем состоянии, а программа пользователя исполняться.
- Блок должен быть открыт online.

Основная последовательность действий для наблюдения статуса программы

Настоятельно рекомендуется не вызывать для отладки всю программу, а вызывать по одному блоки и отлаживать их индивидуально. Начинать следует с блоков на последнем уровне вложения иерархии вызовов, например, путем вызова их из OB1 и создания среды тестирования для блока с помощью функции наблюдения и изменения переменных.

Для тестирования в статусе программы, для установки контрольных точек и для исполнения программы в пошаговом режиме должен быть установлен режим тестирования (см. команду меню **[Отладка > Режим]**). Эти функции тестирования невозможны в режиме обработки (process operation).

19.2 Отображение статуса программы

Отображение **статуса программы** обновляется циклически.

Предустановленные цветовые коды

- Статус выполняется: зеленые непрерывные линии
- Статус не выполняется: синие пунктирные линии
- Статус неизвестен: черные непрерывные линии

Предустановленный тип и цвет линий может быть изменен с помощью команды меню **Options > Customize [Параметры > Настройка]**, закладка "LAD/FBD (или KOP/FUP)".

Статус элементов

- Статус контакта:
  - выполняется, если операнд имеет значение "1"
  - не выполняется, если операнд имеет значение "0"
  - неизвестен, если неизвестно значение операнда.
- Статус элементов с деблокировкой выхода (ENO) соответствует статусу контакта со значением выхода ENO в качестве операнда.
- Статус элементов с выходом Q соответствует статусу контакта со значением адреса.
- Статус вызовов (CALL) выполняется, если после вызова устанавливается бит BR.
- Статус команды перехода выполняется, если производится переход, т. е. если выполнено условие перехода.
- Элементы с деблокировкой выхода (ENO) показываются черным цветом, если деблокирующий выход не подключен.

Статус линий

- Линии имеют черный цвет, если по ним не передается сигнал или их статус неизвестен.
- Статус линий, начинающихся у силовой шины, всегда выполняется ("1").
- Статус линий в начале параллельных ветвей всегда выполняется ("1").
- Статус линии, следующей за элементом, выполняется, если как статус линии перед элементом, так и статус элемента выполняются.
- Статус линии, следующей за NOT, выполняется, если статус линии перед NOT не выполняется (и наоборот).
- Статус линии **после** пересечения нескольких линий выполняется, если:
  - выполняется статус хотя бы одной линии **перед** пересечением
  - Статус линии перед ветвлением выполняется.

Статус параметров

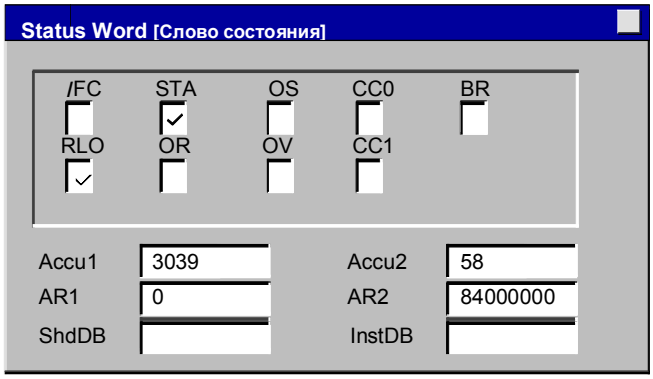
- Значения параметров, напечатанные **полужирным шрифтом**, являются текущими.
- Значения параметров, напечатанные тонкими линиями, являются результатом предыдущего цикла; этот раздел программы в текущем цикле не обрабатывался.

19.3 Что вам следует знать о тестировании в пошаговом режиме и о контрольных точках

При тестировании в пошаговом режиме вы должны делать следующее:

- выполнять программу оператор за оператором (отдельными шагами)
- установить контрольные точки

Функция "testing in single-step mode [тестирование в пошаговом режиме]" возможна не для всех программируемых контроллеров (обратитесь к документации для соответствующего программируемого контроллера).



Предпосылки

- Должен быть установлен режим тестирования. Тестирование в пошаговом режиме невозможно в режиме обработки (process operation) (см. команду меню **Debug > Operation [Отладка > Режим]**).
- Тестирование в пошаговом режиме возможно только для списка команд. Для блоков, представленных в виде контактного или функционального плана, вы должны изменить представление с помощью команды меню **View > STL [Вид > AWL (список команд)]**.
- Блок не должен быть защищен.
- Блок должен быть открыт online.
- Открытый блок не должен быть изменен в редакторе.

Количество контрольных точек

Количество контрольных точек переменно и зависит от следующего:

- количества уже установленных контрольных точек
- количества работающих статусов переменных
- количества работающих статусов программы

Обратитесь к документации на свой программируемый контроллер, чтобы выяснить, поддерживает ли он тестирование в пошаговом режиме.

Вы найдете команды меню, которые вы можете использовать для установки, активизации или удаления контрольных точек, в меню "Debug [Отладка]". Вы можете также выбрать эти команды меню с помощью пиктограмм на панели контрольных точек. Выведите на экран панель контрольных точек с помощью команды меню **View > Breakpoint Bar [Вид > Панель контрольных точек]**.

Разрешенные тестовые функции

- Наблюдение/изменение переменных
- Информация о модуле
- Режим работы

Опасность
Риск опасного состояния установки в режиме HOLD.

19.4 Что вам следует знать о режиме HOLD

Если программа наталкивается на контрольную точку, то программируемый контроллер переходит в режим HOLD [приостановка].

Светодиодная индикация в режиме HOLD

- Светодиод RUN мигает
- Светодиод STOP светится

Обработка программы в режиме HOLD

- В режиме HOLD код S7 не обрабатывается, т. е. более не обрабатываются никакие классы приоритета.
- Все таймеры заморожены:
  - таймерные ячейки не обрабатываются
  - все времена наблюдения приостановлены
  - основная тактовая частота уровней, управляемых временем, приостановлена
- Часы реального времени продолжают работать
- По соображениям безопасности в режиме HOLD выходы всегда заблокированы ("output disable [блокировка выходов]").

Поведение при отказе источника питания в режиме HOLD

- Программируемые контроллеры с батарейным резервированием переходят в состояние STOP и остаются в нем. CPU не производит автоматического перезапуска. Из состояния STOP вы можете определить, как следует продолжать обработку (например, путем установки/сброса контрольных точек, выполнения ручного перезапуска).
- Программируемые контроллеры без батарейного резервирования "не имеют памяти" и поэтому выполняют автоматический теплый рестарт при восстановлении питания независимо от предшествующего режима работы.

20 Тестирование с использованием программы моделирования (дополнительный пакет)

20.1 Тестирование с использованием программы моделирования (дополнительный пакет)

С помощью дополнительного пакета программ PLC Simulation [Моделирование ПЛК] вы можете выполнять и тестировать вашу программу на имитируемом программируемом контроллере, который существует в вашем компьютере или программаторе (например, PG 740). Поскольку имитация полностью реализуется программным обеспечением STEP 7, вам не требуются никакие аппаратные средства S7 (или сигнальные модули). Используя имитируемый CPU S7, вы можете тестировать и выявлять ошибки в программах для CPU S7-300 и S7-400.

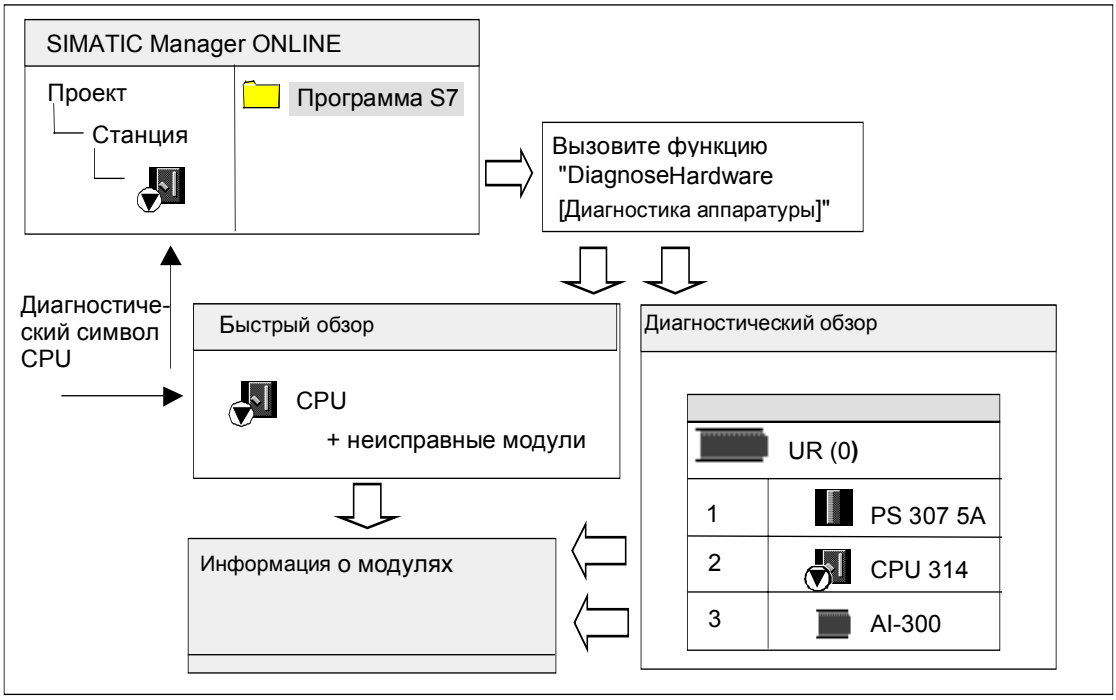
Это приложение обеспечивает простой пользовательский интерфейс для контроля и изменения различных параметров, которые используются в вашей программе (например, для включения и выключения входов). Вы можете также использовать различные приложения из программного обеспечения STEP 7 в то время, как ваша программа обрабатывается имитируемым CPU. Например, вы можете наблюдать и изменять переменные с помощью таблицы переменных (VAT).

21 Диагностика

21.1 Диагностика аппаратных средств и поиск неисправностей

По присутствию диагностических символов вы можете судить, является ли диагностическая информация доступной для модуля. Диагностические символы показывают состояние соответствующего модуля, а для CPU также режим работы.

Диагностические символы отображаются в окне проекта в представлении online, а также в быстром обзоре (установка по умолчанию) или в диагностическом обзоре, когда вы вызываете функцию "Diagnose Hardware [Диагностика аппаратуры]". Подробная диагностическая информация отображается в приложении "Module Information [Информация о модуле]", которое вы можете запустить двойным щелчком по диагностическому символу в быстром обзоре или в диагностическом обзоре.



Как установить местоположение неисправностей

1. Откройте окно проекта online при помощи команды меню **View > Online [Вид > Online]**.
2. Откройте все станции так, чтобы были видимы сконфигурированные в них программируемые модули.
3. Выясните, какой CPU отображает диагностический символ, указывающий на ошибку или неисправность. С помощью клавиши F1 вы можете открыть справочную страницу с объяснением диагностических символов.
4. Выберите станцию, которую вы хотите проверить.
5. Выберите команду меню **PLC > Module Information [ПЛК > Информация о модуле]**, чтобы отобразить информацию для CPU в этой станции.
6. Выберите команду меню **PLC > Diagnose Hardware [ПЛК > Диагностика аппаратуры]**, чтобы отобразить вывести на экран "быстрый обзор" с CPU и неисправными модулями этой станции. Отображение быстрого обзора установлено по умолчанию (команда меню **Options > Customize [Параметры > Настройка]**, вкладка "View" [Вид]).
7. Выберите в быстром обзоре неисправный модуль.
8. Щелкните на кнопке "Module Information [Информация о модуле]", чтобы получить информацию об этом модуле.
9. Щелкните по кнопке "Open Station Online [Открыть станцию online]" в быстром обзоре, чтобы отобразить диагностический обзор. Диагностический обзор содержит все модули станции в порядке расположения их слотов.
10. Дважды щелкните по модулю в диагностическом обзоре, чтобы отобразить информацию о нем. Этим способом вы можете получить информацию также для тех модулей, которые не вышли из строя и поэтому не отображаются в быстром обзоре.

Вам не обязательно нужно выполнять все эти шаги; вы можете остановиться, как только получите требующуюся вам диагностическую информацию.

21.2 Диагностические символы в представлении online

Диагностические символы отображаются в окне проекта online и в окне конфигурации аппаратных средств с конфигурационными таблицами в режиме online.

Диагностические символы облегчают процесс обнаружения неисправности. По символу модуля вы можете наглядно видеть, имеется ли диагностическая информация. Если неисправностей нет, то символы типов модулей отображаются без дополнительных диагностических символов.

Если диагностическая информация для модуля имеется, то в дополнение к символу модуля отображается диагностический символ, либо символ модуля отображается с пониженной контрастностью.

Диагностические символы для модулей (Пример: FM / CPU)

Символ	Значение
	Несоответствие предварительно установленной и фактической конфигурации: сконфигурированный модуль недоступен или вставлен модуль отличающегося типа.
	Отказ: Модуль неисправен. Возможные причины: диагностическое прерывание, ошибка доступа для ввода/вывода или обнаружен светодиод ошибки.
 CPU	Диагностика невозможна, потому что не существует соединения online или CPU не может обеспечить диагностическую информацию для модуля (например, источника питания или субмодуля).

Диагностические символы для режимов работы (Пример: CPU)

Символ	Режим
	STARTUP [запуск]
	STOP
	STOP, вызванный режимом STOP в другом CPU при многопроцессорной обработке
	RUN
	HOLD [приостановка]

Диагностический символ для принудительного задания значений

Символ	Режим
	В этом модуле переменным в программе пользователя назначаются фиксированные значения, которые не могут изменяться программой. Символ принудительного задания значений может появиться также в сочетании с другими символами (здесь с символом режима RUN).

Обновление отображения диагностических символов

Должно активироваться соответствующее окно.

- Нажмите клавишу F5 или
- выберите в окне команду меню **View > Update [Вид > Обновить]**.

21.3 Диагностика аппаратных средств: Быстрый обзор

21.3.1 Вызов быстрого обзора

Быстрый обзор предоставляет быстрый способ использования диагностики аппаратуры "Diagnosing Hardware" с меньшим количеством информации, чем более подробное отображение в диагностическом обзоре HWConfig. Когда вызывается функция "Diagnose Hardware [Диагностика аппаратуры]", по умолчанию отображается быстрый обзор.

Отображение быстрого обзора

Вы вызываете эту функцию из SIMATIC Manager, используя команду меню **PLC > Diagnose Hardware [ПЛК > Диагностика аппаратуры]**.

Вы можете использовать эту команду меню следующим образом:

- В окне проекта online, если выбран модуль или программа S7/M7.
- Если в окне "Accessible Nodes [Доступные узлы]" выбран узел ("MPI=...") и этот вход принадлежит CPU.

Вы можете выбирать модули, информацию о которых вы желаете вывести на дисплей, в отображаемых конфигурационных таблицах.

21.3.2 Информационные функции в быстром обзоре

В быстром обзоре отображается следующая информация:

- Данные для соединения online с CPU.
- Диагностический символ для CPU.
- Диагностические символы для модулей, в которых CPU обнаружил неисправность (например, диагностическое прерывание, ошибка доступа для ввода/вывода).
- Тип модуля и адрес модуля (стойка, слот, master-система DP с номером станции).

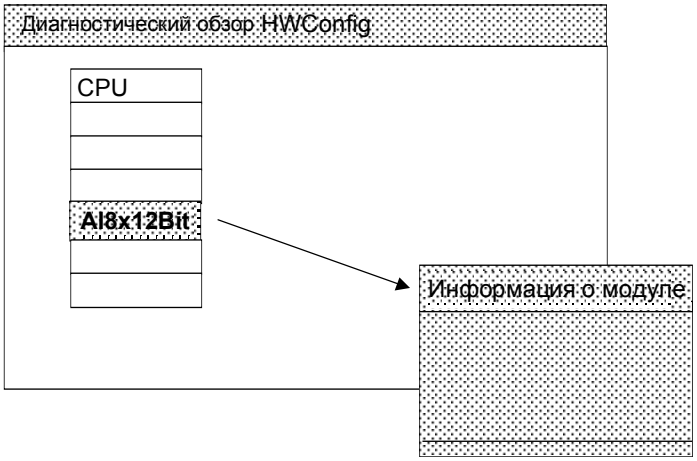
Другие диагностические возможности в быстром обзоре

- **Отображение информации о модулях**
  Вы можете вызывать это диалоговое окно щелчком на кнопке "Module Information [Информация о модуле]". Это диалоговое окно отображает подробную диагностическую информацию, зависящую от диагностических возможностей выбранного модуля. В частности, вы можете отображать введенные в диагностический буфер данные через диагностическую информацию CPU.
- **Отображение диагностического обзора**
  При помощи кнопки "Open Station Online [Открыть станцию online]" вы можете открыть диалоговое окно, которое, в отличие от быстрого обзора, содержит графический обзор целой станции, а также информацию о конфигурации. Он фокусирует внимание на модуле, высвеченном в списке "CPU/Faulty Modules [CPU/Неисправные модули]".

21.4 Диагностика аппаратных средств: Диагностический обзор

21.4.1 Вызов диагностического обзора

Используя этот метод, вы можете открыть диалоговое окно "Module Information [Информация о модуле]" для всех модулей в стойке. Диагностический обзор (конфигурационная таблица) показывает фактическую структуру станции на уровне стоек и станций децентрализованной периферии вместе с их модулями.



Примечание

Если конфигурационная таблица уже открыта offline, то вы можете также получить ее представление в режиме online, используя команду меню **Station > Open Online [Станция > Открыть online]**.

В зависимости от диагностических возможностей модуля, в диалоговом окне ”Module Information [Информация о модуле]" отображается различное количество вкладок.

В окне "Accessible Nodes [Доступные узлы]" модули всегда видны только с адресом их собственного узла (адресом MPI или PROFIBUS).

Вызов из представления проекта online в SIMATIC Manager

11. Установите соединение online с программируемым контроллером, используя команду меню **View > Online [Вид > Online]** в отображении проекта в SIMATIC Manager.
12. Выберите станцию и откройте ее двойным щелчком.
13. Затем откройте в ней объект "Hardware [Аппаратные средства]". Открывается диагностический обзор.

Теперь вы можете выбрать модуль и запросить информацию о нем при помощи команды меню **PLC > Module Information [ПЛК > Информация о модуле]**.

Вызов из представления проекта offline в SIMATIC Manager

Выполните следующие шаги:

1. Выберите станцию из представления проекта в SIMATIC Manager и откройте ее двойным щелчком.
2. Затем откройте в ней объект "Hardware [Аппаратные средства]". Открывается конфигурационная таблица.
3. Выберите команду меню **Station > Open Online [Станция > Открыть online]**.
4. Открывается диагностический обзор HW Config с конфигурацией станции, определенной из модулей (например, CPU). Состояние модулей указывается посредством символов. Для выяснения значений различных символов обратитесь к оперативной помощи. Неисправные модули и отсутствующие сконфигурированные модули перечисляются в отдельном диалоговом окне. Из этого диалогового окна вы можете перейти непосредственно к одному из выбранных модулей (кнопка ”Go To [Перейти]”).
5. Дважды щелкните на символе модуля, состояние которого вас интересует. Диалоговое окно с вкладками (зависящими от типа модуля) даст подробный анализ состояния модуля.

Вызов из окна "Accessible Nodes" в SIMATIC Manager

Выполните следующие шаги:

1. Откройте окно "Accessible Nodes [Доступные узлы]" в SIMATIC Manager, используя команду меню **PLC > Display Accessible Nodes [ПЛК > Отобразить доступные узлы]**.
2. Выберите узел в окне "Accessible Nodes [Доступные узлы]".
3. Выберите команду меню **PLC > Diagnose Hardware [ПЛК > Диагностика аппаратных средств]**.

Примечание

В окне "Accessible Nodes [Доступные узлы]" модули всегда видны только с адресом их собственного узла (адресом MPI или PROFIBUS).

21.4.2 Информационные функции в диагностическом обзоре

В отличие от быстрого обзора, диагностический обзор отображает конфигурацию всей станции, доступную в режиме online. Он содержит:

- Конфигурации стоек.
- Диагностические символы для **всех** сконфигурированных модулей. По ним вы можете прочитать состояние каждого модуля, а в случае модулей CPU – режим работы.
- Тип модуля, заказной номер и подробности адреса, комментарии к конфигурации.

Дополнительные диагностические возможности диагностического обзора

Двойным щелчком по модулю вы можете отобразить режим работы этого модуля.

21.5 Вызов информации о модулях

21.5.1 Функции информации о модулях

Функции информации о модулях можно найти в различных вкладках в пределах диалогового окна "Module Information". При отображении в активной ситуации отображаются только вкладки, существенные для выбранного модуля.

Функция/Вкладка	Информация	Использование
General [Общие сведения]	Идентификационные данные выбранного модуля; например, заказной номер, номер редакции, состояние, слот в стойке.	Информация в режиме online от вставленного модуля может сравниваться с данными для сконфигурированного модуля
Diagnostic Buffer [Диагностический буфер]	Краткий обзор событий в диагностическом буфере и подробная информация о выбранном событии.	Для нахождения причины останова CPU и анализа ведущих к этому событий в выбранном модуле. Используя диагностический буфер, можно анализировать ошибки в системе и в более позднее время для того, чтобы найти причину перехода в STOP или проследить возникновение отдельных диагностических событий и классифицировать их.
Diagnostic Interrupt [Диагностическое прерывание]	Диагностические данные для выбранного модуля.	Для оценки причины отказа модуля.
DP Slave Diagnostics [Диагностика Slave-устройства DP]	Диагностические данные для выбранного Slave-устройства DP (в соответствии с EN 50170).	Для оценки причины отказа Slave-устройства DP.

Функция/Вкладка	Информация	Использование
Memory [Память]	Текущее использование рабочей памяти и загрузочной памяти выбранного CPU или функционального модуля M7.	Для проверки перед переносом в CPU новых или расширенных блоков, достаточно ли доступной загрузочной памяти в CPU/ функциональном модуле, или для сжатия содержимого памяти.
Scan Cycle Time [Время цикла сканирования]	Продолжительность самого длинного, самого короткого и последнего циклов сканирования выбранного CPU или функционального модуля M7.	Для контроля сконфигурированного минимального времени цикла, максимального и текущего времени цикла.
Time System [Система времени]	Текущее время, часы работы и информация о тактовых импульсах (интервалы синхронизации).	Для отображения и установления времени и даты модуля и проверки синхронизации времени.
Performance Data [Эксплуатационные данные]	Конфигурация памяти, области адресов и доступные блоки для выбранного модуля (CPU/FM).	Для проверки перед созданием и во время создания программы пользователя, удовлетворяет ли CPU требованиям для выполнения программы пользователя; например, размер загрузочной памяти или размер образа процесса.
Blocks [Блоки] (может открываться из вкладки ”Performance Data [Эксплуатаци-онные данные])	Отображение всех типов блоков, доступных в комплекте поставки выбранного модуля. Список OB, SFB и SFC, которые вы можете использовать для этого модуля.	Для проверки того, какие стандартные блоки может содержать или вызывать ваша программа пользователя, чтобы быть способной работать на выбранном CPU.
Communication [Связь]	Скорости передачи, обзор коммуникационных соединений, загрузка линии связи и максимальный размер кадра сообщения на коммуникационной шине выбранного модуля.	Для определения того, какие соединения с CPU или FM M7 и в каком количестве возможны и сколько из них используются.
Stacks [Стеки]	Вкладка **Stacks** [стеки]: Может вызываться только в режиме STOP или режиме HOLD. Отображается B-стек для выбранного модуля. Затем вы можете отобразить также I-стек, L-стек и стек вложений и перейти к местоположению ошибки в прерванном блоке.	Для определения причины перехода в STOP и исправления блока

Дополнительная отображаемая информация

В каждой вкладке отображается следующая информация:

- Путь online к выбранному модулю.
- Режим работы соответствующего CPU (например, RUN, STOP)
- Состояние выбранного модуля (например, ошибка, нормальное)
- Режим работы выбранного модуля (например, RUN, STOP), если у него есть свой собственный режим работы (например, CP 342-5).

Режим работы самого CPU и состояние выбранного модуля не могут отображаться, если информация для модуля, не являющегося CPU, открывается из окна "Accessible Nodes [Доступные узлы]".

Одновременное отображение нескольких модулей

Вы можете отображать информацию для нескольких модулей одновременно. Для этого вам нужно переключиться в соответствующий модульный контекст, выбрать другой модуль и затем вызвать информацию для него. Тогда отображается другое диалоговое окно ”Module Information". Для каждого модуля может открываться только одно диалоговое окно.

Обновление отображения информации о модулях

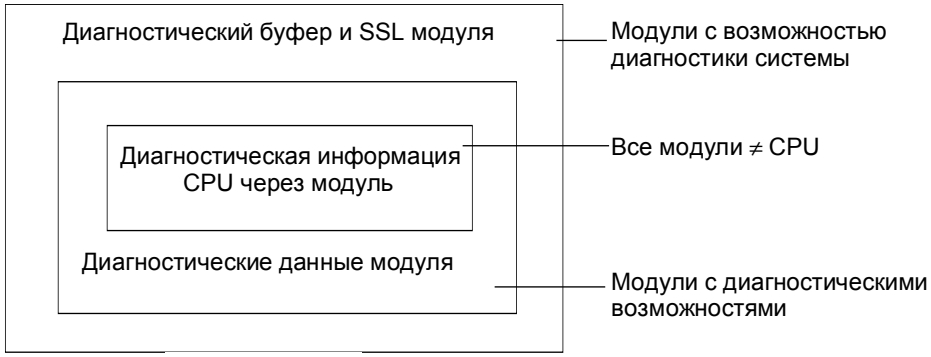
Каждый раз, когда вы в диалоговом окне "Module Information" переключаетесь на вкладку, данные снова считываются из модуля. Однако пока страница отображается, ее содержимое не обновляется. Вы можете считывать данные из модуля без переключения вкладки, щелкая по кнопке "Update [Обновить]".

21.5.2 Объем информации в зависимости от типа модуля

Объем информации, которая может оцениваться и отображаться, зависит от:

- выбранного модуля и
- вида представления, из которого вы запрашиваете информацию о модулях.
  Полный объем информации доступен при запросе из представления конфигурационных таблиц в режиме online или из окна проекта.
  При запросе из окна "Accessible Nodes [Доступные узлы]" доступен ограниченный объем информации.

В зависимости от объема информации модули делятся на категории "с возможностью диагностики системы", "с диагностическими возможностями" и "без диагностических возможностей ". Следующий рисунок показывает эти категории:



SSL – список состояний системы

- Модулями с возможностью диагностики системы являются, например, модули FM 351 и FM 354.
- Большинство модулей аналоговых сигналов – это модули с диагностическими возможностями.
- Большинство модулей цифровых сигналов – это модули без диагностических возможностей.

Отображаемые вкладки

Таблица показывает, какие вкладки со свойствами присутствуют в диалоговом окне ”Module Information [Информация о модуле]" для каждого типа модуля.

Вкладка	CPU или FM M7	Модуль с возможностью диагностики системы	Модуль с диагностическими возможностями	Модуль без диагностических возможностей	Slave-устройство DP
General [Общие данные]	да	да	да	да	да
Diagnostic Buffer [Диагностический буфер]	да	да	–	–	–
Diagnostic Interrupt [Диагностическое прерывание]	–	да	да	–	да
Memory [Память]	да	–	–	–	–
Scan Cycle Time [Время цикла сканирования]	да	–	–	–	–
Time System [Система времени]	да	–	–	–	–
Performance Data [Эксплуатационные данные]	да	–	–	–	–
Stacks [Стеки]	да	–	–	–	–
Communication [Связь]	да	–	–	–	–
DP Slave Diagnostics [Диагностика Slave-устройств DP]	–	–	–	–	да
H-состояние [1)]	да	–	–	–	–

[1)] Только для CPU в H-системах

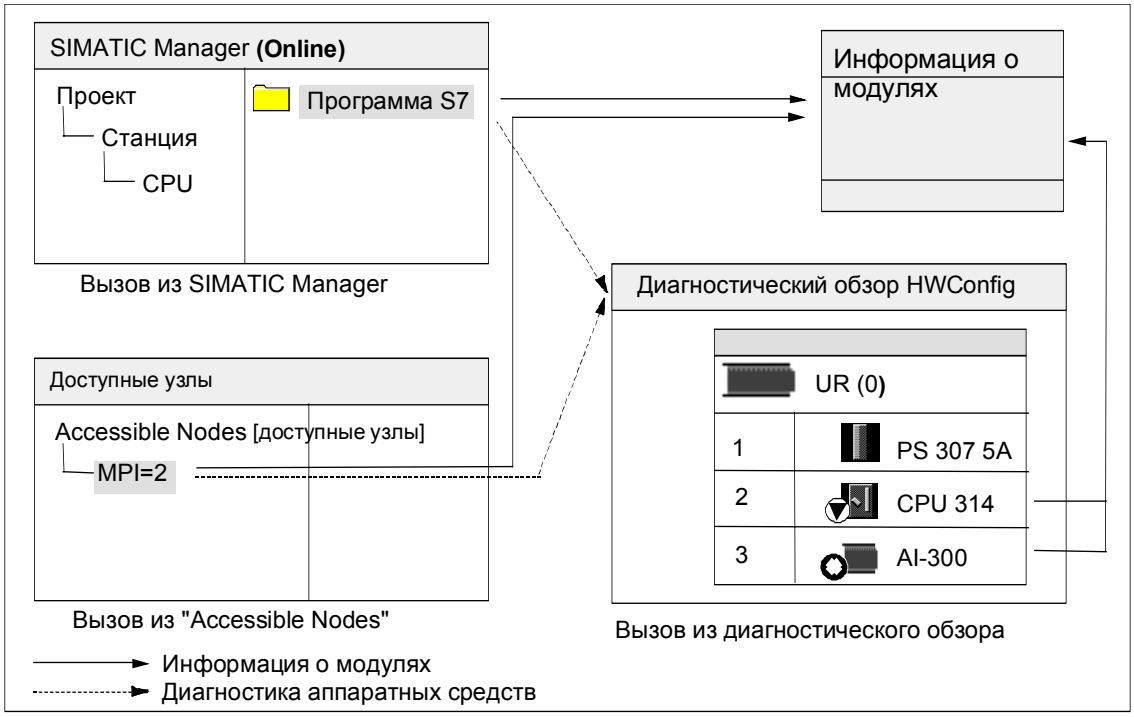
В дополнение к информации о свойствах, представленной на вкладках, для модулей, имеющих режим работы, отображается их режим работы. Когда вы открываете диалоговое окно из конфигурационных таблиц в режиме online, отображается состояние модуля с точки зрения CPU (например, OK, «отказ», «модуль недоступен»).

21.5.3 Запрос информации о модулях

21.5.3.1 Варианты отображения информации о модулях

Вы можете выводить на экран диалоговое окно "Module Information" из различных исходных точек. Следующие процедуры служат примерами часто используемых методов запроса информации о модулях:

- В SIMATIC Manager из окна с представлением проекта "online" или "offline".
- В SIMATIC Manager из окна "Accessible Nodes [Доступные узлы]".
- В диагностическом обзоре конфигурации аппаратных средств HW Config.



Чтобы отобразить состояние **модуля с адресом его собственного узла**, вам необходимо соединение online с программируемым контроллером. Вы устанавливаете это соединение через представление проекта в режиме online или через окно "Accessible Nodes [Доступные узлы]".

21.6 Диагностика в состоянии STOP

21.6.1 Основная последовательность действий для определения причины перехода в STOP

Чтобы определить, почему CPU перешел в режим "STOP", действуйте следующим образом:

1. Выделите CPU, перешедший в STOP.
2. Выберите команду меню **PLC > Module Information [ПЛК > Информация о модуле]**.
3. Выберите вкладку "Diagnostic Buffer [Диагностический буфер]".
4. Вы можете определить причину перехода в STOP по последним записям в диагностическом буфере.

Если встречается ошибка программирования:

5. Например, входное сообщение "STOP because programming error OB not loaded [STOP, так как OB обработки ошибок программирования не загружен]" означает, что CPU обнаружил ошибку в программе и затем попытался запустить (несуществующий) OB, чтобы обработать ошибку программирования. Предыдущие записи указывают на фактическую ошибку программирования.
6. Выберите сообщение, относящееся к ошибке программирования
7. Щелкните по кнопке "Open Block [Открыть блок]".
8. Выберите вкладку "Stacks [Стеки]".

21.6.2 Содержимое стеков в состоянии STOP

Оценивая содержимое диагностического буфера и стеков, вы можете определить причину сбоя в обработке программы пользователя.

Например, если CPU перешел в STOP вследствие ошибки программирования или команды STOP, то вкладка "Stacks [Стеки]" в информации о модулях отображает стек блоков. Вы можете отображать содержимое других стеков, используя кнопки ”I Stack [Стек прерываний]", ”L Stack [Локальный стек]" и ”Nesting Stack [Стек вложений]". Содержимое стека дает вам информацию о том, какая команда и в каком блоке привела к переходу CPU в STOP.

Содержимое B-стека

B-стек, или стек блоков, перечисляет все блоки, которые вызывались перед переключением в режим STOP и не были обработаны полностью.

Содержимое I-стека

Когда вы щелкаете по кнопке ”I Stack", отображаются данные в точке прерывания. I-стек, или стек прерываний, содержит данные или состояния, которые действовали в момент прерывания, например:

- Содержимое аккумуляторов и регистров
- Открытые блоки данных и их размер
- Содержимое слова состояния
- Класс приоритета (уровень вложения)
- Прерванный блок
- Блок, в котором продолжается обработка программы после прерывания.

Содержимое L-стека

Для каждого блока, указанного в B-стеке, вы можете отобразить соответствующие локальные данные, выбирая этот блок и щелкая на кнопке ”L Stack".

L-стек, или стек локальных данных, содержит значения локальных данных блоков, с которыми программа пользователя работала в момент прерывания.

Чтобы интерпретировать и оценивать отображаемые локальные данные, требуется глубокое знание системы. Первая часть отображаемых данных соответствует временным переменным блока.

Содержимое стека вложений

Когда вы щелкаете на кнопке ”Nesting Stack [Стек вложений]", отображается содержимое стека вложений в точке прерывания.

Стек вложений – это область памяти, которую используют логические операции **A(**, **AN(**, **O(**, **ON(**, **X(** и **XN(.**

Эта кнопка активна только тогда, когда в момент прерывания оставались открытыми скобочные выражения.

21.7 Проверка времен цикла сканирования во избежание временных ошибок

21.7.1 Проверка времен цикла сканирования во избежание временных ошибок

Вкладка "Scan Cycle Time [Время цикла сканирования]" в информации о модулях дает сведения о временах цикла сканирования программы пользователя.

Если продолжительность самого длинного цикла близка к сконфигурированному максимальному времени цикла, то имеется опасность того, что флуктуации времени цикла могут вызвать временную ошибку. Этого можно избежать, если вы увеличите максимальное время цикла (контрольное время) для программы пользователя.

Если длительность цикла меньше, чем сконфигурированное минимальное время цикла сканирования, то CPU/FM автоматически продлевает цикл до сконфигурированного минимального времени цикла сканирования. В случае CPU в течение этого продленного времени обрабатывается фоновый OB (OB90) (если он был загружен).

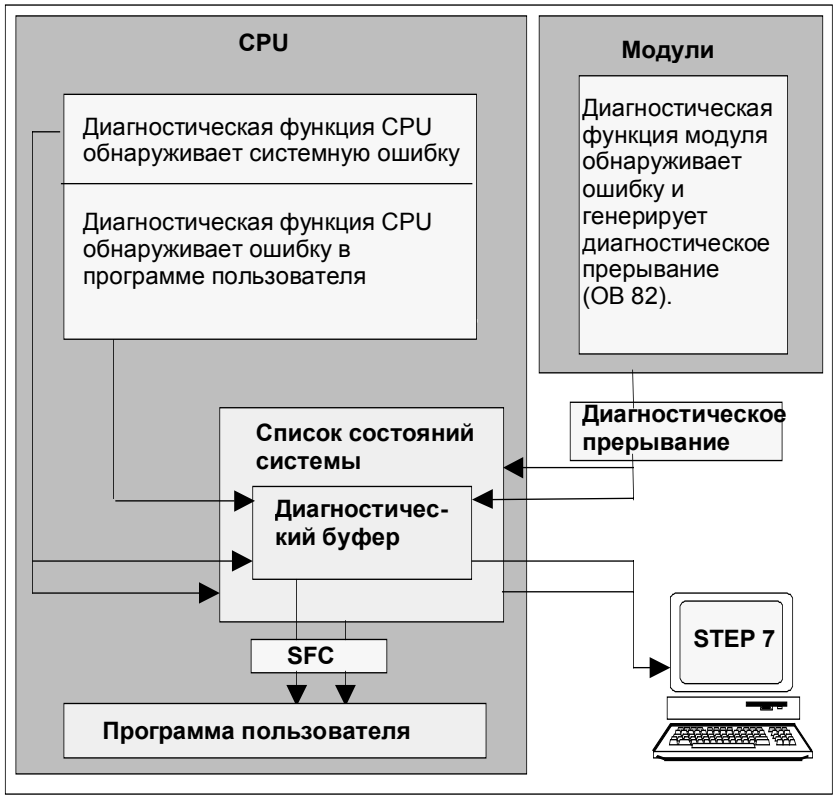
Настройка времени цикла сканирования

Вы можете устанавливать максимальное и минимальное время цикла, когда конфигурируете аппаратные средства. Для этого в представлении конфигурационной таблицы в режиме offline дважды щелкните на CPU/FM, чтобы определить его характеристики. Вы можете вводить соответствующие значения во вкладке ”Cycle/Clock Memory [Цикл/Тактовые меркеры]".

21.8 Поток диагностической информации

21.8.1 Поток диагностической информации

Следующий рисунок показывает поток диагностической информации в SIMATIC S7.



Отображение диагностической информации

Вы можете считывать диагностические записи, используя SFC51 RDSYSST в программе пользователя, или отображать диагностические сообщения на обычном языке с помощью STEP 7.

Они предоставляют информацию о следующем:

- Где и когда появилась ошибка
- Тип диагностического события, к которому относится запись (определяемое пользователем диагностическое событие, синхронная/ асинхронная ошибка, смена режима работы).

Генерирование групповых сообщений системы управления процессом

CPU вводит события стандартной диагностики и расширенной диагностики в диагностический буфер. Он также генерирует групповое сообщение системы управления процессом для стандартных диагностических событий, если выполняются следующие условия:

- Вы указали, чтобы сообщения системы управления процессом генерировались в STEP 7.
- По крайней мере, одно устройство отображения зарегистрировано в CPU для сообщений системы управления процессом.
- Групповое сообщение системы управления процессом генерируется только тогда, когда в текущий момент нет группового сообщения системы управления процессом соответствующего класса (имеется семь классов).
- На класс может генерироваться одно групповое сообщение управления процессом.

21.8.2 Список состояний системы (SSL)

Список состояний системы (SSL) описывает текущее состояние программируемого логического контроллера. Он предоставляет обзор конфигурации, текущего назначения параметров, текущих состояний и последовательностей в CPU и в принадлежащих ему модулях.

Вы можете данные из списка состояний системы только считывать, но не изменять. Это виртуальный список, создаваемый только по запросу.

Информацию, которую вы можете отображать, используя список состояний системы, можно подразделить на четыре области.

Считывание списка состояний системы

Есть два следующих способа считывания информации из списка состояний системы:

- Неявный, через команды меню STEP 7 из устройства программирования (например, конфигурация памяти, статические данные CPU, диагностический буфер, отображения статуса).
- Явный, через системную функцию SFC 51 RDSYSST в программе пользователя посредством ввода номера требуемого частного списка состояний системы (см. Help on Blocks [Справка о блоках]).

Системные данные из списка состояний системы

Системными данными являются собственные или назначаемые характеристические данные CPU. Следующая таблица показывает темы, по которым может отображаться информация (частные списки состояний системы):

Тема	Информация
Идентификация модулей	Заказной номер, идентификатор типа и версия модуля
Характеристики CPU	Система времени, характеристики системы (например, многопроцессорная обработка) и описание языка CPU
Области памяти	Конфигурация памяти модуля (объем рабочей памяти)
Системные области	Системная память модуля (например, количество меркеров, таймеров, счетчиков, тип памяти)
Типы блоков	Какие блоки (OB, DB, SDB, FC, FB) существуют в модуле, максимальное число блоков одного типа и максимальный размер блока каждого типа
Назначение прерываний и ошибок	Назначение прерываний/ошибок для OB
Состояние прерывания	Текущее состояние обработки прерываний/ сгенерированных прерываний
Состояние классов приоритета	Какой блок обрабатывается, какой класс приоритета заблокирован благодаря настройке параметров
Режим работы и переключение режима	Какие режимы работы возможны, последнее переключение режима работы, текущий режим работы

Диагностические данные состояния в CPU

Диагностические данные состояния описывают текущее состояние компонентов, контролируемых диагностикой системы. Следующая таблица показывает темы, по которым может отображаться информация (частные списки состояния системы):

Тема	Информация
Данные состояния связи	Все коммуникационные функции, установленные в системе в текущий момент времени
Диагностика модулей	Модули с диагностическими возможностями, зарегистрированные в CPU
Список стартовой информации OB	Стартовая информация об организационных блоках CPU
Список событий запуска	События запуска и классы приоритета OB
Информация о состоянии модулей	Информация о состоянии всех назначенных модулей, которые вставлены, неисправны или генерируют аппаратные прерывания

Диагностические данные модулей

Кроме CPU, имеются также другие модули с диагностическими возможностями (CM, CP, FM), данные которых вводятся в список состояний системы. Следующая таблица показывает темы, по которым может отображаться информация (частный список состояний системы):

Тема	Информация
Диагностическая информация модуля	Начальный адрес модуля, внутренние /внешние отказы, отказы каналов, ошибки параметров (4 байта)
Диагностические данные модуля	Все диагностические данные отдельного модуля

21.8.3 Передача ваших собственных диагностических сообщений

Вы можете также расширить стандартную диагностику системы SIMATIC S7, используя системную функцию SFC 52 WRUSMSG для того, чтобы:

- Вводить вашу собственную диагностическую информацию в диагностический буфер (например, информацию о выполнении программы пользователя).
- Передавать определяемые пользователем диагностические сообщения зарегистрированным станциям (контрольные устройства типа PG, OP или TD).

Диагностические события, определяемые пользователем

Диагностические события подразделяются на классы событий с 1 по F. Диагностические события, определяемые пользователем, принадлежат к событиям классов с 8 по B. Их можно разбить на две группы следующим образом:

- Классы событий 8 и 9 включают сообщения с фиксированным номером и предопределенным текстом, которые Вы можете вызывать по номеру.
- Классы событий A и B включают сообщения, которым Вы можете присваивать номер (с A000 по A0FF, с B000 по B0FF) и текст по вашему собственному выбору.

Передача диагностических сообщений станциям

Кроме создания определяемой пользователем записи в диагностический буфер, Вы можете также посылать ваши собственные, определяемые пользователем диагностические сообщения зарегистрированным устройствам отображения, используя SFC52 WRUSMSG. Когда SFC52 вызывается с SEND = 1, диагностическое сообщение записывается в буфер передачи и автоматически передается станции или станциям, зарегистрированным в CPU.

Если передавать сообщения невозможно (например, потому что не зарегистрировано никакое устройство отображения или потому что буфер передачи заполнен), то определяемое пользователем диагностическое событие все же вводится в диагностический буфер.

Генерирование сообщения с подтверждением

Если Вы подтверждаете определяемое пользователем событие диагностики и хотите записать подтверждение, то действуйте следующим образом:

- Когда событие входит в состояние события, записывайте 1 в переменную типа BOOL, а когда событие покидает состояние события, записывайте в эту переменную 0.
- Тогда вы сможете контролировать эту переменную, используя SFB33 ALARM.

21.8.4 Диагностические функции

Системная диагностика обнаруживает, анализирует и сообщает об ошибках, происходящих внутри программируемого контроллера. Для этого в каждом CPU и каждом модуле, обладающем возможностью диагностики системы (например, FM 354), имеется диагностический буфер, в который вводится подробная информация обо всех диагностических событиях в порядке их появления.

Диагностические события

В качестве диагностических событий отображаются, например, следующие входные сообщения:

- внутренние и внешние неисправности в модуле
- системные ошибки в CPU
- переключения режимов работы (например, из RUN в STOP)
- ошибки в программе пользователя
- вставка/снятие модулей
- сообщения пользователя, вводимые посредством системной функции SFC52

После сброса памяти содержимое диагностического буфера сохраняется. Используя диагностический буфер, можно анализировать ошибки в системе и в более позднее время с тем, чтобы найти причину перехода в STOP или проследить возникновение отдельных диагностических событий диагностики и классифицировать их.

Сбор диагностических данных

Вам нет нужды программировать сбор диагностических данных средствами диагностики системы. Это стандартный элемент, работающий автоматически. SIMATIC S7 предоставляет различные диагностические функции. Некоторые из этих функций встроены в CPU, а другие предоставляются модулями (SM, CP и FM).

Индикация неисправностей

Внутренние и внешние неисправности модулей отображаются на лицевых панелях модуля. Светодиодные дисплеи и их анализ описаны в руководствах по аппаратным средствам S7. В случае S7-300 внутренние и внешние неисправности отображаются совместно как групповая ошибка.

CPU распознает системные ошибки и ошибки в программе пользователе и вводит диагностические сообщения в список состояний системы и в диагностический буфер. Эти диагностические сообщения можно читать в программаторе.

Сигнальные и функциональные модули с диагностическими возможностями обнаруживают внутренние и внешние ошибки модулей и генерируют диагностическое прерывание, на которое вы можете реагировать при помощи OB прерываний.

21.9 Программные средства обработки ошибок

21.9.1 Программные средства обработки ошибок

Когда обнаруживаются ошибки в обработке программы (синхронные ошибки) и ошибки в программируемом контроллере (асинхронные ошибки), CPU вызывает соответствующий организационный блок (OB) для обработки ошибки:

Ошибка	OB для ошибки
Ошибка резервирования ввода/вывода	OB70
Ошибка резервирования CPU	OB72
Ошибка времени	OB80
Сбой источника питания	OB81
Диагностическое прерывание	OB82
Прерывание при вставке/снятии модуля	OB83
Отказ аппаратных средств CPU	OB84
Ошибка класса приоритета	OB85
Отказ стойки или отказ станции в децентрализованной периферии	OB86
Ошибка связи	OB87
Ошибка программирования	OB121
Ошибка доступа для ввода/вывода	OB122

Если соответствующий OB отсутствует, то CPU переходит в режим STOP. В противном случае в OB можно хранить команды относительно того, как он должен реагировать на такую сбойную ситуацию. Это означает, что воздействие ошибки можно уменьшить или устранить.

Основная последовательность действий

Создание и открытие OB

1. Отобразите информацию о модуле для вашего CPU.
2. Щелкните по кнопке "Blocks" [блоки] во вкладке "Performance Data [Эксплуатационные данные]".
3. На основе отображенного списка определите, разрешен ли OB, который вы хотите программировать, для данного CPU.
4. Вставьте OB в папку "Blocks" вашей программы и откройте OB.
5. Введите программу для обработки ошибки.
6. Загрузите OB в программируемый контроллер.

Программирование мер по обработке ошибок

1. Проанализируйте локальные данные OB, чтобы определить точную причину ошибки. Переменные OB8xFLTID и OB12XSWFLT в локальных данных содержат код ошибки. Их значение описано в справочном руководстве "Системные и стандартные функции”.
2. Перейдите к разделу программы, который реагирует на эту ошибку.

Вы найдете пример обработки диагностических прерываний в оперативной справочной информации System and Standard Functions [Системные и стандартные функции] под заголовком "Example of Module Diagnostics with SFC51 (RDSYSST) [Пример диагностики модуля при помощи SFC51 (RDSYSST)]".

Вы можете найти подробную информацию о блоках OB, SFB и SFC в соответствующей оперативной справке Help on Blocks [справка о блоках].

21.9.2 Анализ выходного параметра RET_VAL

Используя выходной параметр RET_VAL (возвращаемое значение), системная функция указывает, оказался ли CPU способным правильно выполнить функцию SFC или нет.

Информация об ошибках в возвращаемом значении

Возвращаемое значение имеет целочисленный тип данных (INT). Знак целого числа указывает, является ли оно положительным или отрицательным целым числом. Отношение возвращаемого значения к значению ”0" указывает, произошла ли при выполнении функции ошибка или нет (см. таблицу):

- Если во время выполнения функции происходит ошибка, то возвращаемое значение меньше, чем "0". Знаковый бит целого числа равен ”1".
- Если функция выполняется без ошибок, то возвращаемое значение больше, чем "0", или равно "0". Знаковый бит целого числа равен ”0".

Обработка SFC в CPU	Возвращаемое значение	Знак целого числа
Произошла ошибка	Меньше, чем ”0"	Отрицательный (знаковый бит равен ”1")
Ошибки нет	Больше, чем "0", или равно "0"	Положительный (знаковый бит равен ”0")

Реакция на информацию об ошибках

Если во время выполнения SFC происходит ошибка, то SFC предоставляет в возвращаемом значении (RET_VAL) код ошибки.

Различают:

- общий код ошибки, который могут выводить все SFC, и
- конкретный код ошибки, который SFC может выводить в зависимости от свой конкретной функции.

Передача значения функции

Некоторые SFC используют выходной параметр RET_VAL также для того, чтобы передать значение функции, например, SFC64 TIMETCK передает системное время, которое она считала, при помощи RET_VAL.

21.9.3 OB ошибок как реакция на обнаруженные ошибки

Обнаруживаемые ошибки

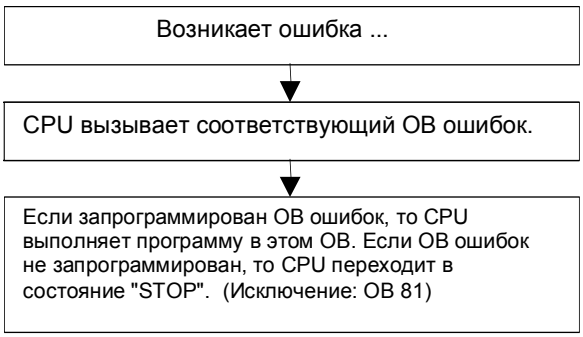
Системная программа может обнаруживать следующие ошибки:

- Неправильное функционирование CPU
- Ошибка в выполнении системной программы
- Ошибка в программе пользователя
- Ошибка при вводе/выводе

В зависимости от типа ошибки CPU устанавливается в режим STOP или вызывается OB ошибки.

Программирование реакций

Вы можете проектировать программы, чтобы реагировать на различные типы ошибок и задавать способ, которым реагирует CPU. Затем программу для конкретной ошибки можно сохранить в OB ошибок. Если вызывается этот OB ошибок, то выполняется данная программа.



OB ошибок

Различают синхронные и асинхронные ошибки следующим образом:

- Синхронные ошибки могут быть поставлены в соответствие команде MC7 (например, команде загрузки для сигнального модуля, который был снят).
- Асинхронные ошибки можно поставить в соответствие классу приоритета или всему программируемому логическому контроллеру (например, превышено время цикла).

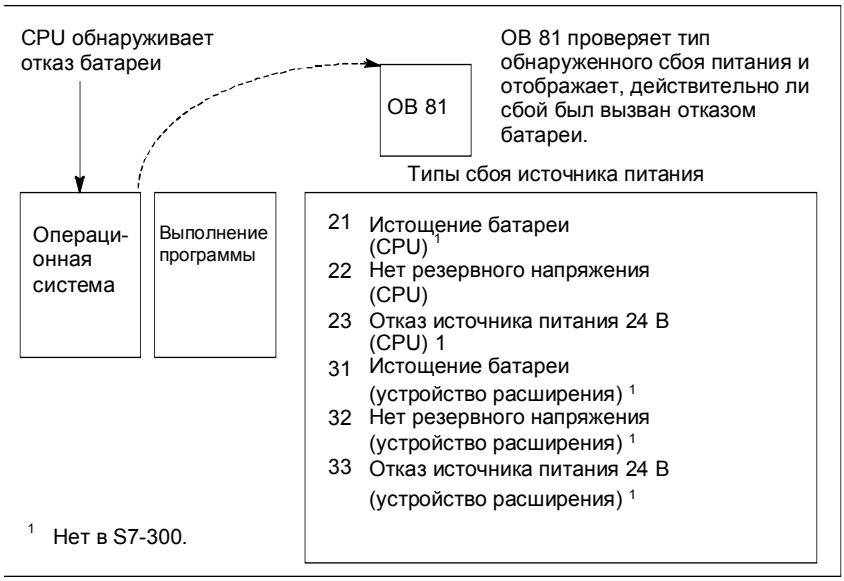
Следующая таблица показывает типы ошибок, которые могут происходить. За справкой о том, поддерживает ли ваш CPU указанные OB, обратитесь к руководству "Программируемый контроллер S7-300: Аппаратные средства и монтаж" или "Программируемые контроллеры S7-400, M7-400: Аппаратные средства и монтаж".

Класс ошибки	Тип ошибки	OB	Приоритет
Резервирование	Ошибка резервирования ввода/вывода (только в H CPU)	OB70	25
	Ошибка резервирования CPU (только в H CPU)	OB72	28
Асинхронная	Ошибка времени	OB80	26
	Сбой источника питания	OB81	(или 28, если OB ошибки вызывается в
	Диагностическое прерывание	OB82	программе запуска)
	Прерывание вставки/снятия модуля	OB83	
	Отказ аппаратных средств CPU	OB84	
	Ошибка последовательности выполнения программы	OB85	
	Отказ стойки	OB86	
	Ошибка связи	OB87	
Синхронная	Ошибка программирования	OB121	Приоритет OB, вызвавшего ошибку
	Ошибка доступа для ввода/вывода	OB122	

Пример использования организационного блока ошибки OB81

Используя локальные данные (стартовую информацию) в OB ошибки, вы можете оценить тип произошедшей ошибки.

Например, если CPU обнаруживает неисправность батареи, то операционная система вызывает OB81 (см. рисунок).



Вы можете написать программу, которая оценивает код события, запускаемый вызовом OB81. Вы можете также написать программу, которая осуществляет реакцию, такую как активизация выхода, связанного с лампой на станции оператора.

Локальные данные организационного блока ошибки OB81

Следующая таблица показывает временные переменные, которые должны быть описаны в этом случае в таблице описания переменных OB81.

Символ *Battery error* [сбой батареи] (BOOL) должен быть идентифицирован как выход (например, Q 4.0) так, чтобы другие части программы могли обращаться к этим данным.

Описание	Имя	Тип	Объяснение
TEMP	OB81EVCLASS	BYTE	Класс ошибки/Идентификатор ошибки 39xx
TEMP	OB81FLTID	BYTE	Код ошибки: b#16#21 = по крайней мере, одна батарея резервного питания CPU истощена [1)] b#16#22 = нет резервного напряжения в CPU b#16#23 = отказ источника питания 24 В в CPU [1)] b#16#31 = по крайней мере, одна батарея резервного питания стойки расширения истощена [1)] b#16#32 = нет резервного напряжения в стойке расширения [1)] b#16#33 = отказ источника питания 24 В стойки расширения [1)]
TEMP	OB81PRIORITY	BYTE	Класс приоритета = 26/28
TEMP	OB81OBNUMBR	BYTE	81 = OB81
TEMP	OB81RESERVED1	BYTE	Зарезервировано
TEMP	OB81RESERVED2	BYTE	Зарезервировано
TEMP	OB81MDLADDR	INT	Зарезервировано
TEMP	OB81RESERVED3	BYTE	Имеет смысл только для кодов ошибки B#16#31, B#16#32, B#16#33
TEMP	OB81RESERVED4	BYTE	
TEMP	OB81RESERVED5	BYTE	
TEMP	OB81RESERVED6	BYTE	
TEMP	OB81DATETIME	DATEANDTIME	Дата и время запуска OB

[1)] = нет в случае S7-300.

Типовая программа для организационного блока ошибки OB81

Типовая программа на языке AWL показывает, как вы можете считывать код ошибки в OB81.

Программа структурирована следующим образом:

- В OB81 (OB81FLTID) считывается код ошибки и сравнивается со значением для события ”battery exhausted [истощение батареи]" (B#16#3921).
- Если код ошибки соответствует коду события ”battery exhausted [истощение батареи]", то программа переходит к метке Berr и активизирует выход *batteryerror*.
- Если код ошибки не соответствует коду события ”battery exhausted [истощение батареи]", то программа сравнивает этот код с кодом события ”battery failure [отказ батареи]".
- Если код ошибки соответствует коду события ”battery failure [отказ батареи]", то программа переходит к метке Berr и активизирует выход *batteryerror*. В противном случае блок завершается.

AWL			Описание
	L	B#16#3921	Сравнить код события ”battery exhausted [истощение батареи]" (B#16#3921) с кодом ошибки для OB81. Если они одинаковы (истощение батареи), то перейти к Berr. Сравнить код события ”battery failure [отказ батареи]" (b#16#3922) с кодом ошибки для OB81. Если они разные (нет отказа батареи в центральной стойке), то закончить блок. Berr устанавливает выход ”batteryerror", если обнаружен отказ батареи или истощение батареи.
	L	#OB81FLTID	
	== I		
	JC	Berr	
	L	b#16#3922	
	<> I		
	BEC		
Berr:	S	#batteryerror	

Вы можете найти подробную информацию о блоках OB, SFB и SFC в соответствующей оперативной справке Help on Blocks [справка о блоках].

21.9.4 Подстановка заменяющих значений при обнаружении ошибок

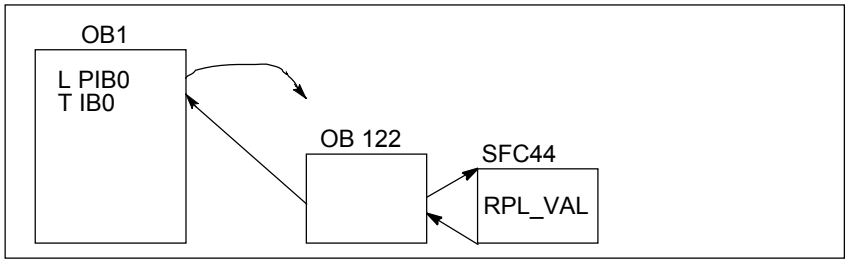
При ошибках некоторых типов (например, обрыв провода, воздействующий на входной сигнал), вы можете предоставлять заменяющие значения для значений, которые недоступны из-за ошибки. Есть два способа, с помощью которых вы можете предоставлять заменяющие значения:

- Вы можете назначать конфигурируемым модулям вывода заменяющие значения, используя STEP 7. Модули вывода, которые не могут иметь назначаемых параметров, имеют по умолчанию заменяющее значение 0.
- Вы можете программировать заменяющие значения в OB ошибок при помощи SFC44 RPLVAL (только для модулей ввода).

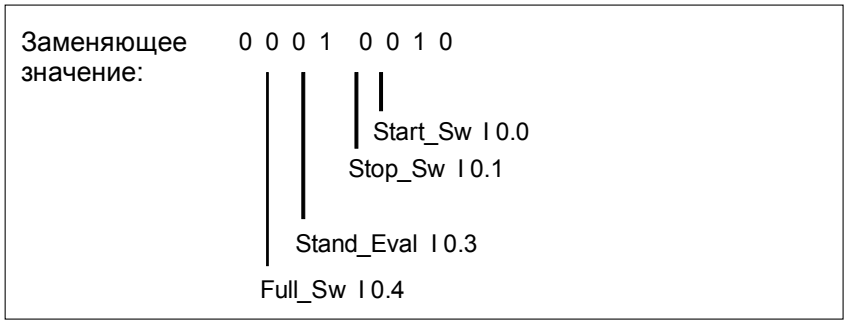
Для всех команд загрузки, приводящих к синхронным ошибкам, вы можете в OB ошибки задать заменяющее значение для содержимого аккумулятора.

Типовая программа для подстановки значения

В следующей типовой программе в SFC44 RPLVAL делается доступным заменяющее значение. Следующий рисунок показывает, как вызывается OB122, когда CPU обнаруживает, что модуль ввода не откликается.



В этом примере заменяющее значение, показанное на следующем рисунке, вводится в программу так, чтобы программа могла продолжать работать с допустимыми значениями.



Если модуль ввода выходит из строя, то обработка оператора L PIB0 вызывает синхронную ошибку и запускает OB122. Команда загрузки считывает обычно значение 0. Однако с помощью SFC44 вы можете определить любое заменяющее значение, подходящее для процесса. Эта SFC заменяет содержимое аккумулятора указанным заменяющим значением.

Следующую типовую программу можно было бы записать в OB122. Следующая таблица показывает временные переменные, которые должны быть описаны в этом случае в таблице описания переменных OB122.

Описание	Имя	Тип	Объяснение
TEMP	OB122EVCLASS	BYTE	Класс ошибки/идентификатор ошибки 29xx
TEMP	OB122SWFLT	BYTE	Код ошибки: 16#42, 16#43, 16#44 [1)], 16#45 [1)]
TEMP	OB122PRIORITY	BYTE	Класс приоритета = приоритет OB, в котором возникла ошибка
TEMP	OB122OBNUMBR	BYTE	122 = OB122
TEMP	OB122BLKTYPE	BYTE	Тип блока, в котором возникла ошибка

Описание	Имя	Тип	Объяснение
TEMP	OB122MEMAREA	BYTE	Область памяти и тип доступа
TEMP	OB122MEMADDR	WORD	Адрес памяти, в котором возникла ошибка
TEMP	OB122BLKNUM	WORD	Номер блока, в котором возникла ошибка
TEMP	OB122PRGADDR	WORD	Относительный адрес команды, вызвавшей ошибку
TEMP	OB122DATETIME	DATEANDTIME	Дата и время запуска OB
TEMP	Error	INT	Хранит код ошибки SFC44

[1] нет в случае S7-300.

AWL		Описание
	L B#16#2942 L #OB122SWFLT ==I JC Aerr L B#16#2943 <> I JC Stop	Сравнить код события OB122 с кодом события (B#16#2942) для подтверждения ошибки времени при чтении входов/выходов. Если они одинаковы, то переход к ”Aerr". Сравнить код события OB122 с кодом события (B#16#2943) для ошибки адресации (запись в несуществующий модуль). Если они неодинаковы, то переход к ”Stop".
Aerr:	CALL ”REPLVAL" VAL : = DW#16#2912 RETVAL : = #Error L #Error L 0 ==I BEC	Метка ”Aerr": передает DW#16#2912 (двоичный 10010) в SFC44 (REPLVAL). SFC44 загружает это значение в аккумулятор 1 (и заменяет значение, запущенное вызовом OB122). Код ошибки SFC сохраняется в #Error. Сравнить #Error с 0 (если совпадают, то во время выполнения OB122 ошибки не было). Если ошибки не было, то закончить блок.
Stop:	CALL ”STP"	Метка "Stop": вызывает SFC46 ”STP" и переключает CPU в состояние STOP.

21.9.5 Ошибка резервирования ввода/вывода (OB70)

Описание

Операционная система H CPU вызывает OB70, если на шине PROFIBUS DP происходит потеря резервирования (например, если имеется отказ шины на активном master-устройстве DP или ошибка в интерфейсном модуле slave-устройства DP) или если активный DP-master переключается со slave-устройств DP с переключаемыми входами/выходами.

Программирование OB70

Вы должны создать OB70 как объект вашей программы S7, используя STEP 7. Запишите программу, которая должна выполняться в OB70, в сгенерированный блок и загрузите его в CPU как часть вашей программы пользователя.

Вы можете использовать OB70, например, для следующих целей:

- Чтобы оценивать стартовую информацию OB70 и определять, какое событие вызвало потерю резервирования ввода/вывода.
- Чтобы определять состояние вашей системы, используя SFC51 RDSYSST (SZLID=B#16#71).

Если происходит ошибка резервирования ввода/вывода, а OB70 не запрограммирован, то CPU не переключается в режим STOP.

Если OB70 загружен и H-система не находится в резервном режиме, то OB70 обрабатывается в обоих CPU. H-система остается в резервном режиме.

Вы можете найти подробную информацию о блоках OB, SFB и SFC в соответствующей оперативной справке Help on Blocks [справка о блоках].

21.9.6 Ошибка резервирования CPU (OB72)

Описание

Операционная система H CPU вызывает OB72, если происходит одно из следующих событий:

- потеря резервирования в CPU
- ошибка сравнения (например, RAM, PIQ)
- переключение на резервное master-устройство
- ошибка синхронизации
- ошибка в субмодуле SYNC
- прерывание процесса обновления
- OB72 выполняется всеми CPU, находящимися в режиме RUN или STARTUP, после сопровождающего стартового события.

Программирование OB72

Вы должны создать OB72 как объект вашей программы S7, используя STEP 7. Запишите программу, которая должна выполняться в OB72, в сгенерированный блок и загрузите его в CPU как часть вашей программы пользователя.

Вы можете использовать OB72, например, для следующих целей:

- Чтобы оценивать стартовую информацию OB72 и определять, какое событие вызвало потерю резервирования CPU.
- Чтобы определять состояние вашей системы, используя SFC51 RDSYSST (SZLID=B#16#71).
- Чтобы реагировать на потерю резервирования CPU с учетом специфики установки.

Если происходит ошибка резервирования CPU, а OB72 не запрограммирован, то CPU не переключается в режим STOP.

Вы можете найти подробную информацию о блоках OB, SFB и SFC в соответствующей оперативной справке Help on Blocks [справка о блоках].

21.9.7 Ошибка времени (OB80)

Описание

Когда происходит ошибка времени, операционная система CPU вызывает OB80. Ошибки времени включают в себя, например, следующее:

- превышено максимальное время цикла
- пропуск прерываний по времени, вследствие сдвига времени вперед
- слишком большая задержка при обработке класса приоритета.

Программирование OB80

Вы должны создать OB80 как объект вашей программы S7, используя STEP 7. Запишите программу, которая должна выполняться в OB80, в сгенерированный блок и загрузите его в CPU как часть вашей программы пользователя.

Вы можете использовать OB80, например, для следующих целей:

- Чтобы оценивать стартовую информацию OB80 и определять, какие прерывания по времени были пропущены.
- Включая SFC29 CANTINT, вы можете деактивировать пропущенное прерывание по времени таким образом, что оно не выполнится, и будут выполняться только прерывания по времени, относящиеся к новому времени.

Если вы не деактивируете в OB80 пропущенные прерывания по времени, то первое пропущенное прерывание по времени выполнится, а все остальные игнорируются.

Если вы не запрограммируете OB80, то в случае обнаружении ошибки времени CPU переключается в режим STOP.

Вы можете найти подробную информацию о блоках OB, SFB и SFC в соответствующей оперативной справке Help on Blocks [справка о блоках].

21.9.8 Сбой источника питания (OB81)

Описание

Операционная система CPU вызывает OB81, если в CPU или стойке расширения происходит отказ одного из следующих элементов:

- источник питания 24 В
- батарея
- резервный комплект

Этот OB вызывается также тогда, когда проблема устраняется (этот OB вызывается, когда событие наступает и уходит).

Программирование OB81

Вы должны создать OB81 как объект вашей программы S7, используя STEP 7. Запишите программу, которая должна выполняться в OB81, в сгенерированный блок и загрузите его в CPU как часть вашей программы пользователя.

Вы можете использовать OB81, например, для следующих целей:

- Чтобы оценивать стартовую информацию OB81 и определять, какой сбой источника питания произошел.
- Чтобы найти номер стойки с неисправным источником питания.
- Для активизации на станции оператора лампы, указывающей на то, что обслуживающий персонал должен заменить батарею.

Если вы не запрограммируете OB81, то в случае обнаружении сбоя источника питания CPU не переключается в режим STOP, в отличие от всех других OB асинхронных ошибок. Однако ошибка вводится в диагностический буфер, и соответствующий светодиод на лицевой панели указывает на этот сбой.

Вы можете найти подробную информацию о блоках OB, SFB и SFC в соответствующей оперативной справке Help on Blocks [справка о блоках].

21.9.9 Диагностическое прерывание (OB82)

Описание

Операционная система CPU вызывает OB82, когда модуль с диагностическими возможностями, в котором вы разблокировали диагностическое прерывание, обнаруживает ошибку и когда ошибка устраняется (OB вызывается, когда событие наступает и уходит).

Программирование OB82

Вы должны создать OB82 как объект вашей программы S7, используя STEP 7. Запишите программу, которая должна выполняться в OB82, в сгенерированный блок и загрузите его в CPU как часть вашей программы пользователя.

Вы можете использовать OB82, например, для следующих целей:

- Чтобы оценивать стартовую информацию OB82.
- Чтобы получить точную диагностическую информацию о произошедшей ошибке.

Когда включается диагностическое прерывание, модуль, в котором возникла проблема, автоматически вводит 4 байта диагностических данных и их начальный адрес в стартовую информацию OB диагностического прерывания и в диагностический буфер. Это обеспечивает вас информацией о том, когда произошла ошибка и в каком модуле.

С помощью подходящей программы в OB82 вы можете оценивать дополнительные диагностические данные модуля (в каком канале произошла ошибка, какая ошибка произошла). Вы можете считывать диагностические данные модуля с помощью SFC51 RDSYSST и вводить эту информацию в диагностический буфер с помощью SFC52 WRUSRMSG. Вы можете также передавать определяемое пользователем диагностическое сообщение на контролирующее устройство.

Если вы не запрограммируете OB82, то в случае запуска диагностического прерывания CPU переключается в режим STOP.

Вы можете найти подробную информацию о блоках OB, SFB и SFC в соответствующей оперативной справке Help on Blocks [справка о блоках].

21.9.10 Прерывание вставки/снятия модуля (OB83)

Описание

CPU S7-400 контролируют присутствие модулей в центральной стойке и стойках расширения через интервалы длительностью примерно в 1 секунду.

После включения источника питания CPU проверяет, вставлены ли фактически все модули, перечисленные в конфигурационной таблице, созданной с помощью STEP 7. Если присутствуют все модули, то фактическая конфигурация сохраняется и используется как опорное значение для циклического контроля модулей. В каждом цикле сканирования вновь выявленная фактическая конфигурация сравнивается с предыдущей фактической конфигурацией. Если между этими конфигурациями имеются несоответствия, то передается сигнал прерывания вставки/снятия модуля и производится запись в диагностический буфер и в список состояний системы. В режиме RUN запускается OB прерывания вставки/снятия модуля.

Примечание

Модули источников питания, CPU и IM не должны сниматься в режиме RUN.

Между снятием и вставкой модуля нужно выдержать, по крайней мере, две секунды, чтобы CPU мог обнаружить, что был снят или вставлен модуль.

Назначение параметров вновь вставленному модулю

Если модуль вставляется в режиме RUN, то CPU проверяет, соответствует ли тип нового модуля типу первоначального модуля. Если типы согласуются, то модулю назначаются параметры. Модулю передаются либо заданные по умолчанию параметры, либо параметры, назначенные вами с помощью STEP 7.

Программирование OB83

Вы должны создать OB83 как объект вашей программы S7, используя STEP 7. Запишите программу, которая должна выполняться в OB83, в сгенерированный блок и загрузите его в CPU как часть вашей программы пользователя.

Вы можете использовать OB83, например, для следующих целей:

- Чтобы оценивать стартовую информацию OB83.
- Чтобы назначать параметры вновь вставленному модулю посредством включения системных функций с SFC55 по SFC59

Если вы не запрограммируете OB83, то при появлении прерывания вставки/снятия модуля CPU переключается из режима RUN в режим STOP.

Вы можете найти подробную информацию о блоках OB, SFB и SFC в соответствующей оперативной справке Help on Blocks [справка о блоках].

21.9.11 Отказ аппаратных средств CPU (OB84)

Описание

Операционная система CPU вызывает OB84, когда обнаруживается ошибка в интерфейсе с сетью MPI, коммуникационной шиной или сетевой платой для децентрализованной периферии; например, когда в линии обнаруживается неправильный уровень сигнала. Этот OB вызывается также тогда, когда ошибка устраняется (OB вызывается, когда событие наступает и уходит).

Программирование OB84

Вы должны создать OB84 как объект вашей программы S7, используя STEP 7. Запишите программу, которая должна выполняться в OB84, в сгенерированный блок и загрузите его в CPU как часть вашей программы пользователя.

Вы можете использовать OB84, например, для следующих целей:

- Чтобы оценивать стартовую информацию OB84.
- Чтобы передавать сообщение в диагностический буфер посредством включения системной функции SFC52 WRUSMSG.

Если вы не запрограммируете OB84, то CPU переключается в режим STOP в случае обнаружения отказа аппаратных средств CPU.

Вы можете найти подробную информацию о блоках OB, SFB и SFC в соответствующей оперативной справке Help on Blocks [справка о блоках].

21.9.12 Ошибка последовательности выполнения программы (OB85)

Описание

Операционная система CPU вызывает OB85:

- Когда для OB прерывания существует стартовое событие, но этот OB не может выполняться, потому что он не был загружен в CPU.
- Когда происходит ошибка доступа к экземплярному блоку данных системного функционального блока.
- Когда происходит ошибка обновления таблицы образа процесса (модуль не существует или неисправен).

Программирование OB85

Вы должны создать OB85 как объект вашей программы S7, используя STEP 7. Запишите программу, которая должна выполняться в OB85, в сгенерированный блок и загрузите его в CPU как часть вашей программы пользователя.

Вы можете использовать OB85, например, для следующих целей:

- Чтобы оценивать стартовую информацию OB85 и определять, какой модуль неисправен или не вставлен (определяется начальный адрес модуля).
- Чтобы выяснить слот соответствующего модуля посредством включения SFC49 LGCGADR.

Если вы не запрограммируете OB85, то в случае обнаружения ошибки класса приоритета CPU переключается в режим STOP.

21.9.13 Отказ стойки (OB86)

Описание

Операционная система CPU вызывает OB86, когда обнаруживается отказ стойки, например:

- Отказ стойки (отсутствие или неисправность IM, либо разрыв соединительного кабеля).
- Неисправность децентрализованного источника питания на стойке
- Отказ slave-устройства DP в master-системе шины SINEC L2-DP.

Этот OB вызывается также тогда, когда ошибка устраняется (OB вызывается, когда событие наступает и уходит).

Программирование OB86

Вы должны создать OB86 как объект вашей программы S7, используя STEP 7. Запишите программу, которая должна выполняться в OB86, в сгенерированный блок и загрузите его в CPU как часть вашей программы пользователя.

Вы можете использовать OB86, например, для следующих целей:

- Чтобы оценивать стартовую информацию OB86 и определять, какая стойка неисправна или отсутствует.
- Чтобы вводить сообщение в диагностический буфер при помощи системной функции SFC52 WRUSMSG и передавать сообщение в контролирующее устройство.

Если вы не запрограммируете OB86, то в случае обнаружения отказа стойки CPU переключается в режим STOP.

Вы можете найти подробную информацию о блоках OB, SFB и SFC в соответствующей оперативной справке Help on Blocks [справка о блоках].

21.9.14 Ошибка связи (OB87)

Описание

Операционная система CPU вызывает OB87, когда происходит ошибка связи при обмене данными с использованием коммуникационных функциональных блоков или при передаче глобальных данных, например:

- При приеме глобальных данных был обнаружен неправильный идентификатор кадра.
- Блок данных для информации о состоянии глобальных данных не существует или слишком короткий.

Программирование OB87

Вы должны создать OB87 как объект вашей программы S7, используя STEP 7. Запишите программу, которая должна выполняться в OB87, в сгенерированный блок и загрузите его в CPU как часть вашей программы пользователя.

Вы можете использовать OB87, например, для следующих целей:

- Чтобы оценивать стартовую информацию OB87.
- Чтобы создавать блок данных, если блок данных для информации о состоянии связи с помощью глобальных данных отсутствует.

Если вы не запрограммируете OB87, то в случае обнаружения ошибки связи CPU переключается в режим STOP.

Вы можете найти подробную информацию о блоках OB, SFB и SFC в соответствующей оперативной справке Help on Blocks [справка о блоках].

21.9.15 Ошибка программирования (OB121)

Описание

Операционная система CPU вызывает OB121, когда появляется ошибка программирования, например:

- Адресованные таймеры не существуют.
- Вызванный блок не загружен.

Программирование OB121

Вы должны создать OB121 как объект вашей программы S7, используя STEP 7. Запишите программу, которая должна выполняться в OB121, в сгенерированный блок и загрузите его в CPU как часть вашей программы пользователя.

Вы можете использовать OB121, например, для следующих целей:

- Чтобы оценивать стартовую информацию OB121.
- Чтобы вводить причину ошибки в блок данных сообщений.

Если вы не запрограммируете OB121, то в случае обнаружения ошибки программирования CPU переключается в режим STOP.

Вы можете найти подробную информацию о блоках OB, SFB и SFC в соответствующей оперативной справке Help on Blocks [справка о блоках].

21.9.16 Ошибка доступа для входов/выходов (OB122)

Описание

Операционная система CPU вызывает OB122, когда команда STEP 7 обращается к входу или выходу сигнального модуля, которому не был назначен модуль при последнем теплом рестарте, например:

- Ошибки прямого доступа к входам/выходам (модуль неисправен или отсутствует)
- Обращение к адресу входов/выходов, который не известен CPU.

Программирование OB122

Вы должны создать OB122 как объект вашей программы S7, используя STEP 7. Запишите программу, которая должна выполняться в OB122, в сгенерированный блок и загрузите его в CPU как часть вашей программы пользователя.

Вы можете использовать OB122, например, для следующих целей:

- Чтобы оценивать стартовую информацию OB122.
- Чтобы вызывать системную функцию SFC 44 и предоставлять заменяющее значение для модуля ввода так, чтобы выполнение программы могло продолжаться с использованием имеющего смысл значения, зависящего от процесса.

Если вы не запрограммируете OB122, то в случае обнаружения ошибки доступа к входам/выходам CPU переключается в режим STOP.

Вы можете найти подробную информацию о блоках OB, SFB и SFC в соответствующей оперативной справке Help on Blocks [справка о блоках].

22 Печать и архивирование

22.1 Печать проектной документации

22.1.1 Печать проектной документации

Когда вы завершите создание программы для вашей задачи автоматизации, вы можете в целях документирования проекта распечатать все важные данные при помощи функций печати, встроенных в STEP 7.

Части проекта, которые вы можете печатать

Вы можете печатать содержимое объектов как непосредственно из SIMATIC Manager, так и посредством открытия соответствующего объекта и запуска процедуры печати.

Непосредственно через SIMATIC Manager можно напечатать следующие части проекта:

- Дерево объектов (структура проекта/библиотеки)
- Списки объектов (содержимое папки объектов)
- Содержимое объектов
- Сообщения

Открывая соответствующий объект, можно напечатать следующие части проекта:

- Блоки, представленные в форме контактного плана, списка операторов или функционального плана или на других языках (дополнительное программное обеспечение).
- Таблица символов с символическими именами для абсолютных адресов.
- Конфигурационная таблица со схемой расположения модулей в программируемом контроллере и параметрами модулей.
- Содержимое диагностического буфера.
- Таблица переменных с форматами наблюдения, наблюдаемыми и изменяемыми значениями.
- Справочные данные, такие как списки перекрестных ссылок, списки назначений, структуры программ, списки неиспользованных адресов, списки адресов без символов.
- Таблица глобальных данных.
- Информация о модулях с состоянием модулей.
- Списки текстов пользователя.
- Документы из дополнительных пакетов, таких как другие языки программирования.

Дополнительный пакет DOCPRO

Для создания, редактирования и печати стандартизированных руководств по монтажу вы можете использовать дополнительный пакет программ DOCPRO. Он создает документацию установки, удовлетворяющую стандартам ANSI и DIN.

22.1.2 Основная последовательность действий при печати

Чтобы печатать, действуйте следующим образом:

1. Откройте соответствующий объект, чтобы отобразить на экране информацию, которую вы хотите печатать.
2. Откройте диалоговое окно ”Print [Печатать]", используя команду меню **File > Print [Файл > Печатать]** в окне приложения. В зависимости от того, в каком приложении вы находитесь, первым входом в строке меню может быть не ”File [Файл]", а объект, обрабатываемый приложением, такой как ”Symbol Table [Таблица символов]".
3. В случае необходимости измените в диалоговом окне опции печати (принтер, диапазон печати, число копий и т.д.) и закройте его.

Некоторые диалоговые окна имеют кнопку "Print [Печатать]", например, диалоговое окно "Module Information [Информация о модуле]". Щелкните на этой кнопке, чтобы распечатать содержимое диалогового окна.

Блоки не требуют открытия. Вы можете печатать их непосредственно из SIMATIC Manager, используя команду меню **File > Print [Файл > Печатать]**.

22.1.3 Функции печати

Для печати объектов имеются следующие дополнительные функции:

Объекты печати	Команда меню	Функция	Функция	Функция	Функция
		Предварительный просмотр перед печатью	Настройка страниц	Верхние и нижние колонтитулы	Настройка печати
Блоки, исходные файлы на AWL	File [Файл] > *	•	•	–	•
Информация о модулях		–	•	–	–
Таблица глобальных данных	GD Table [Таблица ГД] > *	•	•	–	•
Конфигурационная таблица	Station [Станция] > *	•	•	–	•
Объект, папка объектов	File [Файл] > *	–	•	•	•
Справочные данные	Reference Data [Справочные данные] > *	•	•	–	•
Таблица символов	Symbol Table [Таблица символов] > *	•	•	–	•
Таблица переменных	Table [Таблица] > *	–	•	–	•
Таблица соединений	Network [Сеть] > *	•	•	–	•
Список текстов пользователя	Texts [Тексты] > *	•	•	–	•
* : Символ * служит как подстановочный знак для соответствующей функции в команде меню (например, предварительный просмотр перед печатью или настройка страниц)					

Команды пошаговой печати отдельных объектов печати можно найти в: How to Print [Как печатать].

Предварительный просмотр перед печатью

Вы можете использовать функцию "Print Preview [Предварительный просмотр перед печатью]" для вывода на экран расположения страниц печатаемого документа.

Если документ состоит из нескольких страниц, то в правом нижнем углу страницы после номера страницы появляются две точки. Последняя страница не имеет этих точек, что указывает на отсутствие последующих страниц.

Примечание

Формат печати законченного документа в предварительном просмотре перед печатью не отображается.

Установка формата страницы

С помощью функции "Page setup [Настройка страниц]", вы можете устанавливать формат страницы для документа, который хотите печатать (например, A4, A5, Letter [Письмо]).

Настраивайте расположение документа так, чтобы он соответствовал требуемому формату бумаги. Если документ слишком широкий, то правое поле будет печататься на следующей странице.

Если вы выбираете формат страницы с полем (например, A4 Margin), то напечатанный документ имеет поле на левой стороне страницы, которое вы можете использовать, чтобы перфорировать отверстия для брошюровки.

Примечание

Если вы нуждаетесь в справке о диалоговом окне "Page Setup [Настройка страниц]", то щелкните на кнопке "Help [Помощь]" или нажмите клавишу F1, когда курсор находится в диалоговом окне.

Установка верхних и нижних колонтитулов

С помощью функции "File > Headers and Footers [Файл > Верхние и нижние колонтитулы]" в SIMATIC Manager вы можете устанавливать верхние и нижние колонтитулы для документов, которые хотите печатать, на протяжении всего проекта. В отдельных приложениях вы можете задавать только формат страницы. Если документ состоит из нескольких страниц, то в правом нижнем углу страницы после номера страницы появляются две точки. Последняя страница не имеет этих точек, что указывает на отсутствие последующих страниц. Это быстрый способ проверки того, закончена ли печать. Эти две точки видны также в предварительном просмотре перед печатью.

Настройка печати

С помощью функции "Print Setup [Настройка печати]" вы можете выбирать принтер и устанавливать формат бумаги (книжный или альбомный). Параметры настройки, доступные для этой функции, зависят от типа используемого драйвера печати.

22.1.4 Специальное примечание к печати дерева объектов

Из диалогового окна "Print Object List [Печать списка объектов]", кроме списка объектов, вы можете печатать также дерево объектов, выбирая опцию "Tree window [Окно дерева]".

Если вы из-под "Print range [Диапазон печати]" выбираете опцию "All [Все]", то печатается вся структура дерева. Если вы выбираете кнопку "Selection [Выбор]", то печатается структура дерева, идущая вниз от выбранного объекта.

Примечание

Установки, сделанные в диалоговом окне, применяются только к печати списка или дерева, но не к печати содержимого объектов; для этого используются параметры настройки в соответствующих приложениях.

22.2 Архивирование проектов и библиотек

22.2.1 Архивирование проектов и библиотек

Вы можете хранить отдельные проекты или библиотеки в сжатой форме в архивном файле. Такое сжатое хранение возможно на жестком диске или на переносном носителе данных (например, на гибком диске).

Архивирование программ

Функция архивирования обеспечивает вас интерфейсом для вызова программы архивирования по вашему выбору. Программы архивирования ARJ и PKZIP 2.50 включены в пакет STEP 7 как его составная часть. Если вы используете одну из перечисленных ниже программ архивирования, вам потребуются следующие версии (или более новая версия):

- ARJ с версии 2.4.1a
- PKZIP с версии 2.04g
- LHARC с версии 2.13
- WinZip с версии 6.0
- JAR с версии 1.02

Рекомендация по архивированию

Проекты, которые имеют длинные имена файлов (длиннее, чем в соглашении 8.3 DOS) или содержат структуры каталогов глубокой вложенности (каталоги с длиной абсолютного имени пути более 64 символов), нужно архивировать только с помощью программ архивирования PKZIP 2.50, WinZip или JAR. В случае других программ архивирования нет никакой гарантии того, что эти структуры будут поддерживаться и что архивные файлы будут полностью и правильно распаковываться. Это особенно относится к проектам, которые содержат объекты из дополнительного пакета WinCC.

22.2.2 Использование для сохранения/архивирования

Save As [сохранить как …]

С помощью этой функции вы создаете **копию** проекта под другим именем.

Вы можете использовать эту функцию:

- для создания резервных копий
- для дублирования существующего проекта, чтобы адаптировать его к другим целям.

Чтобы использовать самый быстрый метод создания копии, выбирайте в диалоговом окне опцию "Save As" [сохранить как...] без переупорядочивания. Вся файловая структура, идущая вниз от каталога проекта, копируется без проверки и сохраняется под другим именем.

На носителе данных должно быть достаточно места для хранения резервной копии. Не пытайтесь сохранять проекты на дискете, так как там в общем случае не будет достаточного доступного пространства. Для переноса проектных данных на дискету используйте функцию "Archive [Архив]".

Сохранение с переупорядочиванием протекает дольше, но при этом выводится сообщение, если объект не может быть скопирован и сохранен. Причинами этого могут быть отсутствие дополнительного пакета или поврежденные данные для объекта.

Архив

Вы можете хранить отдельные проекты или библиотеки в сжатой форме в архивном файле. Такое сжатое хранение возможно на жестком диске или на переносном носителе данных (например, на гибком диске).

Переносите проекты на дискету только в форме архивного файла. Если проект слишком большой, то выберите программу архивирования, с помощью которой можно создавать многотомные архивы.

Проекты или библиотеки, сжатые в архивный файл, невозможно редактировать. Если вы хотите редактировать их повторно, то вам нужно распаковать эти данные, что означает извлечение проекта или библиотеки.

22.2.3 Предпосылки для архивирования

Перед архивированием проекта или библиотеки должны быть выполнены следующие требования:

- В вашей системе должна иметься установленная программа архивирования. Связь с STEP 7 объясняется в разделе оперативной справки "Steps for Archiving/Retrieving [Этапы архивирования/извлечения]".
- Все данные проекта без исключения должны быть в каталоге или подкаталоге проекта. При работе со средой разработкой на языке C можно хранить данные в других местах. Такие данные не будут в этом случае включены в файл архива
- Имена файлов должны удовлетворять соглашениям об именах в DOS (восемь символов для имени плюс три символа для расширения), если вы работаете с одной из программ архивирования ARJ, PKZip версии 2.04g или LHArc, потому что эти программы архивирования являются программами DOS. PKZip версии 2.50, Jar и WinZip не обязаны удовлетворять этим соглашениям об именах.

22.2.4 Процедура архивирования/извлечения

Вы архивируете/извлекаете ваш проект или библиотеку, используя команду меню **File > Archive [Файл > Архивировать]** или **File > Retrieve [Файл > Извлечь]**.

Примечание

Проекты или библиотеки, сжатые в архивный файл, невозможно редактировать. Если вы хотите редактировать их снова, то вам нужно распаковать эти данные, что означает извлечение проекта или библиотеки.

Извлекаемые проекты или библиотеки при извлечении автоматически включаются в список проектов/библиотек.

Установка целевого каталога

Чтобы установить целевой каталог, используйте команду меню **Options > Customize [Параметры > Настройка]** в SIMATIC Manager для открытия диалогового окна "Customize [Настройка]".

Во вкладке "Archive [Архив]" этого диалогового окна вы можете включать и выключать опцию "Check target directory on retrieval [Проверить целевой каталог при извлечении]".

Если эта опция деактивирована, то путь, установленный во вкладке "SIMATIC Manager" того же самого диалогового окна, используется как целевой каталог для извлечения.

Копирование архивного файла на дискету

Вы можете заархивировать проект/библиотеку и затем скопировать архивный файл на дискету. Возможно также в диалоговом окне ”Archive [Архив]" выбрать дисковод гибкого диска в качестве целевого каталога.

Примечание к извлечению посредством PKZIP 2.04g

Если в то время, когда вы создавали архив на дискете при помощи программы архивирования PKZIP, была активизирована опция "Disk-crossing archive [Многотомный архив]", то при извлечении вами архива программа попросит вас вставить последнюю дискету архива. PKUNZIP всегда выводит в окно DOS следующее сообщение:

```
Insert the LAST disk of the backup set - Press a key when
ready. [Вставьте ПОСЛЕДНИЙ диск архивного набора - Нажмите
клавишу, когда будет готово.]
```

Это сообщение появляется даже тогда, когда архив создавался с опцией "Disk-crossing archive [Многотомный архив]", но весь архив помещается на одной единственной дискете.

В этом случае игнорируйте сообщение и подтвердите диалог нажатием любой клавиши.

23 Редактирование одного проекта несколькими пользователями

23.1 Многопользовательская конфигурация в сети

Краткий обзор

Вы можете работать с пакетом STEP 7 в сетях Windows 95/98/NT Workgroups и NT/Novell в многопользовательской конфигурации. Возможны три разных метода:

- Проект находится на локальном диске и используется также из другой рабочей станции

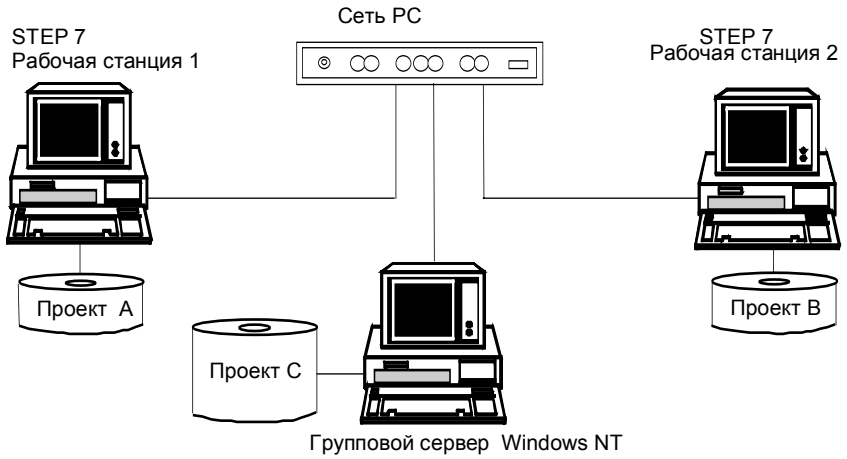
  Пример: Рабочие станции 1 и 2 обращаются к проекту A на рабочей станции 1.

- Проект находится на сервере проекта/сети.

  Пример: Рабочие станции 1 и 2 обращаются к проекту C на сервере сети.

- Проекты распределены среди локальных дисков и одного или большего количества серверов проекта/сети.

  Пример: Рабочие станции 1 и 2 обращаются к проектам A, B и C.



Рекомендации по хранению проектов на сетевых серверах

- Когда вы храните ваши проекты на сетевых серверах, пути доступа всегда должен быть назначен символ дисковода.
- Когда вы храните ваши проекты на сетевых серверах или на разрешенных дисках других пользователей сети, система Windows 95/98/NT в этих серверах или сетевых узлах может выключаться только тогда, когда закрыты все приложения STEP 7, которые обращаются к этим проектам.

Рекомендации по многопользовательскому редактированию программ S7

Вам следует обратить внимание следующее:

- Прежде чем несколько пользователей смогут работать с одной программой S7, вы должны задать конфигурацию рабочих станций (команда меню **Start > Simatic > STEP 7 > Configure SIMATIC Workspace [Пуск > Simatic > STEP 7 > Конфигурирование рабочего пространства SIMATIC]**).
- Блоки и исходные файлы на AWL:
  Каждый пользователь должен программировать отдельный блок или исходный файл. Если два пользователя пытаются в одно и то же время редактировать один и тот же блок или исходный файл, то на экран выводится сообщение и второму пользователю дается отказ в доступе.
- Таблица символов:
  Несколько пользователей могут открывать таблицу символов в одно и то же время, но только один пользователь может редактировать ее. Если два пользователя пытаются редактировать таблицу символов одновременно, то на экране отображается сообщение, а доступ для второго пользователя отклоняется.
- Таблицы переменных:
  Несколько пользователей могут открывать таблицу переменных в одно и то же время, но только один пользователь может редактировать ее. Если два пользователя пытаются редактировать таблицу переменных одновременно, то на экран выводится сообщение и второму пользователю дается отказ в доступе. В программе S7 может быть много таблиц переменных. Конечно, их можно редактировать отдельно и независимо друг от друга.

Рекомендации по многопользовательскому редактированию станций

Вам следует обратить внимание следующее:

- Аппаратная конфигурация и сетевая конфигурация станции должны редактироваться централизованно только одним пользователем.

24 Работа с программируемыми системами управления M7

24.1 Процедура для систем M7

Стандартная архитектура PC компьютеров для решения задач автоматизации M7-300/M7-400 образует свободно программируемое расширение для платформы автоматизации SIMATIC. Вы можете создавать программы пользователя для SIMATIC M7 на языке высокого уровня типа C или графически, используя CFC (Continuous Function Chart – Схему непрерывных функций).

Для создания программ вам, кроме STEP 7, потребуются также системное программное обеспечение M7-SYS RT для M7-300/400 и среда разработки программ M7 (ProC/C++ или CFC).

Основная последовательность действий

Когда вы разрабатываете решение по автоматизации с помощью SIMATIC M7, существует ряд основных задач. Следующая таблица показывает задачи, которые требуется выполнить для большинства проектов, и ставит их в соответствие основной процедуре. Эта таблица дает также ссылки на соответствующие главы данного руководства или других руководств.

Процедура	Описание
7. Проектирование решения задачи автоматизации	Специфична для M7. Обратитесь к Руководству по программированию для M7-SYS RT
8. Запуск STEP 7	Как для S7
9. Разработка структуры проекта 10. Установка станции 11. Конфигурирование аппаратных средств	Как для S7
12. Конфигурирование коммуникационных соединений	Как для S7
13. Определение таблицы символов	Как для S7
14. Разработка программы пользователя на языке C или CFC	Специфична для M7. Обратитесь к ProC/C++
15. Конфигурирование операционной системы 16. Установка операционной системы на M7-300/M7-400 17. Загрузка конфигурации аппаратных средств и программы пользователя в M7	Специфично для M7 Обратитесь к Руководству пользователя M7-SYS RT
18. Тестирование и отладка программы пользователя	ProC/C++
19. Контроль работы и диагностика M7	Как для S7, но без определяемой пользователем диагностики
20. Печать и архивирование	Как для S7

Чем отличается M7?

Для M7-300/M7-400 не поддерживаются следующие функции STEP 7:

- Многопроцессорная обработка – синхронная работа нескольких CPU
- Принудительно устанавливаемые переменные
- Связь с помощь глобальных данных
- Определяемая пользователем диагностика

Управление программируемыми контроллерами M7

STEP 7 предоставляет вам определенную поддержку по следующим задачам на программируемых контроллерах M7:

- Установка операционной системы на M7-300/M7-400
- Конфигурирование операционной системы посредством редактирования системных файлов
- Загрузка программ пользователя в M7-300/M7-400
- Обновление программ, записанных в ПЗУ.

Для обращения к программируемой системе управления M7 выберите из контекста проекта, содержащего станции с CPU или FM M7 с выбранной папкой программ M7, следующую команду меню:

PLC > Manage M7 System [ПЛК > Управление системой M7]

Вы найдете подробно описанные команды в оперативной справке и руководстве пользователя для M7-SYS RT.

24.2 Дополнительное программное обеспечение для программирования M7

Дополнительное программное обеспечение M7

STEP 7 предоставляет основные функции, которые потребуются вам для выполнения следующих действий:

- Разработка и управление проектами
- Конфигурирование и назначение параметров аппаратным средствам
- Конфигурирование сетей и соединений
- Управление символьными данными

Эти функции предоставляются независимо от того, используете ли вы программируемый контроллер SIMATIC S7 или SIMATIC M7.

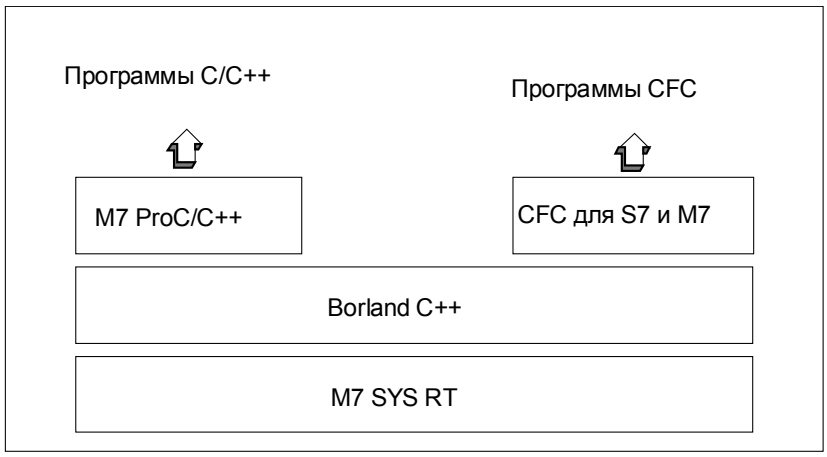
Для создания приложений M7 вам, кроме STEP 7, потребуется дополнительное программное обеспечение M7.

Программное обеспечение	Содержание
M7-SYS RT	Операционная система M7 RMOS32 Системная библиотека M7-API Поддержка для MPI
CFC для S7 и M7	Программное обеспечение для программ CFC (Continuous Function Chart [Схема непрерывных функций])
M7-ProC/C++	Связь для среды разработки Borland в STEP 7 Редактор импорта и генератор символов Средство отладки языка высокого уровня Organon xdb386
Borland C++	Среда разработки Borland C/C++

STEP 7 в сочетании с дополнительным программным обеспечением M7 может поддерживать также следующие дополнительные задачи:

- Загрузка данных в программируемый контроллер M7 через многоточечный интерфейс (MPI)
- Запрос информации о программируемой системе управления M7
- Выполнение специальной настройки параметров в программируемой системе управления M7 и сброс M7.

Следующий рисунок показывает подчиненность внутри дополнительного программного обеспечения M7 для программирования M7.



Резюме

Чтобы создавать ...	**Вам требуется дополнитнльное программное обеспечение M7...**
программы C/C++	1. M7-SYS RT 2. M7-ProC/C++ 3. Borland C++
программы CFC	1. M7-SYS RT 2. CFC для S7 и M7 3. Borland C++

Какое программное обеспечение и какой тип поддержки предоставляет?

Определенные инструментальные средства, требуемые для создания приложений M7, встроены частично в STEP 7 и частично в дополнительные программные средства M7.

Следующая таблица показывает, какой пакет программ какие задачи поддерживает:

Программное обеспечение	Предоставляемая поддержка
STEP 7	Установка операционной системы M7 Ведение программируемой системы управления M7 Загрузка, запуск и удаление программ M7 Отображение состояния и диагностических данных Сброс CPU
M7-SYS RT	Операционная система M7 и утилиты системного программного обеспечения M7 оказывают содействие при: управлении обработкой программы управлении памятью и ресурсами доступе к аппаратуре компьютера и аппаратуре SIMATIC обработке прерываний диагностике контроле состояния коммуникациях
M7-ProC/C++	Посредством создания встроенного кода (интегрирование среды разработки Borland в STEP 7) Посредством связывания символов проекта с исходным кодом Посредством встроенных функций отладки
Borland C++	Создание программ C и C++
CFC для S7 и M7	Создание, тестирование и отладка программ CFC Запуск и выполнение программ CFC

24.3 Операционные системы M7-300/M7-400

Предоставляемые операционной системой утилиты очень важны для приложений, созданных с помощью языков высокого уровня C и C++. Операционная система принимает на себя следующие задачи для приложения:

- обращение к аппаратным средствам
- управление ресурсами
- системная интеграция
- связь с другими компонентами системы.

Для решения задач автоматизации компьютером SIMATIC M7 используется многозадачная операционная система реального времени M7 RMOS32 (**R**ealtime **M**ultitasking **O**perating **S**ystem). M7 RMOS32 была расширена так, чтобы включить в себя интерфейс вызовов, M7 API (**A**pplication **P**rogramming **I**nterface [прикладной программный интерфейс]), для интегрирования его в систему SIMATIC.

Операционная система реального времени M7 RMOS32 используется для 32-битных приложений в решениях, критических по времени, в реальном времени и в многозадачных решениях. Она доступна в следующих конфигурациях модулей M7:

- M7 RMOS32
- M7 RMOS32 с MS-DOS

Конфигурация операционной системы, выбираемая вами для вашей программируемой системы управления M7, зависит от используемых вами модулей:

Конфигурация операционной системы	Модуль / Основная память	PROFIBUS-DP и TCP/IP да/нет	Установка на памяти большого объема
M7 RMOS32	FM 356-4 / 4 Мбайт	Нет	Плата памяти ≥4 Мбайт
	FM 356-4 / 8 Мбайт	Да	или жесткий диск
	CPU 388-4 / 8 Мбайт	Да	
	FM 456-4 / 16 Мбайт	Да	
	CPU 488-3 / 16 Мбайт	Да	
	CPU 486-3 / 16 Мбайт	Да	
M7 RMOS32 с MS-DOS	FM 356-4 / 8 Мбайт	Нет	Плата памяти ≥4 Мбайт
	CPU 388-4 / 8 Мбайт	Нет	или жесткий диск
	FM 456-4 / 16 Мбайт	Да	
	CPU 488-3 / 16 Мбайт	Да	
	CPU 486-3 / 16 Мбайт	Да	

25 Секреты и советы

25.1 Переупорядочивание

Если при работе с STEP 7 возникают необъяснимые проблемы, то это часто способствует тому, чтобы переупорядочить базу данных проекта или библиотеки.

Для выполнения этого выберите команду меню **File > Rearrange [Файл > Переупорядочить]**. Она удаляет любые пробелы в памяти, появляющиеся при стирании содержимого; это означает, что уменьшается объем памяти, требуемый для данных проекта/библиотеки.

Эта функция оптимизирует хранение данных проекта или библиотеки способом, подобным тому, как программа дефрагментации жесткого диска оптимизирует хранение файлов на жестком диске.

Продолжительность процесса переупорядочения зависит от количества перемещаемых данных и может занимать некоторое время. Поэтому функция не выполняется автоматически (например, когда вы закрываете проект), а должна запускаться пользователем тогда, когда он/она желает переупорядочить проект или библиотеку.

Предпосылка

Проекты и библиотеки могут переупорядочиваться только тогда, когда никакие объекты в них не редактируются другими приложениями и, следовательно, не блокированы для доступа.

25.2 Виртуальная рабочая память

Другой причиной проблем, возникающих в STEP 7, может быть слишком малая виртуальная рабочая память.

Чтобы работать с пакетом STEP 7 под Windows 95, вам нужно скорректировать параметры настройки виртуальной памяти. Коррекцию параметров настройки выполняйте следующим образом:

1. Откройте панель управления с помощью команды **Start > Settings > Control Panel [Пуск > Настройка > Панель управления]**.
2. Дважды щелкните на пиктограмме "System [Система]".
3. В диалоговом окне "System Properties [Свойства системы]" выберите вкладку "Performance [Быстродействие]".
4. Щелкните на кнопке "Virtual Memory [Виртуальная память]".
5. В диалоговом окне "Virtual Memory [Виртуальная память]" выберите опцию "Let me specify my own virtual memory settings [Параметры виртуальной памяти устанавливаются вручную]".
6. Введите, по крайней мере, 40 Мбайт как "Minimum" и, по крайней мере, 150 Мбайт как "Maximum".
7. Убедитесь, что опция "Disable virtual memory [Блокировать виртуальную память]" деактивирована.

Примечание

Поскольку виртуальная память находится на жестком диске (по умолчанию C:) и является динамической, вы должны гарантировать, что для каталога TMP или TEMP доступна достаточная память (примерно от 20 до 30 Мбайт):

Если проект S7 тоже находится в том же самом разделе, в котором установлена виртуальная память, то в качестве свободного пространства памяти должен быть доступен объем, примерно равный удвоенному объему проекта S7.

Если проект хранится в другом разделе, то это требование становится несущественным.

---

26 Как создавать и редактировать проекты

26.1 Как создавать проекты

26.1.1 Создание проекта с использованием мастера

Для создания проекта с использованием мастера действуйте следующим образом:

1. Выберите в SIMATIC Manager команду меню File > "New Project" Wizard [Файл > Мастер нового проекта].
2. Введите подробности, запрашиваемые мастером в диалоговых окнах.

26.1.2 Создание проекта вручную

Для создания проекта вручную действуйте следующим образом:

1. Выберите в SIMATIC Manager команду меню **File > New [Файл > Новый]**.
2. В диалоговом окне "New [Новый]" выберите опцию "New Project [Новый проект]".
3. Введите имя проекта и подтвердите ваш ввод посредством ”OK".

26.1.3 Вставка станции

Чтобы создать в проекте новую станцию, откройте проект так, чтобы отображалось окно проекта.

1. Выделите проект.
2. Создайте для требуемых аппаратных средств объект "Station [Станция]", используя команду меню **Insert > Station [Вставить > Станция]**".

Если станция не отображается, то щелкните по знаку ”+" перед значком проекта в окне проекта.

26.2 Как редактировать проекты

26.2.1 Копирование проекта

Для копирования проекта действуйте следующим образом:

1. Выделите проект, который вы хотите копировать.
2. Выберите в SIMATIC Manager команду меню **File > Save As [Файл > Сохранить как...]**.
3. Определите в диалоговом окне "Save As [Сохранить как...]", хотите ли вы выполнить переупорядочивание перед сохранением или нет. В случае старых проектов или проектов, в которых вы сделали много изменений, вам нужно выбрать опцию "Rearrange before saving [Переупорядочить перед сохранением]", чтобы оптимизировать хранение данных и проверить структуру проекта.
4. В диалоговом окне "Save Project As [Сохранить проект как...]" введите имя нового проекта и, если необходимо, новый путь к месту хранения. Подтвердите посредством "OK".

26.2.2 Копирование части проекта

Если вы хотите скопировать часть проекта, такую как станции, программные средства, блоки, и т.д., то действуйте следующим образом:

1. Выделите часть проекта, которую вы хотите копировать.
2. Выберите в SIMATIC Manager команду меню **Edit > Copy [Редактировать > Копировать]**.
3. Выделите папку, в которой нужно сохранить скопированную часть проекта.
4. Выберите команду меню **Edit > Paste [Редактировать > Вставить]**.

26.2.3 Удаление проекта

Для удаления проекта действуйте следующим образом:

1. Выберите в SIMATIC Manager команду меню **File > Delete [Файл > Удалить]**.
2. В диалоговом окне "Delete [Удалить]" активизируйте кнопку выбора "Project [Проект]". Проекты перечисляются ниже в списковом окне.
3. Выберите часть проекта, которую вы хотите удалить, и подтвердите посредством "OK".
4. Подтвердите подсказку посредством "Yes [Да]".

26.2.4 Удаление части проекта

Для удаления части проекта действуйте следующим образом:

1. Выделите часть проекта, которую вы хотите удалить.
2. Выберите в SIMATIC Manager команду меню **Edit > Delete [Редактировать > Удалить]**.
3. Подтвердите подсказку посредством "Yes [Да]".

26.2.5 Конфигурирование аппаратных средств

Для конфигурирования аппаратных средств действуйте следующим образом:

1. Щелкните на новой станции. Она содержит объект ”Hardware [Аппаратные средства]".
2. Откройте объект ”Hardware [Аппаратные средства]". Отображается окно ”Hardware Configuration [Конфигурирование аппаратных средств]".
3. В окне ”Hardware Configuration [Конфигурирование аппаратных средств]" спланируйте структуру станции. Для оказания помощи в вашем распоряжении имеется каталог модулей. Каталог открывается командой меню **View > Catalog [Вид > Каталог]**.
4. Сначала в пустое окно вставьте стойку (rack) из каталога модулей. Затем выберите модули и разместите их в слотах стойки. Для станции нужно сконфигурировать, по крайней мере, один CPU.

Если эти объекты еще не видны в окне проекта, то щелкните на знаке ”+" перед значком станции в окне проекта, чтобы отобразить модуль, и щелкните на поле перед модулем, чтобы отобразить программу S7/M7 и объект ”Connections [Соединения]".

26.2.6 Создание программного обеспечения в проекте (общий случай)

Чтобы создать программное обеспечение для вашего проекта действуйте следующим образом:

1. Откройте программу S7 или программу M7.
2. Откройте объект ”Symbols [Символы]" в программе S7 или M7 и назначьте символы. (Этот шаг можно сделать также и позже.)
3. Откройте папку ”Blocks [Блоки]", если вы хотите создавать блоки, или папку ”Source Files [Исходные файлы]", если вы хотите создать исходный файл.
4. Вставьте блок или исходный файл. Команды меню для этого:
   - Insert > S7 Block [Вставить > Блок S7]
   - Insert > S7 Software [Вставить > Программное средство S7]
   - Insert > M7 Software [Вставить > Программное средство M7]
5. Откройте блок или исходный файл и введите программу. Вы найдете подробную информацию о программах в руководствах по языкам программирования.

6. Документируйте проект, используя команду меню **Insert > Project Documentation [Вставить > Проектная документация]**.
   Чтобы документировать проект STEP 7, вы можете все данные конфигурации, созданные с помощью STEP 7, организовать в виде руководств по монтажу. Эта функция доступна только тогда, когда установлен дополнительный пакет ”DOCPRO".

В зависимости от вашей задачи вам, возможно, будет нужно выполнять не все эти шаги.

26.3 Как управлять объектами

26.3.1 Копирование объектов

Непосредственное копирование при помощи мыши (Drag & Drop [переместить и отпустить])

1. Обеспечьте, чтобы отображались как объект, который вы хотите скопировать, так и требуемая целевая папка (в случае необходимости откройте дополнительное окно проекта).
2. Выделите объект, который вы хотите скопировать, и держите нажатой левую кнопку мыши.
3. Переместите указатель мыши на целевую папку. При этом удерживайте нажатой левую кнопку мыши. Если вы пытаетесь скопировать объект на запрещенное место, то вместо курсора появляется знак запрета.
4. Отпустите левую кнопку мыши снова.

Копирование посредством команд меню

1. Выделите объект, который вы хотите скопировать.
2. Выберите команду меню **Edit > Copy** [Редактировать > Копировать].
3. Выделите требуемую целевую папку.
4. Выберите команду меню **Edit > Paste** [Редактировать > Вставить].

Примечание

Таблица соединений из папки "Connections [Соединения]" не может быть скопирована. Обратите внимание на то, что когда вы копируете списки текстов, предназначенных для оператора, принимаются только языки, установленные в целевом объекте.

26.3.2 Переименование объектов

Для переименования объекта действуйте следующим образом:

1. Выделите требуемый объект
2. Щелкните на имени выбранного объекта, чтобы активизировать функцию редактирования имени. Вокруг поля имени появляется рамка, а указатель мыши становится текстовым курсором.
3. Отредактируйте имя объекта. В общем случае, применяются соглашения об именах Windows 95/98.
4. Для закрытия функции переименования вы можете выполнить любое действие из следующих:
   - Нажмите клавишу ENTER, чтобы ввести новое имя. Если новое имя не разрешено, то восстанавливается предыдущее имя.
   - Нажмите клавишу ESC, чтобы прервать процедуру редактирования и восстановить предыдущее имя объекта.

26.3.3 Перемещение объектов

Непосредственное перемещение при помощи мыши (Drag & Drop [переместить и отпустить])

1. Обеспечьте, чтобы отображались как объект, который вы хотите переместить, так и требуемая целевая папка (в случае необходимости откройте дополнительное окно проекта).
2. Выделите объект, который вы хотите переместить, и держите нажатой левую кнопку мыши.
3. Нажмите клавишу SHIFT и переместите указатель мыши на целевую папку. При этом удерживайте нажатой левую кнопку мыши. Если вы пытаетесь переместить объект на запрещенное место, то вместо курсора появляется знак запрета.
4. Снова отпустите левую кнопку мыши.

Перемещение посредством команд меню

Используя команды меню, вы можете перемещать объекты из одной папки в другую только косвенно; это означает, что вы должны вырезать объект, который хотите переместить, и вставить его в новое место. Действуйте следующим образом:

1. Выделите объект, который вы хотите переместить.
2. Выберите команду меню **Edit > Cut [Редактировать > Вырезать]**.
3. Выделите требуемую целевую папку.
4. Выберите команду меню **Edit > Paste [Редактировать > Вставить]**.

26.3.4 Удаление объектов

Для удаления объекта действуйте следующим образом:

1. Выделите объект, который вы хотите удалить.
2. Чтобы удалить объект, вы можете выполнить любое из следующих действий:
   - Выберите команду меню **Edit > Delete [Редактировать > Удалить]**.
   - Нажмите клавишу DEL.
3. Подтвердите процедуру удаления щелчком на кнопке "Yes [Да]" в подсказке, появляющейся на экране.

27 Как программировать с помощью STEP 7

27.1 Как редактировать таблицу символов

27.1.1 Создание таблицы символов

Объект "Symbols" (таблица символов) создается автоматически ниже программы S7 или программы M7. Чтобы использовать символы совместно используемых данных в программе, их нужно назначить в таблице символов.

Чтобы создать таблицу символов, действуйте следующим образом:

1. Дважды щелкните на программе S7 или программе M7 в окне проекта так, чтобы объект "Symbols [Символы]" отобразился в правой половине окна.
2. С помощью команды меню **Insert > Symbol Table** [Вставить > Таблица символов] вставьте новую таблицу символов, если таблица была удалена или должна быть переписана.
3. Откройте объект "Symbols [Символы]", например, двойным щелчком на объекте.

Появляется окно, в котором для вас отображается таблица символов для того, чтобы ее редактировать.

27.1.2 Открытие таблицы символов

Чтобы открыть какую-либо таблицу символов в окне "Symbol Editor [Редактор символов]", действуйте следующим образом:

1. Чтобы открыть диалоговое окно для выбора таблицы символов, вы можете:
   - Выберите команду меню **Symbol Table > Open**.[Таблица символов > Открыть].
   - Щелкните на соответствующей кнопке на панели инструментов.
   - Нажмите CTRL + O.
2. В диалоговом окне, которое тогда отображается, выделите проект и требуемую таблицу символов.
3. Подтвердите посредством "OK.".

Чтобы открыть таблицу символов, которую вы редактировали недавно, выберите одну из команд меню **Symbol Table > 1, 2, 3** или **4 [Таблица символов > 1, 2, 3** или **4]**.

Примечание

Вы можете открывать таблицы символов также в SIMATIC Manager, открывая объект "symbols [символы]" двойным щелчком. Вы можете также открывать таблицу символов из окна "LAD/STL/FBD (KOP/AWL/FUP)", используя команду меню **Options > Symbol Table [Параметры > Таблица символов]**.

27.1.3 Определение отдельных символов

1. Активизируйте символическое представление в окне "LAD/STL/FBD (KOP/AWL/FUP)", используя команду меню **View > Display > Symbolic Representation [Вид > Отобразить > Символическое представление]**. Перед командой меню изображается метка, показывающая, что символическое представление активно.
2. Щелкните в сегменте по адресу, для которого вы хотите определить символ.
3. Выберите команду меню **Edit > Symbol** [Редактировать > Символ].
4. В появляющемся диалоговом окне введите символ, тип данных адреса и комментарий, если требуется. Символ должен быть уникальным во всей таблице символов и должен иметь длину не более 24 знаков. Использование кавычек (") не разрешается.
5. Подтвердите ваш ввод с помощью "OK". Определенный символ вводится в таблицу символов и вставляется в раздел кода вместо выбранного адреса.

27.1.4 Вставка символьных строк

Чтобы вставить пустую символьную строку перед позицией курсора, выберите команду меню **Insert > Symbol [Вставить > Символ]**.

Чтобы вставить одну или большее количество символьных строк из буфера обмена, вы можете:

- Щелкните левой кнопкой мыши на строке (не по номеру строки), начиная с которой должны вставляться символьные строки.
- Выберите команду меню **Edit > Paste [Редактировать > Вставить]**.
- Щелкните на соответствующей кнопке на панели инструментов
- Нажмите CTRL + V.

27.1.5 Удаление символьных строк

Чтобы вырезать выбранную символьную строку в буфер обмена, вы можете:

- Выберите команду меню **Edit > Cut** [Редактировать > Вырезать].
- Щелкните на соответствующей кнопке на панели инструментов
- Нажмите CTRL + X.

Чтобы удалить выбранные символьные строки без сохранения резервной копии, вы можете:

- Выберите команду меню **Edit > Delete** [Редактировать > Удалить].
- Нажмите DEL.

Обратите внимание на то, что если вы вырезаете или удаляете специальные свойства объекта, то вы не можете полностью вернуть назад это действие с помощью **Undo** [Отменить].

27.1.6 Фильтрация таблицы символов

Чтобы установить фильтр для отображения в активном окне:

1. Выберите команду меню **View > Filter** [Вид > Фильтр] или щелкните на кнопке "Filter [Фильтр]" на панели инструментов.
2. В диалоговом окне "Filter [Фильтр]" выберите существующий фильтр через соответствующий номер или определите новый фильтр.
3. Для этого щелкните на кнопке "New Filter [новый фильтр]".
4. Назначьте фильтру уникальное имя.
5. Выберите требуемую настройку параметров.
6. Щелкните на кнопке "Save [Сохранить]". Теперь вы можете выбрать новый фильтр в списковом окне

   или

- выбрать существующий фильтр из спискового окна "Filtered According To [Отфильтровано в соответствии с]" на панели инструментов.

Фильтры "All symbols [все символы]", "Unique symbols [уникальные символы]" и "Non-unique symbols [неуникальные символы]" являются предварительно определенными.

27.1.7 Сортировка таблицы символов

Чтобы установить критерии сортировки отображения в активном окне:

1. Выберите команду меню **View > Sort** [Вид > Сортировать].
2. Выберите требуемые критерии сортировки в диалоговом окне "Sort [Сортировка]".
3. Подтвердите посредством "OK"

   или

4. Поместите указатель мыши на верхнем краю заголовка столбца так, чтобы он изображался в виде диагональной двунаправленной стрелки.
5. Нажмите левую кнопку мыши. Появляется диалоговое окно с примечанием о том, как отменить действия и запрос.
6. Щелкните на кнопке "Yes [Да]" в диалоговом окне.

Таблица символов сортируется в соответствии с входными данными этого столбца. Если вы повторяете действие, то возвращается прежний порядок сортировки.

27.1.8 Поиск заданных строк

Если вы хотите обновить или изменить таблицу символов, то часто вы можете сэкономить ценное время, применяя поиск и замену текстовых объектов, которые хотите изменить.

Для поиска заданных строк в столбцах таблицы символов доступны следующие возможности:

1. Выберите команду меню Edit > Find/Replace [Редактировать > Найти/Заменить] или нажмите CTRL + F.
2. Введите требуемую искомую строку в поле "Find what [Что искать]" диалогового окна "Find and Replace [Найти и заменить]" или выберите существующую искомую строку в списковом окне.
3. Выберите требуемые опции (search range [диапазон поиска], find whole words only [искать только целые слова], match case [с учетом регистра]).
4. Подтвердите посредством "Find [Найти]".

Примечание

- Используя кнопку "Find [Найти]", команду меню **Edit > Continue [Редактировать > Продолжать]** или CTRL + W, вы можете продолжать искать следующее появление заданной искомой строки.
- Если вам требуется справка о диалоговом окне "Find and Replace [Найти и заменить]", то щелкните на кнопке "Help [Помощь]" или нажмите клавишу F1, пока диалоговое окно открыто.

27.1.9 Отображение и изменение свойств таблицы символов

Чтобы отобразить и изменить свойства активной таблицы символов:

1. Выберите команду меню **Symbol Table > Properties [Таблица символов > Свойства]**.
   Отображается диалоговое окно "Properties [Свойства]".
2. Выполните изменения в диалоговом окне.
3. Подтвердите посредством "OK".

27.1.10 Выделение символьных строк

Для выделения символьной строки, в которой расположен курсор, вы можете:

- Выберите команду меню **Edit > Select > Row [Редактировать > Выделить > Строка]**.
- Щелкните на номере строки слева от требуемой символьной строки.
- Нажмите SHIFT + SPACEBAR.

Примечание

Если вы выделяете этим способом всю строку, то атрибуты копируются только тогда, когда вы активизировали опцию "Copy with special object properties [Копировать со специальными свойствами объекта]" из-под **Options > Customize [Параметры > Настройка]**.

Если вы выбираете ячейку, а затем расширяете выбор с помощью SHIFT + CTRL + клавиша со стрелкой, то никакие существующие атрибуты не копируются вместе с этим выделением.

Чтобы выделить все строки активной таблицы символов, вы можете:

- Выберите команду меню **Edit > Select > All [Редактировать > Выделить > Все]**.
- Нажмите CTRL + A.

Чтобы отменить выделение, выберите команду меню **Edit > Undo Selection [Редактировать > Отменить выделение]**.

27.1.11 Копирование символьных строк в буфер обмена

Чтобы скопировать одну или более выделенных символьных строк в буфер обмена, вы можете:

- Выберите команду меню **Edit > Copy [Редактировать > Копировать]**.
- Щелкните на соответствующей кнопке на панели инструментов.
- Нажмите CTRL + C.

Тогда существующее содержимое буфера обмена заменяется.

Примечание

При копировании символа специальные свойства объекта не учитываются. Для предотвращения потери специальных свойств объекта во время копирования выберите в редакторе символов команду меню **Options > Customize [Параметры > Настроить]** и активизируйте триггерную кнопку "Copy with special object properties [Копировать со специальными свойствами объекта]".

27.1.12 Редактирование атрибута управления и наблюдения со стороны оператора

Чтобы назначить символу атрибут управления и наблюдения со стороны оператора:

1. Выберите в таблице символов строку, содержащую символ.
2. Выберите команду меню **Edit > Special Object Properties > Operator Control and Monitoring [Редактировать > Специальные свойства объекта > Управление и наблюдение со стороны оператора]**.
3. В диалоговом окне "Operator Control and Monitoring [Управление и наблюдение со стороны оператора]" активизируйте триггерную кнопку "Operator control and monitoring [Управление и наблюдение со стороны оператора]".
4. Выберите вкладку "General [Общие свойства]".
   Здесь имя символа отображается в том виде, в каком оно появляется в WinCC (S7 program name_symbol [Имя_символ программы S7]).
   При необходимости вы можете вводить дополнительную текстовую информацию о редактируемом символе в поле "Comment [Комментарий]".
5. Выберите вкладку "WinCC Attributes [Атрибуты WinCC]", чтобы редактировать атрибуты WinCC для выбранного символа.
6. В отображающейся таблице введите требуемые значения атрибутов.
7. Закройте диалоговое окно "Operating Control and Monitoring [Управление и наблюдение со стороны оператора]", щелкнув на кнопке "OK".
   Результат: Для редактируемого символа в столбце "B" таблицы символов вводится знак "X", показывающий, что теперь оператор может управлять этим символом и контролировать его.
8. Сохраните таблицу символов.

Примечание

Вы должны обратить внимание на то, что данные, введенные для операторского управления и контроля, сохраняются только тогда, когда сохраняется сама таблица символов. Если вы выходите из редактора символов без сохранения таблицы символов, то введенные вами данные для атрибутов WinCC будут потеряны.

При копировании символа специальные свойства объекта не учитываются. Для предотвращения потери специальных свойств объекта во время копирования выберите в редакторе символов команду меню **Options > Customize [Параметры > Настроить]** и активизируйте триггерную кнопку "Copy with special object properties [Копировать со специальными свойствами объекта]".

27.1.13 Редактирование атрибута сообщений

Чтобы иметь возможность наблюдать символы для двоичных переменных (например, меркера или входа процесса):

1. Выделите символ, для которого вы хотите определить атрибут сообщений.
2. Выберите команду меню **Edit > Special Object Properties > Message [Редактировать > Специальные свойства объекта > Сообщение]**.
3. В появляющемся диалоговом окне введите сообщения, которые должны отображаться, когда происходит изменение с 0 на 1 или с 1 на 0. Вы можете также исправлять сообщения, которые уже были введены.

Примечание

Для конфигурирования сообщений SCAN разрешаются только символы "Input [Вход]", "Output [Выход]" и "Bit memory [Меркер]" типа данных "Bool".

При копировании символа специальные свойства объекта не учитываются. Для предотвращения потери специальных свойств объекта во время копирования выберите в редакторе символов команду меню **Options > Customize [Параметры > Настроить]** и активизируйте триггерную кнопку "Copy with special object properties [Копировать со специальными свойствами объекта]".

27.1.14 Редактирование атрибута связи

Редактирование атрибута связи возможно только тогда, когда установлен NCM.

Чтобы определить для символа коммуникационные свойства:

1. Выделите символ, для которого вы хотите определить атрибут связи.
2. Выберите команду меню **Edit > Special Object Properties > Communication [Редактировать > Специальные свойства объекта > Связь]**.
3. В появляющемся диалоговом окне установите атрибут связи.
4. Когда этот атрибут установлен, вы можете еще в одной вкладке внутри диалогового окна определить атрибуты, существенные для связи.

Примечание

При копировании символа специальные свойства объекта не учитываются. Для предотвращения потери специальных свойств объекта во время копирования выберите в редакторе символов команду меню **Options > Customize [Параметры > Настроить]** и активизируйте триггерную кнопку "Copy with special object properties [Копировать со специальными свойствами объекта]".

27.1.15 Сохранение таблицы символов

Для сохранения активной таблицы символов вы можете:

- Выберите команду меню **Symbol Table > Save [Таблица символов > Сохранить]**.
- Щелкните на соответствующей кнопке на панели инструментов.
- Нажмите CTRL + S.

Таблица сохраняется.

27.1.16 Закрытие таблицы символов

Для закрытия активной таблицы символов вы можете:

- Выберите команду меню **Symbol Table > Close [Таблица символов > Закрыть]**.
- Нажмите CTRL + F4.

Если таблица символов отображается в расщепленном окне, то эта команда закрывает обе части окна.

Примечание:

- Если вы сделали изменения в таблице символов с тех пор, как она была сохранена в последний раз, то вы получите запрос о том, хотите ли вы сохранить таблицу символов перед закрытием.
- Если таблица символов содержит неуникальные или неполные символы, то вы получите запрос о том, хотите ли вы все еще закрыть таблицу символов.

Если вы закрываете без сохранения, то все не сохраненные изменения теряются.

Чтобы закрыть все открытые таблицы символов, выберите команду меню **Window > Close All [Окно > Закрыть все]**.

27.1.17 Выход из редактора символов

Чтобы закрыть все открытые таблицы символов и выйти из приложения, вы можете:

- Выберите команду меню **Symbol Table > Exit [Таблица символов > Выйти]**.
- Нажмите ALT + F4.

Примечание:

- Если вы сделали изменения в таблице символов с тех пор, как она была сохранена в последний раз, то вы получите запрос о том, хотите ли вы сохранить таблицу символов перед закрытием.
- Если таблица символов содержит неуникальные или неполные символы, то вы получите запрос о том, хотите ли вы все еще сохранить таблицу символов.
- Если вы закрываете без сохранения, то все не сохраненные изменения теряются.

27.2 Как импортировать и экспортировать таблицы символов

27.2.1 Импортирование таблицы символов

Вы можете импортировать символьные строки из файла с отличающимся форматом:

1. Откройте таблицу символов, в которую хотите импортировать данные.
2. Выберите команду меню **Symbol Table > Import [Таблица символов > Импортировать]**.
   Отображается диалоговое окно "Import [Импорт]".
3. Выберите каталог, в котором расположен требуемый файл.
4. Выберите тип файла, с которым файл был сохранен.
5. Выберите файл и подтвердите посредством "Open [Открыть]".

27.2.2 Импорт файла Excel в таблицу символов

Чтобы импортировать данные из приложения Microsoft Excel и экспортировать данные в приложение Microsoft Excel, используйте файловый формат DIF.

Чтобы импортировать данные из Excel, действуйте следующим образом:

1. Создайте в Excel таблицу с четырьмя столбцами ”Symbol [Символ]", ”Address [Адрес]", ”Data type [Тип данных]" и ”Comment [Комментарий]" и заполните эту таблицу.
2. Откройте диалоговое окно ”Save As [Сохранить как...]", используя команду меню **File > Save As [Файл > Сохранить как...]**.
3. В диалоговом окне выберите расширение файла ”.DIF" (Data Interchange Format [Формат обмена данными]).
4. Выберите путь для каталога и имя файла и закройте диалоговое окно.
5. Откройте таблицу символов.
6. Откройте диалоговое окно, используя команду меню **Table > Import [Таблица > Импортировать]**.
7. В диалоговом окне выберите файл *.DIF, который вы только что создали, и закройте диалоговое окно.

27.2.3 Экспорт таблицы символов

Вы можете экспортировать таблицу символов в файл с отличающимся форматом, сохраняя ее в этом формате.

1. Откройте таблицу символов.
2. Используйте фильтры, чтобы выбрать символы, которые вы хотите экспортировать.
3. Выберите команду меню **Symbol Table > Export [Таблица символов > Экспортировать]**.
   Отображается диалоговое окно "Export [Экспорт]".
4. Выберите каталог, в котором вы хотите сохранить таблицу символов.
5. Выберите формат файла и введите имя, под которым вы хотите сохранить таблицу символов.
6. Подтвердите посредством "OK".

Примечание

Обработка неуникальных символов при экспорте:

При экспорте во внимание принимаются только те символы, которые вы выбрали с помощью фильтра (например, все символы или только уникальные символы или только неуникальные символы).

27.3 Как изменить настройку параметров окна в редакторе символов

27.3.1 Включение/выключение панели инструментов

Чтобы включить или выключить отображение панели инструментов, выберите команду меню **View > Toolbar [Вид > Панель инструментов]**.

Когда отображается панель инструментов, около этой команды меню видна метка.

27.3.2 Включение/выключение строки состояния

Чтобы включить или выключить отображение строки состояния, выберите команду меню **View > Status Bar [Вид > Строка состояния]**.

Когда отображается строка состояния, около этой команды меню видна метка.

27.3.3 Расположение панели инструментов

Чтобы изменить местоположение отображаемой панели инструментов:

1. Переместите указатель мыши в свободную область на соответствующей панели.
2. Удерживая нажатой левую кнопку мыши, перетащите панель в требуемую позицию.
3. Снова отпустите левую кнопку мыши.

27.3.4 Регулирование размера окна для отображения

Чтобы увеличивать размер отображения в активном окне **постепенно**:

- Выберите команду меню **View > Zoom In [Вид > Раскрыть]**.
- Нажмите CTRL + Num+.

Чтобы уменьшать размер отображения в активном окне **постепенно**:

- Выберите команду меню **View > Zoom Out [Вид > Сжимать]**.
- Нажмите CTRL + Num-.

Чтобы установить размер отображения на **заданное значение**:

1. Выберите команду меню View > Zoom Factor [Вид > Коэффициент увеличения изображения].
2. В диалоговом окне "Zoom Factor [Коэффициент увеличения изображения]" выберите требуемый коэффициент увеличения изображения.
3. Подтвердите посредством "OK".

27.3.5 Разбиение окна таблицы

Разделение активного окна открытой таблицы символов

1. Выберите команду меню **Window > Split [Окно > Разделить]**.
   Указатель мыши изменяет свой вид.
2. С помощью мыши перетащите горизонтальную полосу в требуемое положение.
3. Нажмите левую кнопку мыши.

Изменение позиции разбиения в расщепленном окне

1. Установите указатель мыши на разделительную линию (разделитель) между двумя панелями окна.
   Указатель мыши изменяет свой вид.
2. С помощью мыши перетащите горизонтальную полосу в требуемое положение.
3. Нажмите левую кнопку мыши.

Удаление разделения

1. Выберите команду меню **Window > Split [Окно > Разделить]** или установите указатель мыши на разделительную линию (разделитель) между двумя частями окна.
2. С помощью мыши перетащите горизонтальную полосу к верхнему или нижнему краю окна.
3. Нажмите левую кнопку мыши

   или

- выберите команду меню **Window > Undo Split [Окно > Отменить разделение]**.

Примечание

Для переключения между частями окна используйте функциональную клавишу F6.

27.3.6 Изменение расположения окон таблиц символов

Если вы хотите расположить все окна, содержащие открытые таблицы символов так, чтобы они перекрывались, а строка заголовка каждого окна оставалась видимой, то вы можете:

- Выберите команду меню **Window > Arrange > Cascade [Окно > Расположить > Каскад]**.
- Нажмите SHIFT + F5.

Если вы хотите расположить все окна, содержащие открытые таблицы символов, равномерно сверху вниз, то выберите команду меню **Window > Arrange > Horizontally [Окно > Расположить > Горизонтально]**.

Если вы хотите расположить все окна, содержащие открытые таблицы символов, равномерно слева направо, то выберите команду меню **Window > Arrange > Vertically [Окно > Расположить > Вертикально]**.

Если вы хотите расположить значки свернутых окон равномерно по нижнему краю основного окна, то выберите команду меню **Window > Arrange Icons [Окно > Упорядочить значки]**.

27.3.7 Изменение параметров настройки программы

Чтобы устанавливать различные параметры настройки и опции программы, выберите команду меню **Options > Customize [Параметры > Настроить]**.

В диалоговом окне "Settings [Параметры настройки]" вы можете указать, должны ли сохраняться отдельные параметры настройки при закрытии таблицы символов. Например, изменение ширины столбца, модификация масштаба изображения, выбор критериев сортировки.

При копировании символов в другую таблицу символов специальные свойства объекта не учитываются.

Для предотвращения потери этих специальных свойств объекта во время копирования активизируйте триггерную кнопку "Copy with special object properties [Копировать со специальными свойствами объекта]".

27.4 Как создавать блоки

27.4.1 Создание блоков с помощью SIMATIC Manager

1. Откройте в вашем проекте папку "Blocks [Блоки]", в которую вы хотите вставить блок S7.
2. Выберите команду меню:
   - **Insert > S7 Block > Function Block** (FB) **[Вставить > Блок S7 > Функциональный блок** (FB)**]**, если вы хотите программировать функциональный блок
   - **Insert > S7 Block > Function** (FC) **[Вставить > Блок S7 > Функция** (FC)**]**, если вы хотите программировать функцию
   - **Insert > S7 Block > Organization Block** (OB) **[Вставить > Блок S7 > Организационный блок** (OB)**]**, если вы хотите программировать организационный блок
   - **Insert > S7 Block > Data Block** (DB) **[Вставить > Блок S7 > Блок данных** (DB)**]**, если вы хотите создать блок данных
   - **Insert > S7 Block > User-Defined Data Type** (UDT) **[Вставить > Блок S7 > Тип данных, определяемый пользователем** (UDT)**]**, если вы хотите создать определяемый пользователем тип данных
   - **Insert > S7 Block > Variable Table [Вставить > Блок S7 > Таблица переменных]**, если вы хотите создать таблицу переменных (VAT), чтобы наблюдать и изменять переменные в вашей программе пользователя для целей тестирования.

27.4.2 Создание блоков с помощью редактора пошагового ввода

Вы запускаете соответствующий редактор двойным щелчком по существующему блоку. Вы можете также создавать большее количество блоков, используя этот редактор.

1. В окне редактора выберите команду меню **File > New [Файл > Новый]**.
2. В последующем диалоге выберите программу пользователя S7, с которой должен быть связан блок.
3. Введите имя логического блока, который вы хотите создать.
4. Подтвердите посредством "OK".

Создается блок, и открывается окно для редактирования. Верхняя часть окна служит для того, чтобы редактировать таблицу описания переменных, а нижняя часть содержит раздел кода, где вы программируете новый блок.

Примечание

Когда вы создаете функциональный блок (FB), делается настройка того, можете ли вы описывать мультиэкземпляры в этом функциональном блоке. Это свойство устанавливается для каждого нового функционального блока в соответствии с вашими установками во вкладке "Editor [Редактор]" в диалоговом окне "Customize [Настройка]" (с помощью команды меню **Options > Customize [Параметры > Настроить]**).

Логические блоки можно создавать также, компилируя исходные файлы на AWL.

Вы можете создавать логические блоки также в SIMATIC Manager, вставляя их в соответствующую программу пользователя S7.

27.4.3 Создание блоков данных (DB)

Блоки данных создаются в SIMATIC Manager или в редакторе пошагового ввода точно так же, как и другие блоки.

1. В редакторе пошагового ввода выберите команду меню **File > New [Файл > Новый]** или щелкните на соответствующей кнопке на панели инструментов.
2. В диалоговом окне выберите программу пользователя S7, с которой вы хотите связать создаваемый блок.
3. Определите в диалоговом окне блок данных, который вы хотите создать. Вы не можете использовать DB0, потому что этот номер зарезервирован для системы.
4. В диалоговом окне "New Data Block [Новый блок данных]", выберите, какого типа блок данных вы хотите создать:
   - Data block (совместно используемый блок данных)
   - Data block referencing a user-defined data type [блок данных, ссылающийся на определяемый пользователем тип данных (совместно
     используемый блок данных)
   - Data block referencing a function block [блок данных, ссылающийся на функциональный блок] (экземплярный блок данных)

Для третьей опции вы должны выбрать также функциональный блок, которому должен принадлежать экземплярный блок данных.

Диалоговое окно "New Data Block [Новый блок данных]" отображается также тогда, когда вы в SIMATIC Manager впервые открываете существующий блок.

Примечание

При некоторых обстоятельствах STEP 7 предоставляет вам возможность хранить данные для различных функциональных блоков в единственном блоке данных (мультиэкземплярный блок данных, см. раздел Ввод мультиэкземпляров в таблицу описания переменных).

27.4.4 Установка свойств блока

1. Откройте папку блоков в программе S7.
2. Откройте блок двойным щелчком по нему или с помощью команды меню **Edit > Open Object [Редактировать > Открыть объект]**.
3. Выберите команду меню **File > Properties [Файл > Свойства]**, чтобы открыть вкладку "General - Part 1 [Общие - часть 1]", "General - Part 2 [Общие - часть 2]" или "Attributes [Атрибуты]".
4. Введите имя, семейство, версию и автора блока в страницу свойств "General [Общие]". Присвоение имени и семейства блоку помогает тогда, когда вы программируете вызовы блока в контактном плане. Когда вы выбираете блок в диалоговом окне "Program Elements [Элементы программы]", под окном списка немедленно отображается дополнительная информация, если она имеется в наличии. Вкладка "General - Part 2 [Общие – часть 2]" содержит информацию о длине блока, кода MC7 и локальных данных. Окно свойств содержит также следующие подробности:
   - Блок данных защищен от записи в программируемом контроллере.
   - Защита ноу-хау. Блок со свойством KNOW_HOW_PROTECT является защищенным блоком, который не может редактироваться.
   - Стандартный блок.
   - Несвязанный.
5. Введите значения для системных атрибутов в окне свойств "Attributes [Атрибуты]".

27.4.5 Установка приоритета адреса (символический/абсолютный)

В диалоговом окне свойств папки блоков вы можете указать, должно ли использоваться символическое значение или абсолютное значение, когда блоки открываются после того, как была изменена таблица символов. В предыдущих версиях STEP 7 решающим всегда было абсолютное значение.

Пример:

Сохраненный блок содержит оператор "A Symbol_A", где Symbol_A определен для абсолютного значения I0.1 в таблице символов. Теперь таблица символов изменяется. Тогда установка приоритета адреса воздействует на этот оператор следующим образом:

Приоритет адреса	Изменение в назначении "Symbol_A > I0.1"	Оператор после того, как блок открыт	Объяснение
Абсолютное значение	Symbol_A > I0.2	A I0.1	В операторе отображается абсолютное значение I0.1, потому что теперь ему не назначен никакой символ.
Абсолютное значение	Symbol_B > I0.1	A Symbol_B	В операторе отображается новый символ для все еще имеющего силу абсолютного значения I0.1.
Символ	Symbol_A > I0.2	A Symbol_A	Оператор остается неизменным. Отображается сообщение, касающееся измененного назначения символа.
Символ	Symbol_B > I0.1	A Symbol_A	Оператор отмечается как неверный (красный шрифт), потому что Symbol_A больше не определен.

27.4.6 Сравнение блоков

Чтобы сравнить блоки:

1. Выделите в SIMATIC Manager папку блоков или отдельные блоки, которые вы хотите сравнить.
2. Выберите команду меню **Options > Compare Blocks [Параметры > Сравнить блоки]**.
3. Выберите в диалоговом окне "Compare Blocks [сравнение блоков]" тип сравнения (ONLINE/offline или Path1/Path2 [путь1/путь2]).
4. Сравнение Path1/Path2 [путь1/путь2]: Выберите в SIMATIC Manager папку блоков или отдельные блоки, с которыми вы хотите сравнивать. Тогда блоки вводятся в диалоговое окно автоматически.
5. В диалоговом окне щелкните на кнопке "OK".

Результаты сравнения отображаются в другом диалоговом окне.

27.4.7 Переключение

Можно переключать следующие блоки и адреса:

- Входы, выходы
- Меркеры, таймеры, счетчики
- Функции, функциональные блоки

Для переключения:

1. Выберите в SIMATIC Manager папку блоков или отдельные блоки, которые вы хотите переключить.
2. Выберите команду меню **Options > Rewire [Параметры > Переключить]**.
3. В диалоговом окне "Rewire [Переключение]" введите в таблицу требуемые замены (старый адрес/новый адрес).
4. Если вы хотите переключить области адресов (BYTE, WORD, DWORD), то выберите опцию "All addresses within the specified address area [Все адреса в пределах заданной области адресов]".
   Пример: Вы вводите в качестве областей адресов IW0 и IW4. Тогда адреса I0.0 - I1.7 будут переключены на адреса I4.0 - I5.7. В этом случае адреса из переключенной области (например, I0.1) больше не могут вводиться в таблицу индивидуально.
5. Щелкните на кнопке "OK"..

Это запускает процесс переключения. После завершения переключения вы можете указать в диалоговом окне, хотите ли вы увидеть информационный файл о переключении. Этот информационный файл содержит списки адресов "Old address [Старый адрес]" и "New address [Новый адрес]". Отдельные блоки перечисляются с указанием количества процессов переключения, которые были выполнены в каждом из них.

27.5 Как работать с библиотеками

27.5.1 Создание библиотеки

Чтобы создать библиотеку, действуйте следующим образом:

1. Выберите команду меню **File > New [Файл > Новый]**.
2. В появляющемся диалоговом окне выберите путь назначения (диск и каталог) для новой библиотеки.

Примечание

Имена каталогов в пути должны быть длиной не более восьми символов. Иначе могут быть проблемы при архивировании и использовании "C для M7" (компилятор Borland).

---

3. В диалоговом окне введите имя новой библиотеки.
4. Под "Type [Тип]" вы можете указать, должна ли библиотека создаваться для текущей версии STEP 7 (значение по умолчанию) или для редактирования с помощью более старой версии STEP 7.
5. Щелкните на кнопке "OK", чтобы создать библиотеку. Появляется разделенное окно, показывающее в левой половине значок библиотеки.

Теперь вы можете копировать программы S7/M7 из проектов в библиотеку или создавать программы S7/M7 в библиотеке (используя команду меню **Insert > Program > S7 Program [Вставить > Программа > Программа S7]** или **Insert > Program > M7 Program [Вставить > Программа > Программа M7]**).

27.5.2 Использование библиотеки

Чтобы использовать библиотеку, действуйте следующим образом:

1. Выберите команду меню File > Open [Файл > Открыть].
2. В появляющемся диалоговом окне выберите библиотеку.
3. В окне библиотеки выберите программу S7, компоненты которой вы хотите использовать.
4. Скопируйте требуемую программу пользователя или требуемые блоки в программу пользователя в структуре вашего текущего проекта.

27.5.3 Копирование библиотеки

Чтобы скопировать библиотеку, действуйте следующим образом:

1. Выберите копируемую библиотеку.
2. В SIMATIC Manager выберите команду меню **File > Save [Файл > Сохранить]**.
3. В диалоговом окне "Save As [Сохранить как...]" решите, хотите ли вы выполнить переупорядочение перед сохранением или нет. В случае старых библиотек или библиотек, в которых вы сделали много изменений, вам нужно выбрать опцию "Rearrange before saving [Переупорядочить перед сохранением]", чтобы хранение данных было оптимизировано и была проверена структура библиотеки.
4. В диалоговом окне "Save Library As [Сохранить библиотеку как...]" введите имя новой библиотеки и, в случае необходимости, нового пути хранения. Подтвердите посредством "OK".

27.5.4 Копирование части библиотеки

Если вы хотите копировать часть библиотеки, такую как программные средства, блоки и т.д., то действуйте следующим образом:

1. Выделите часть библиотеки, которую вы хотите копировать.
2. Выберите в SIMATIC Manager команду меню **Edit > Copy [Редактировать > Копировать]**.
3. Выберите папку, в которой должна храниться скопированная часть библиотеки.
4. Выберите команду меню **Edit > Paste [Редактировать > Вставить]**.

27.5.5 Удаление библиотеки

Чтобы удалить библиотеку, действуйте следующим образом:

1. Выберите в SIMATIC Manager команду меню **File > Delete** [Файл > Удалить].
2. В диалоговом окне "Delete [Удалить]" активизируйте кнопку выбора "Library [Библиотека]". В списковом окне ниже нее перечисляются библиотеки.
3. Выделите часть библиотеки, которую вы хотите удалить и подтвердите посредством "OK".
4. Подтвердите запрос посредством "Yes [Да]".

27.5.6 Удаление части библиотеки

Чтобы удалить часть библиотеки, действуйте следующим образом:

1. Выделите часть библиотеки, которую вы хотите удалить.
2. Выберите в SIMATIC Manager команду меню **Edit > Delete [Редактировать > Удалить]**.
3. Подтвердите запрос посредством "Yes [Да]".

27.6 Определение представления в окне редактирования

27.6.1 Уменьшение масштаба изображения

Вы можете пошагово уменьшать размер изображения, включая шрифты, для каждого окна (блок данных, логический блок или исходный файл на AWL).

Чтобы уменьшить масштаб изображения, действуйте следующим образом:

1. Активизируйте окно, содержимое которого вы хотите уменьшить на один шаг.
2. Выберите команду меню **View > Zoom Out** [Вид > Сжать]. Если вы еще не достигли минимального размера масштаба изображения, то активное изображение уменьшится на один шаг.

27.6.2 Изменение масштаба изображения

Вы можете увеличивать или уменьшать размер изображения, включая шрифты, путем ввода масштабного коэффициента, вы можете также восстановить стандартный размер для каждого окна (блок данных, логический блок или исходный файл на AWL).

Чтобы установить масштабный коэффициент, действуйте следующим образом:

1. Активизируйте окно, для содержимого которого вы хотите изменить размер изображения.
2. Выберите команду меню **View > Zoom Factor [Вид > Масштабный коэффициент]**.
3. В диалоговом окне введите требуемые установки и подтвердите посредством "OK".

27.6.3 Установка разделения окна

Как логические блоки, так и исходные файлы отображаются в разделенных окнах. В случае логических блоков верхняя часть содержит таблицу описания переменных, а нижняя часть – раздел кода. Активное окно для исходных файлов на AWL разбито на программную часть и список сообщений об ошибках. Вы можете изменять размер одной части относительно другой, перемещая разделитель.

Действуйте следующим образом:

- Щелкните мышью по разделительной линии и, удерживая нажатой кнопку, перемещайте мышь в направлении, в котором вы хотите передвинуть разделительную линию.
- Выберите команду меню **Window > Move Split [Окно > Переместить разделитель]**. Происходит выделение разделительной линии, и вы можете перемещать ее либо с помощью мыши, либо с помощью клавиш управления курсором.

27.6.4 Регулирование ширины столбцов

Вы можете изменить ширину любого столбца таблицы описаний.

Действуйте следующим образом:

- Переместите указатель мыши на линию, разделяющую два столбца, пока он не изменит свою форму на двойную стрелку. Удерживая нажатой левую кнопку мыши и перемещая мышь по горизонтали, вы можете визуально регулировать ширину столбца.
- Поместите курсор в поле столбца, ширину которого вы хотите изменить, и выберите команду меню **View > Column Width [Вид > Ширина столбца]**. Тогда в диалоговом окне вы можете задать ширину столбца или с помощью кнопки "Optimum Width [Оптимальная ширина]" установить ширину с подгонкой под содержимое столбца.

27.6.5 Переключение между языками программирования

В пределах стандартного пакета STEP 7 в вашем распоряжении имеются три формы представления языка для программирования блоков; это контактный план (Ladder Logic = LAD, или KOP), функциональный план (Function Block Diagram = FBD, или FUP) и список операторов (Statement List = STL, или AWL).

1. Активизируйте окно логического блока, для которого вы хотите изменить форму представления программы.
2. Выберите одну из следующих команд меню:
   - **View > LAD [Вид > LAD** (**KOP)]**, чтобы редактировать раздел кода, используя контактный план.
   - **View > FBD [Вид > FBD** (**FUP)]**, чтобы редактировать раздел кода, используя функциональный план.
   - **View > STL [Вид > STL** (**AWL)]**, чтобы редактировать раздел кода, используя список операторов.

Примечание

Переключение с контактного плана на функциональный план и обратно возможно в любое время.

Переключение с AWL на KOP/FUP возможно только для операторов AWL, имитирующих полный набор параметров соответствующего элемента контактного плана и поддерживающих правильную последовательность. Параметры, не использованные в AWL, должны устанавливаться в ”NOP 0”.

27.7 Как работать с таблицей описания переменных

27.7.1 Вставка пустых строк в таблицы описания переменных

Чтобы вставить пустую строку перед текущей строкой:

1. Поместите курсор в строке таблицы, перед которой вы хотите вставить пустую строку.
2. Выберите команду меню **Insert > Declaration Row > Before Selection [Вставить > Строка описания > Перед выбором]**.

Чтобы вставить пустую строку после текущей строки:

- Поместите курсор в ячейке "Comment [Комментарий]" этой строки и нажмите RETURN или
- Выберите команду меню **Insert > Declaration Row > After Selection [Вставить > Строка описания > После выбора]**.

27.7.2 Ввод элементарных типов данных в таблицу описаний

Чтобы ввести новое описание, действуйте следующим образом:

1. Введите имя переменной после требуемого типа описания.
2. Затем переместите курсор в соседнюю ячейку, используя клавишу табуляции.
3. Затем введите:
   - тип данных,
   - начальное значение (необязательно),
   - комментарий (необязателен).

Когда строка завершается, переменной назначается адрес.

Каждый раз, когда вы редактируете ячейку таблицы, выполняется проверка синтаксиса и любые ошибки отображаются красным шрифтом. Вы не обязаны исправлять эти ошибки немедленно; вы можете продолжить редактирование и исправить ошибки на более поздней стадии.

27.7.3 Ввод мультиэкземпляра в таблицу описания переменных

1. Откройте функциональный блок, из которого должны вызываться подчиненные функциональные блоки.
2. В описании переменных вызывающего функционального блока определите статическую переменную для каждого вызова функционального блока, с экземпляром которого вы не хотите использовать экземплярный блок данных.
   - Поместите курсор в пустой строке с описанием "stat" во втором столбце.
   - В столбце "Name [Имя]" после типа описания "stat" введите имя для вызова FB.
   - В столбце "Type [Тип]" введите функциональный блок, который вы хотите вызвать, в виде абсолютного адреса или посредством его символического имени.
   - В столбце комментария вы можете вводить любые требуемые объяснения.

Вызовы в разделе кода

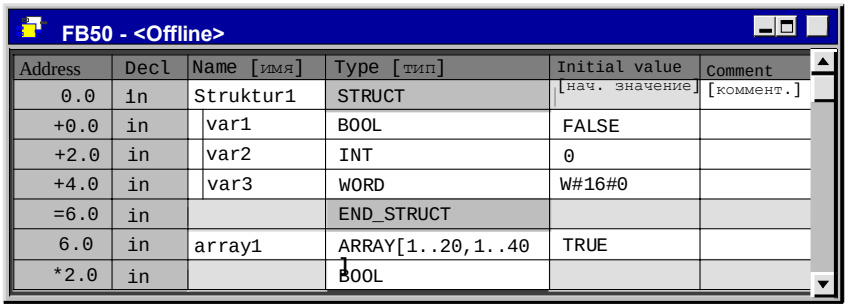
Если вы объявили мультиэкземпляры, то вы можете использовать вызовы FB без указания экземплярного DB.

Пример: Если определена статическая переменная "Имя Motor_1, Тип данных: FB20", то экземпляр можно вызывать следующим образом:

```
Call Motor_1          // Вызов FB20 без экземплярного DB
```

27.7.4 Ввод элементов данных типа STRUCT

1. Опишите тип данных:
   - Поместите курсор в ячейке столбца "Data Type [Тип данных]" и выберите команду меню **Insert > Data Type > Complex Type > STRUCT [Вставить > Тип данных > Составной тип > STRUCT]**.
   - Выберите ячейку столбца "Data Type [Тип данных]" и нажмите правую кнопку мыши. Выберите соответствующий тип данных в контекстно-зависимом меню.
   - Введите в ячейку столбца "Data Type [Тип данных]" ключевое слово STRUCT.
2. В столбце "Name [Имя]" введите символическое имя и выйдите из строки таблицы, используя клавишу TAB или RETURN. Первоначально вставятся одна пустая строка и последняя строка описания (END_ STRUCT).
3. В пустой строке введите элементы структуры, определяя их символическое имя, тип данных, (необязательное) начальное значение и произвольный комментарий. Вы можете добавлять большее количество строк, используя функции меню "Insert [Вставка]" или нажимая RETURN, либо вы можете дублировать переменные или снова их удалять при помощи меню "Edit [Редактирование]".



Address	Decl	Name [имя]	Type [тип]	Initial value [нач. значение]	Comment [коммент.]
0.0	in	Struktur1	STRUCT		
+0.0	in	var1	BOOL	FALSE	
+2.0	in	var2	INT	0	
+4.0	in	var3	WORD	W#16#0	
=6.0	in		END_STRUCT		
6.0	in	array1	ARRAY[1..20,1..40	TRUE	
*2.0	in		BOOL		

27.7.5 Ввод элементов данных типа ARRAY

1. Поместите курсор в ячейке столбца "Data Type [Тип данных]" таблицы описаний.
2. Выберите команду меню **Insert > Data Type > Complex Type > ARRAY** [Вставить > Тип данных > Составной тип > ARRAY]. Тогда в выбранную ячейку вводится ARRAY. Вы можете также напечатать слово ARRAY, используя клавиатуру.
3. Сразу после ARRAY введите открывающую квадратную скобку, нижний предел индекса, две точки, верхний предел индекса и закрывающую квадратную скобку, например, ARRAY [1 .. 14] для одномерного массива или ARRAY [1 .. 20,1 .. 24] для двумерного массива.
4. В поле столбца "Initial Value [Начальное значение]" вы можете ввести начальные значения для отдельных элементов массива (см. примеры ниже).
5. В поле столбца " Comment [Комментарий]" вы можете ввести комментарии к массиву.
6. Завершите ввод строки таблицы при помощи клавиши TAB или RETURN.
7. В создаваемой автоматически второй строке введите тип данных элементов массива.

Примеры ввода начальных значений в массивы

- Индивидуально:
  Вы назначаете каждому отдельному элементу свое собственное начальное значение. Значения перечисляются через запятую.
- Коэффициент повторения:
  Вы назначаете одно и то же начальное значение множеству элементов. Значение задается в скобках с предшествующим коэффициентом повторения для указания количества элементов.

Тип	**Начальное значение**	**Объяснение**
ARRAY[1..14]	1234	Начальное значение 1234 назначено только первому элемент у массива.
ARRAY[1..14]	1234, 56, 78, 90	Начальные значения 1234, 56, 78, 90 назначены первым четырем элементам массива в этом порядке.
ARRAY[1..14]	14 (9876)	Начальное значение 9876 назначено всем 14 элементам массива.

27.7.6 Копирование переменных в таблицах описаний

1. Выделите переменные, которые вы хотите копировать:
   - Щелкните на ячейке "Address [Адрес]", чтобы выделить переменную.
   - Удерживая нажатой клавишу SHIFT, щелкните левой кнопкой мыши на ячейке "Address [Адрес]" в другой строке. Тогда выделяются все строки между первой и второй выбранными переменными (выбор множества переменных).
2. Выберите команду меню **Edit > Copy [Редактировать > Копировать]** или соответствующую кнопку на панели инструментов.
3. Поместите курсор на позиции, после которой вы хотите вставить копируемую переменную, и выберите команду меню **Edit > Paste [Редактировать > Вставить]** или соответствующую кнопку на панели инструментов.

Копируемые переменные вставляются. Чтобы символические имена переменных оставались уникальными, именам скопированных переменных автоматически дается дополнительный порядковый номер.

27.7.7 Удаление переменных в таблицах описаний

1. Выделите переменные, которые вы хотите удалить:
   - Щелкните на ячейке "Address [Адрес]", чтобы выделить переменную.
   - Удерживая нажатой клавишу SHIFT, щелкните левой кнопкой мыши на ячейке "Address [Адрес]" в другой строке. Тогда выбираются все строки между первой и второй выбранными переменными (выбор множества переменных).
2. Выберите команду меню **Edit > Cut [Редактировать > Вырезать]** или команду меню **Edit > Delete [Редактировать > Удалить]** или соответствующую кнопку на панели инструментов.

Примечание

При удалении массивов [ARRAY] и структур [STRUCT]:

Если вы выделяете для удаления первую строку массива [ARRAY], то выделяется также принадлежащая ему вторая строка.

Если вы выделяете для удаления первую строку структуры [STRUCT], то выделяются также все строки до строки END STRUCT включительно.

27.7.8 Изменение ширины столбцов

Вы можете изменять ширину столбцов таблицы. Действуйте следующим образом:

1. Поместите указатель мыши между двумя столбцами.
2. При нажатой левой кнопке мыши перемещайте указатель мыши в горизонтальном направлении.

Альтернативно, вы можете изменять ширину столбца, используя команду меню **View > Column Width** [Вид > Ширина столбца]. Если вам не требуются необязательные записи для комментария или начального значения, то таким способом вы можете минимизировать размер этих столбцов, чтобы полностью сосредоточиться на других столбцах.

27.7.9 Назначение системных атрибутов

Системные атрибуты могут назначаться блокам и параметрам. Они управляют конфигурацией сообщений и конфигурацией соединений, функциями интерфейса с оператором и конфигурацией управления процессом.

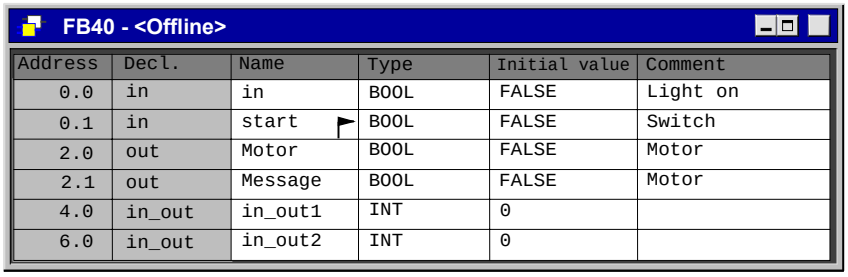
Вы можете назначать системные атрибуты для параметров в таблице описания переменных:

1. Выберите имя параметра в таблице описания переменных.
2. Выберите команду меню **Edit > Object Properties [Редактировать > Свойства объекта]**.
3. В появляющемся на экране диалоговом окне "Parameter Properties [Свойства параметров]" введите требуемый системный атрибут и соответствующее значение.

Список действительных системных атрибутов для параметров вы найдете в интерактивной справке по системным атрибутам (Jumps to Language Descriptions and Help on Blocks and System Attributes [переходы к описаниям языков и справке по блокам и системным атрибутам]).

Идентификаторы назначенных системных атрибутов

Если вы назначили переменной системные атрибуты, то в столбце "Name [Имя]" появляется символ флажка (как показано на следующем рисунке около "start"). Двойной щелчок по "флажку" открывает диалоговое окно "Parameter Properties [Свойства параметров]".



Address	Decl.	Name	Type	Initial value	Comment
0.0	in	in	BOOL	FALSE	Light on
0.1	in	start	BOOL	FALSE	Switch
2.0	out	Motor	BOOL	FALSE	Motor
2.1	out	Message	BOOL	FALSE	Motor
4.0	in_out	in_out1	INT	0	
6.0	in_out	in_out2	INT	0	

Пояснения к рисунку: Address –адрес; Decl. – описание; Name – имя; Type – тип; Initial value – начальное значение; Comment – комментарий

27.7.10 Ввод комментариев к блокам и комментариев к сегментам

1. Активизируйте комментарии при помощи команды меню **View > Display > Comments [Вид > Отобразить > Комментарии]** (перед командой меню видна метка).
2. Щелчком мыши расположите курсор в сером поле под именем блока или под именем сегмента. Серое поле комментария становится белым и имеет границу.
3. В открытом текстовом поле введите свой комментарий. Вам разрешается использовать 64 Килобайта на блок для комментариев к блоку и комментариев к сегментам.
4. Выйдите из текстового поля, щелкнув мышью вне этого поля, нажав клавишу TAB или комбинацию клавиш SHIFT+TAB.
5. Вы можете выключить комментарии, повторно выбрав команду меню **View > Display > Comments [Вид > Отобразить > Комментарии]** (метка исчезает).

27.7.11 Создание шаблонов сегментов

Для создания шаблона сегмента в проекте должна иметься библиотека, в которой Вы можете хранить шаблона сегмента.

1. В случае необходимости создайте в SIMATIC Manager новую библиотеку.
2. Откройте блок, содержащий сегмент(ы), из которых вы хотите создать шаблон сегмента.
3. Замените заголовок, комментарий или адреса трафаретными символами требуемым образом. Вы можете использовать в качестве трафаретных символов строки с %00 по %99.
4. Выберите сегмент(ы), который(е) вы хотите включить в шаблон сегмента.
5. Выберите команду меню **Edit > Create Network Template [Редактировать > Создать шаблон сегмента]**.
6. В появляющемся диалоговом окне введите комментарий для каждого использованного трафаретного символа.
7. В появляющемся браузере выберите папку исходных файлов библиотеки и введите имя шаблона сегмента.
8. Подтвердите ваш ввод щелчком на кнопке "OK". Шаблон сегмента сохраняется в выбранной библиотеке.

27.7.12 Вставка шаблона сегмента в программу

1. В блоке щелкните по сегменту, после которого вы хотите вставить шаблон сегмента.
2. Откройте каталог "Program Elements [Элементы программы]" (команда меню **View > Catalog** [Вид > Каталог]).
3. В каталоге откройте папку "S7 Program [Программа S7]" подходящей библиотеки.
4. Дважды щелкните по шаблону сегмента.
5. В диалоговом окне введите требуемые замены для трафаретных символов в шаблоне сегмента.
6. Щелкните на кнопке "OK". Тогда шаблон сегмента вставится после текущего сегмента.

Примечание

Вы можете также копировать шаблон из каталога в окно редактора, используя буксировку (drag & drop).

27.8 Как вводить элементы контактного плана

27.8.1 Ввод элементов контактного плана

1. Выберите в сегменте точку, после которой вы хотите вставить элемент контактного плана.
2. Вставьте в сегмент требуемый элемент одним из следующих способов:
   - В меню "Insert [Вставка]" выбрать подходящую команду меню, например, **Insert > LAD Element > Normally Open Contact [Вставить > Элемент КОР > Нормально разомкнутый контакт]**.
   - На панели инструментов щелкнуть на кнопке нормально разомкнутого контакта, нормально замкнутого контакта или выходной катушки.
   - Ввести нормально разомкнутый контакт, нормально замкнутый контакт или выходную катушку при помощи функциональной клавиши F2, F3 или F7.
   - Выберите команду меню **Insert > Program Elements [Вставить > Элементы программы]**, чтобы открыть диалоговое окно "Program Elements [Элементы программы]", и выбрать в каталоге требуемый элемент.

Выбранный элемент контактного плана вставляется, и для представления адресов и параметров используются символы вопросительного знака (??.?).

Примечание

Вы можете также редактировать раздел кода, выделяя существующие элементы контактного плана и затем выбирая одну из команд меню **Edit > Cut [Редактировать > Вырезать]**, **Edit > Copy [Редактировать > Копировать]** или **Edit > Paste [Редактировать > Вставить]**.

27.8.2 Ввод и редактирование адресов или параметров в элементах контактного плана

Когда элемент контактного плана вставлен, символы ??? и ... используются в качестве замещающих символов для адресов и параметров.

Красные символы ??? замещают адреса и параметры, которые должны быть подключены.

Черные символы ... замещают адреса и параметры, которые могут быть подключены.

1. Поместите курсор на замещающих символах, щелкнув по ним мышью или применив клавишу TAB.
2. Напечатайте адрес или параметр вместо замещающих символов (прямая или косвенная адресация). Если активизировано отображение выбора символов (команда меню **View > Display > Symbol Selection [Вид > Отобразить > Выбор символа]**), то отображается список существующих символических имен. Символическое имя, начинающееся с введенных символов, выделяется и может быть введено нажатием клавиши RETURN.
3. Нажмите RETURN. Программное обеспечение выполняет проверку синтаксиса.
   - Если синтаксис верен, то адрес форматируется и отображается черным шрифтом, а редактор автоматически открывает следующее текстовое поле, требующее ввода адреса или параметра.
   - Если имеется синтаксическая ошибка, то выход из поля ввода не происходит, и в строке состояния отображается сообщение об ошибке. Снова нажмите клавишу RETURN; происходит выход из поля ввода, а неправильный ввод отображается красным курсивным шрифтом.

27.8.3 Замена адресов или параметров в элементах контактного плана

1. С помощью клавиши INSERT переключитесь в режим замены. Текущий режим отображается в строке состояния в правом нижнем углу экрана.
2. Поместите курсор в текстовом поле адреса или параметра, щелкнув по нему мышью или применив клавишу TAB.
3. Перепишите адрес или параметр.
4. Нажмите RETURN. Программное обеспечение выполняет проверку синтаксиса
   - Если синтаксис правилен, то адрес форматируется и отображается черным шрифтом, а редактор автоматически открывает следующее текстовое поле, требующее ввода адреса или параметра.
   - Если имеется синтаксическая ошибка, то выход из поля ввода не происходит и в строке состояния отображается сообщение об ошибке. Снова нажмите клавишу RETURN; происходит выход из поля ввода, а неправильный ввод отображается красным курсивным шрифтом.

27.8.4 Замена элементов контактного плана

Режим замены позволяет заменять однотипные элементы контактного плана. Это имеет то преимущество, что не нужно снова вводить адреса и параметры. Элемент контактного плана можно заменить только элементом того же самого типа. Например, вы можете заменить нормально разомкнутый контакт на нормально замкнутый контакт, RS-триггер на SR-триггер или таймер на счетчик.

1. С помощью клавиши INSERT переключитесь в режим замены. Текущий режим отображается в строке состояния в правом нижнем углу экрана.
2. Выделите элемент контактного плана, который вы хотите заменить.
3. Вставьте в сегмент требуемый элемент одним из следующих способов:
   - В меню "Insert [Вставка]" выберите команду меню, например, **Insert > LAD Element > Coil [Вставить > Элемент КОР > Катушка]**.
   - На панели инструментов щелкните на кнопке нормально разомкнутого контакта, нормально замкнутого контакта или выходной катушки .
   - Введите нормально разомкнутый контакт, нормально замкнутый контакт или выходную катушку при помощи функциональной клавиши F2, F3 или F7.
   - Нажмите F11 или выберите команду меню **Insert > Program Elements** [Вставить > Элементы программы], чтобы открыть диалоговое окно "Program Elements [Элементы программы]", и выберите в каталоге требуемый элемент.

Существующий элемент контактного плана заменяется новым, который вы выбрали.

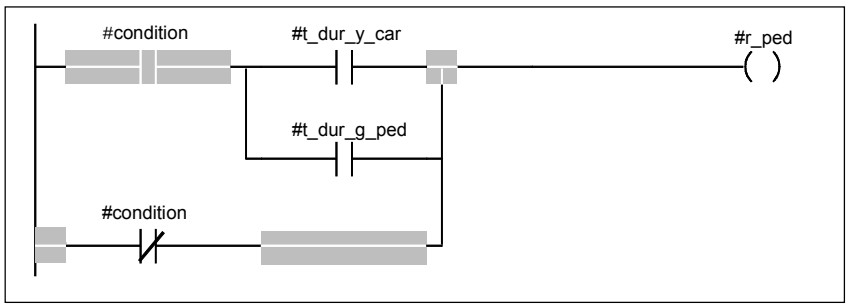
- Если вы снова нажмете INSERT, то переключитесь обратно в режим вставки. Текущий режим отображается в строке состояния в правом нижнем углу экрана.

27.8.5 Выделение элементов в сегментах контактного плана

Вы получаете доступ к сегменту, щелкнув мышью по элементу контактного плана в сегменте. В пределах сегмента вы можете выделять три основные области, щелкая по ним один раз мышью:

- элементы контактного плана, такие как контакт или блок
- точки соединения (узлы)
- пустые элементы (линии или открытые ветви)

Вы можете выделить одновременно только одну область. Следующий рисунок показывает примеры нескольких выделений, сделанных одновременно.



В диалоговом окне "Customize [Настройка]" во вкладке "LAD/FBD (KOP/FUP)" вы можете самостоятельно выбрать цвет выделения. Вы открываете это диалоговое окно при помощи команды меню **Options > Customize [Параметры > Настроить]**.

27.8.6 Вставка дополнительных сегментов контактного плана

Чтобы вставить новый сегмент, выберите команду меню **Insert > Network [Вставить > Сегмент]** или щелкните на соответствующей кнопке на панели инструментов. Новый сегмент вставится ниже выбранного сегмента. Он содержит только одну ветвь с одной катушкой.

Если вы вводите большее количество элементов, чем может отображаться на вашем экране, то сегмент на экране сдвигается влево. Используя команды меню **View > Zoom Out/Zoom In/Zoom Factor [Вид > Сжать/Раскрыть/ Масштабный коэффициент]**, вы можете регулировать размер изображения так, чтобы получить улучшенный обзор.

Вы получаете доступ к сегменту, щелкая мышью по элементу контактного плана в сегменте. В пределах сегмента вы можете выбрать три основные области, щелкнув по ним один раз мышью.

27.8.7 Создание параллельных ветвей в сегментах контактного плана

Для создания команд ИЛИ (OR) в сегментах контактного плана, вы должны создавать параллельные ветви.

Чтобы создать параллельную ветвь, действуйте следующим образом:

1. Выделите элемент, перед которым вы хотите открыть параллельную ветвь.
2. Чтобы открыть параллельную ветвь, используйте один из следующих методов:
   - Выберите команду меню **Insert > LAD Element > Open Branch [Вставить > Элемент КОР > Открыть ветвь]**.
   - Нажмите функциональную клавишу F8.
   - Щелкните на соответствующей кнопке на панели инструментов.
3. Вставьте требуемые элементы контактного плана в открытую параллельную ветвь.
4. На "основной ветви" выберите элемент, после которого вы хотите закрыть параллельную ветвь.
5. Чтобы закрыть параллельную ветвь, используйте один из следующих методов:
   - Выберите команду меню **Insert > LAD Element > Close Branch** [Вставить > Элемент КОР > Закрыть ветвь].
   - Нажмите функциональную клавишу F9.
   - Щелкните на соответствующей кнопке на панели инструментов.

27.8.8 Создание новых ветвей в сегментах контактного плана

Вы можете вставить несколько параллельных ветвей в один сегмент контактного плана.

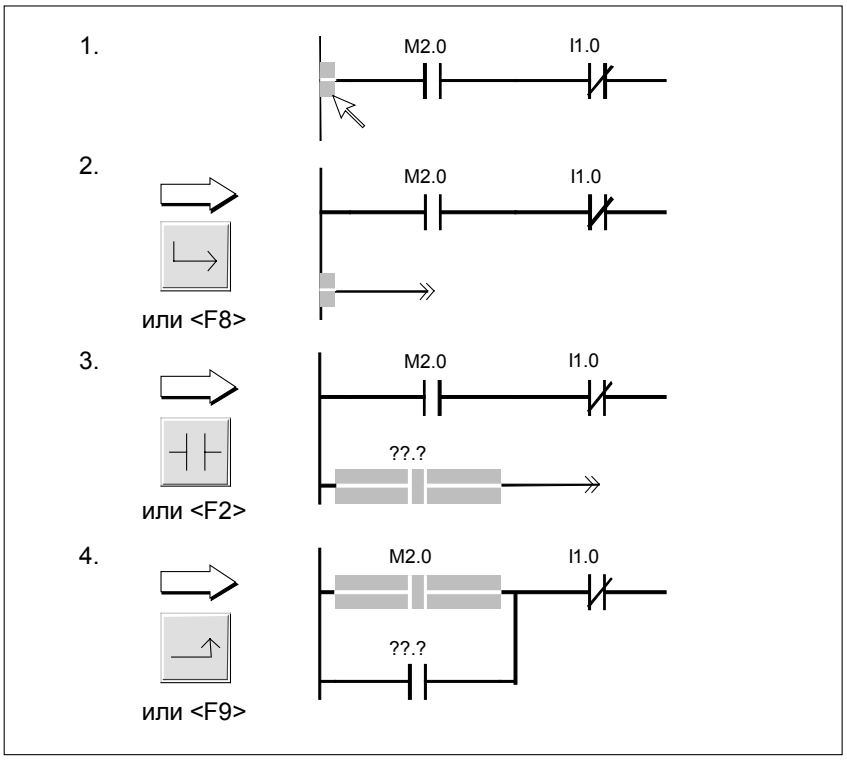
1. С помощью указателя мыши выберите исходную точку ветви, ниже которой вы хотите вставить новую ветвь.
2. Чтобы открыть новую ветвь, используйте один из следующих методов:
   - Выберите команду меню **Insert > LAD Element > Open Branch [Вставить > Элемент КОР > Открыть ветвь]**.
   - Нажмите функциональную клавишу F8.
   - Щелкните на соответствующей кнопке на панели инструментов.

27.8.9 Создание закрытой ветви в сегментах контактного плана

Чтобы создать закрытую ветвь, действуйте следующим образом:

1. Выделите элемент, перед которым вы хотите открыть параллельную ветвь.
2. Откройте параллельную ветвь с помощью F8.
3. Вставьте элемент контактного плана.
4. Закройте ветвь с помощью F9.

Следующий пример показывает, как создается ветвь при помощи только функциональных клавиш или кнопок панели инструментов.



Когда вы закрываете параллельные ветви, добавляются все необходимые пустые элементы. В случае необходимости ветви располагаются так, чтобы они не пересекались. Если вы закрываете ветвь непосредственно из параллельной ветви, то ветвь закрывается после следующего возможного элемента контактного плана

27.8.10 Открытие закрытых параллельных ветвей контактного плана

Вы можете открыть закрытую параллельную ветвь следующим образом:

1. Выделите узел, в котором параллельная ветвь снова встречается с основной ветвью.
2. Удалите выделение с помощью команды меню **Edit > Cut [Редактировать > Вырезать]**.

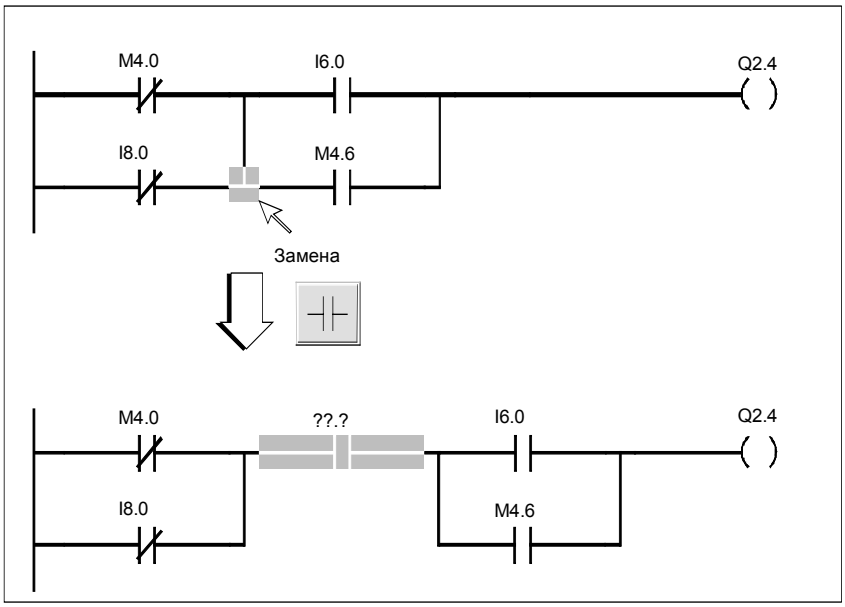
27.8.11 Расщепление узла в сегментах контактного плана

Если в сегменте контактного плана в одной точке закрывается одна параллельная ветвь и открывается другая, то эту точку будем называть узлом. Вы можете расщепить узел следующим образом:

1. С помощью клавиши INSERT переключитесь в режим замены. Текущий режим отображается в строке состояния в правом нижнем углу экрана.
2. Выделите узел в верхней или нижней точке разветвления.
3. Используйте один из следующих методов, чтобы вставить элемент контактного плана:
   - В меню "Insert [Вставка]" выберите команду меню для элемента, например, **Insert > LAD Element > Normally Open Contact [Вставить > Элемент KOP > Нормально разомкнутый контакт]**.
   - На панели инструментов щелкните на кнопке нормально разомкнутого контакта, нормально замкнутого контакта или выходной катушки .
   - Введите нормально разомкнутый контакт, нормально замкнутый контакт или выходную катушку при помощи функциональной клавиши F2, F3 или F7.
   - Выберите команду меню **Insert > Program Elements [Вставить > Элементы программы]**, чтобы открыть диалоговое окно "Program Elements [Элементы программы]", и выберите в каталоге требуемый элемент.

Узел расщепляется, и вставляется элемент контактного плана.

- Если вы снова нажмете INSERT, то переключитесь обратно в режим вставки. Текущий режим отображается в строке состояния в правом нижнем углу экрана.



27.8.12 Создание Т-образных ветвей с катушками в сегментах контактного плана

Внутри сегмента контактного плана вы можете запрограммировать ряд Т-образных ветвей.

Т-образная ветвь открывает параллельную ветвь с катушкой, начинающуюся перед выбранным элементом программы. Вы можете вставлять в новую ветвь дальнейшие элементы контактного плана.

1. Выделите элемент программы, перед которым вы хотите открыть новую Т-образную ветвь.
2. Используйте один из следующих методов, чтобы открыть новую Т- образную ветвь:
   - Выберите команду меню **Insert > LAD Element > T Branch [Вставить > Элемент КОР > Т-образная ветвь]**.
   - Нажмите функциональную клавишу F11.

Вставится Т-образная ветвь с катушкой.

27.8.13 Создание Т-образных ветвей в сегментах контактного плана

Команда меню **Insert > LAD Element > Open Branch [Вставить > Элемент КОР > Открыть ветвь]** открывает параллельную ветвь без катушки, начинающуюся перед выбранным элементом программы. Вы можете вставлять в новую ветвь дальнейшие элементы контактного плана.

1. Выделите элемент программы, перед которым вы хотите открыть новую Т-образную ветвь.
2. Используйте один из следующих методов, чтобы открыть новую Т- образную ветвь:
   - Выберите команду меню **Insert > LAD Element > Open Branch [Вставить > Элемент КОР > Открыть ветвь]**.
   - Нажмите функциональную клавишу F8.
3. Теперь выберите элемент контактного плана, который вы хотите вставить в ветвь.

27.9 Как вводить элементы FUP

27.9.1 Ввод элементов FUP

Действуйте следующим образом :

1. В сегменте выберите точку, после которой вы хотите вставить элемент FUP.
2. Вставьте в сегмент требуемый элемент, используя один из следующих методов:
   - Щелкните на соответствующей кнопке на панели инструментов.
   - Введите блок AND [И] или блок OR [ИЛИ], используя функциональную клавишу F2 или F3.
   - Выберите команду меню **Insert > Program Elements [Вставить > Элементы программы]**, чтобы открыть диалоговое окно "Program Elements [Элементы программы]", и выберите в каталоге требуемый элемент.

Выбранный элемент FUP вставляется, и для представления адресов и параметров используются символы вопросительного знака (??.?).

Примечание

Вы можете также редактировать раздел кода, выбирая существующие элементы FUP и затем выбирая одну из команд меню **Edit > Cut [Редактировать > Вырезать]**, **Edit > Copy [Редактировать > Копировать]** или **Edit > Paste [Редактировать > Вставить]**.

27.9.2 Вставка элементов FUP из каталога

1. Откройте каталог аппаратных средств, используя команду меню **View > Catalog [Вид > Каталог]**.
2. В каталоге дважды щелкните по требуемому семейству. В этом семействе перечислены все компоненты, доступные вам для выбора.
3. Скопируйте компонент в раздел кода FUP, используя буксировку.

Этим способом вы можете очень быстро копировать целые блоки в ваш сегмент и программировать обращения к другим блокам.

Если вы выбираете это семейство...	... то перечисляются эти элементы
Функциональные блоки (FB)	Все FB в программе S7
Функции (FC)	Все FC в программе S7
Системные функциональные блоки (SFB)	Все SFB, доступные в CPU
Системные функции (SFC)	Все SFC, доступные в CPU
Библиотеки	Стандартные библиотеки STEP 7 и библиотеки, созданные вами самостоятельно

27.9.3 Ввод адресов или параметров в элементах FUP

1. Поместите курсор на замещающих символах, щелкнув по ним мышью или применив клавишу TAB.
2. Напечатайте адрес или параметр вместо замещающих символов (прямая или косвенная адресация). Если активизировано отображение выбора символов (команда меню **View > Display > Symbol Selection [Вид > Отобразить > Выбор символов]**), то отображается список существующих символических имен. Символическое имя, начинающееся с введенных символов, выбирается и может быть введено нажатием клавиши RETURN.
3. Нажмите RETURN. Программное обеспечение выполняет проверку синтаксиса.
   - Если синтаксис верен, то адрес форматируется и отображается черным шрифтом, а редактор автоматически открывает следующее текстовое поле, требующее ввода адреса или параметра.
   - Если имеется синтаксическая ошибка, то выход из поля ввода не происходит и в строке состояния отображается сообщение об ошибке. Снова нажмите клавишу RETURN; происходит выход из поля ввода, а неправильный ввод отображается красным курсивным шрифтом.

Примечание

Комбинация символов ">>" на выходе означает, что этот выход должен быть подключен перед сохранением или загрузкой.

27.9.4 Замена элементов FUP

Режим замены позволяет заменять однотипные элементы FUP. Это имеет то преимущество, что вам не нужно снова вводить адреса и параметры. Элемент FUP можно заменить только элементом того же самого типа. Например, вы можете поменять RS-триггер на SR-триггер или таймер на счетчик

Чтобы заменить элемент FUP, действуйте следующим образом:

1. С помощью клавиши INSERT переключитесь в режим замены. Текущий режим отображается в строке состояния в правом нижнем углу экрана.
2. Выделите элемент FUP, который вы хотите заменить.
3. Вставьте в сегмент требуемый элемент одним из следующих способов:
   - Щелкните на соответствующей кнопке на панели инструментов.
   - Введите блок AND [И] или блок OR [ИЛИ], используя функциональную клавишу F2 или F3.
   - Выберите команду меню **Insert > Program Elements [Вставить > Элементы программы]**, чтобы открыть диалоговое окно "Program Elements [Элементы программы]", и выберите в каталоге требуемый элемент. Существующий элемент FUP заменяется новым, который вы выбрали.
4. Если вы снова нажмете INSERT, то переключитесь обратно в режим вставки. Текущий режим отображается в строке состояния в правом нижнем углу экрана

27.9.5 Выделение элементов в сегментах с FUP

В пределах сегмента вы можете выделять следующие области, щелкая по ним один раз мышью:

- элементы FUP, например, блок AND [И] или стандартный блок типа счетчика
- соединительные линии
- адреса
- входные/выходные контакты

В диалоговом окне "Customize [Настройка]" во вкладке "LAD/FBD (KOP/FUP)" вы можете самостоятельно выбрать цвет выделения. Вы открываете это диалоговое окно при помощи команды меню **Options > Customize** [Параметры > Настроить].

Чтобы выделить сегмент, в который вы можете вводить элементы FUP, действуйте следующим образом:

1. Щелкните по заголовку сегмента (например, "Network 1").
2. Затем вы можете вырезать, вставлять или копировать сегмент, который вы выбрали таким образом.

27.9.6 Вставка дополнительных сегментов FUP

Чтобы вставить новый сегмент, действуйте следующим образом:

1. Выберите команду меню **Insert > Network [Вставить > Сегмент]** или щелкните на соответствующей кнопке на панели инструментов. Новый сегмент вставится ниже выбранного сегмента.
2. Если вы вводите большее количество элементов, чем может отображаться на вашем экране, то сегмент на экране сдвигается влево. Используя команды меню **View > Zoom Out/Zoom In/Zoom Factor [Вид > Сжать/Раскрыть/Масштабный коэффициент]**, вы можете регулировать размер изображения так, чтобы получить улучшенный обзор.

27.9.7 Создание Т-образных ветвей в сегментах с FUP

Внутри сегмента с FUP вы можете программировать ряд Т- образных ветвей. Т- образная ветвь открывает параллельную ветвь, начинающуюся перед выбранным двоичным входом. Вы можете вставлять в новую ветвь дальнейшие элементы FUP.

1. Выберите двоичный ввод, перед которым вы хотите открыть новую Т-образную ветвь.
2. Используйте один из следующих методов, чтобы открыть новую Т-образную ветвь:
   - Выберите команду меню **Insert > FBD Element > T Branch [Вставить > Элемент FUP > Т-образная ветвь]**.
   - Нажмите функциональную клавишу F11.
   - Щелкните на соответствующей кнопке на панели инструментов.

27.9.8 Создание соединений в сегментах с FUP

В пределах сегмента с FUP вы можете соединить два логических пути, однако, только один из логических путей может содержать назначение.

Чтобы создать соединение, действуйте следующим образом:

1. Выберите двоичный вход и двоичный выход, которые вы хотите соединить.
2. Используйте один из следующих методов, чтобы соединить двоичные объекты:
   - Выберите команду меню **Insert > FBD Element > Connection [Вставить > Элемент FUP > Соединение]**.
   - Нажмите функциональную клавишу F12.
   - Щелкните на соответствующей кнопке на панели инструментов.

27.9.9 Разъединение и связывание соединений в сегментах с FUP

1. Выделите двоичный вход.
2. Разъедините соединение нажатием клавиши DEL..
3. При необходимости вставьте в точке разъединения новые элементы FUP.
4. Выделите двоичный выход.
5. При нажатой левой кнопке мыши перетащите соединение к требуемому двоичному входу.

В случае необходимости элементы будут переупорядочены графически.

27.10 Как вводить операторы AWL

27.10.1 Ввод операторов AWL

Когда вы создаете новый логический блок, вы можете сразу редактировать первый сегмент. Вы обращаетесь к сегменту, щелкнув мышью на строке сегмента. В пределах отдельных сегментов вы вводите операторы строка за строкой при помощи клавиатуры. Вам доступны для использования все обычные функции редактирования.

1. Откройте текстовое поле для сегмента, щелкнув по свободной области ниже серого поля комментария (или ниже заголовка сегмента, если комментарии выключены).
2. Введите команду, нажмите пробел и введите адрес (прямой или косвенный адрес).
3. Нажмите пробел и введите (необязательный) комментарий, начиная с двойного слеша //.
4. После завершения операторной строки с комментарием или без комментария нажмите клавишу RETURN.

После завершения ввода строки выполняется проверка синтаксиса, и оператор форматируется и отображается. Любые символы нижнего регистра в команде или абсолютном адресе преобразуются в символы верхнего регистра.

Любые обнаруженные синтаксические ошибки отображаются красным курсивом. Вы должны исправить все ошибки прежде, чем сохранить логический блок.

27.10.2 Выделение текстовых областей в операторах AWL

Вы можете в сегменте с AWL выделять текст символ за символом.

1. Поместите курсор на первом символе.
2. Выделите текст, перемещая курсор при нажатой левой кнопке мыши через текст, который вы хотите выделить.

Вы можете выделить ряд операторных строк одновременно, удерживая левую кнопку мыши нажатой и перемещая курсор по вертикали. Альтернативно, вы можете выбирать текстовые области, используя клавиши со стрелками «вправо», «влево», «вверх» и «вниз» при нажатой клавише SHIFT.

Примечание

Вы можете самостоятельно выбрать цвет выделения. Чтобы сделать это, откройте вкладку "LAD/FBD (KOP/FUP)" при помощи команды меню **Options > Customize [Параметры > Настроить]** выберите цвет для выделенного элемента.

27.10.3 Вставка дополнительных сегментов в AWL

Чтобы создать новый сегмент, действуйте следующим образом:

1. Выберите команду меню **Insert > Network [Вставить > Сегмент]** или щелкните на соответствующей кнопке на панели инструментов. Новый сегмент вставится ниже выделенного сегмента.
2. Чтобы выделить сегмент, щелкните по заголовку сегмента (например, "Network 1"). Затем вы можете вырезать, вставлять или копировать сегмент, который выбрали таким образом.

27.10.4 Ввод комментариев в сегменты AWL

В случае представлении языка программирования в форме списка операторов вы можете вводить комментарий для каждого оператора.

1. После каждого адреса или символического имени нажмите клавишу пробела.
2. Начните комментарий к оператору двумя слешами (//).
3. Завершите комментарий нажатием RETURN.

27.11 Этапы сохранения кодовых блоков

27.11.1 Сохранение логических блоков

Чтобы ввести в базу данных устройства программирования вновь созданные блоки или изменения в разделе кода или в таблицах описаний логических блоков, вам нужно сохранить соответствующий блок. Тогда данные записываются на жесткий диск устройства программирования.

Чтобы сохранить блоки на жестком диске устройства программирования:

1. Активизируйте рабочее окно блока, который вы хотите сохранить.
2. Выберите одну из следующих команд меню:
   - **File > Save [Файл > Сохранить]** сохраняет блок под прежним именем
   - **File > Save As [Файл > Сохранить как...]** сохраняет блок под другой программой пользователя S7 или под другим именем. Введите новый путь или новое имя блока в появляющемся диалоговом окне.

В обоих случаях блок сохраняется только при условии, что его синтаксис не содержит ошибок. Синтаксические ошибки идентифицируются сразу, когда блок создается, и отображаются красным шрифтом. Эти ошибки нужно исправить прежде, чем блок можно будет сохранить.

Примечание

- В SIMATIC Manager вы можете также сохранять блоки или исходные файлы ниже других проектов или библиотек (например, с помощью буксировки).
- В SIMATIC Manager на плате памяти вы можете сохранять только блоки или полные программы пользователя.
- Если при сохранении или компиляции больших блоков возникают проблемы, то вы должны реорганизовать проект. Чтобы сделать это, используйте в SIMATIC Manager команду меню **File > Reorganize [Файл > Реорганизовать]**. Затем снова попытайтесь сохранить или компилировать.

27.12 Как вводить и сохранять блоки данных

27.12.1 Ввод структур данных совместно используемых блоков данных

Если вы открываете блок данных, который не назначен определяемому пользователем типу данных или функциональному блоку, то вы можете определить его структуру в представлении описаний блока данных. В блоках данных, не используемых совместно, представление описаний не может изменяться.

1. Откройте совместно используемый блок данных, то есть блок, который не связан с UDT или FB.
2. Отобразите представление описаний блока данных, если это представление еще не установлено.
3. Определите структуру, заполняя отображаемую таблицу в соответствии с информацией, приведенной ниже.

В блоках данных, не используемых совместно, представление описаний не может изменяться

Столбец	Объяснение
Address [Адрес]	Отображает адрес, который STEP 7 автоматически назначает переменной, когда вы заканчиваете ввод описания.
Name [Имя]	Введите символическое имя, которое вы должны назначить здесь каждой переменной.
Type [Тип]	Введите тип данных, который вы хотите назначить переменной (BOOL, INT, WORD, ARRAY и т. д.). Переменные могут иметь элементарные типы данных, составные данных или типы данных, определяемые пользователем.
Initial Value [Начальное значение]	Здесь вы можете вводить начальное значение, если вы не хотите, чтобы программное обеспечение использовало значение по умолчанию для введенного типа данных. Все значения должны быть совместимы с типом данных. Когда вы сохраняете блок впервые, начальное значение используется как фактическое значение, если вы явно не определили фактические значения переменных.
Comment [Комментарий]	Ввод необязательного комментария в этом поле помогает документировать переменную. Комментарий может иметь длину до 80 символов.

27.12.2 Ввод и отображение структуры данных для блоков данных, ссылающихся на FB (экземплярные DB)

Ввод

Когда вы связываете блок данных с функциональным блоком (экземплярный DB), описание переменных функционального блока определяет структуру этого блока данных. Любые изменения могут производиться только в соответствующем функциональном блоке.

1. Откройте соответствующий функциональный блок (FB).
2. Отредактируйте таблицу описания переменных функционального блока.
3. Снова создайте экземплярный блок данных.

Отображение

В представлении описаний экземплярного блока данных вы можете отображать то, как были описаны переменные в функциональном блоке.

1. Откройте блок данных.
2. Отобразите представление описаний блока данных, если это представление еще не установлено.
3. Дополнительную информацию об отображаемой таблице смотрите ниже.

В блоках данных, не используемых совместно, представление описаний не может изменяться

Столбец	Объяснение
Address [Адрес]	Отображает адрес, который STEP 7 автоматически назначает переменной.
Declaration [описание]	Этот столбец показывает, как описаны переменные в описании переменных функционального блока: • Входной параметр (IN) • Выходной параметр (OUT) • Проходной параметр (IN_OUT) • Статические данные (STAT) Описанные временные локальные данные функционального блока не находятся в экземплярном блоке данных.
Name [Имя]	Символическое имя, назначенное в описании переменных функционального блока.
Type [Тип]	Отображает тип данных, назначенный в описании переменных функционального блока. Переменные могут иметь элементарные типы данных, составные типы данных или типы данных, определяемые пользователем. Если внутри функционального блока вызываются дополнительные функциональные блоки, для вызова которых были описаны статические переменные, то функциональный блок или системный функциональный блок (SFB) также может быть задан здесь как тип данных.
Initial Value [Начальное значение]	Начальное значение, которое вы ввели для переменной в описании переменных функционального блока, если вы не хотите, чтобы программное обеспечение использовало значение по умолчанию. Когда вы сохраняете блок данных впервые, начальное значение используется как фактическое значение, если вы явно не определили фактические значения переменных.
Comment [Комментарий]	Комментарий, введенный в описание переменных функционального блока, чтобы документировать элемент данных. Вы не можете редактировать это поле.

Примечание

В блоках данных, назначенных функциональному блоку, вы можете редактировать только фактические значения переменных. Чтобы вводить фактические значения переменных, вам нужно быть в представлении данных блоков данных.

27.12.3 Ввод структуры данных для типов данных, определяемых пользователем (UDT)

1. Откройте определяемый пользователем тип данных (UDT).
2. Отобразите представление описаний, если это представление еще не установлено.
3. На основе информации, приведенной ниже в таблице, определите структуру UDT, задавая последовательность переменных, их типы данных и, при необходимости, начальные значения.
4. Завершайте ввод переменной выходом из строки при помощи клавиши TAB или RETURN.

Столбец	Объяснение
Address [Адрес]	Отображает адрес, который STEP 7 автоматически назначает переменной, когда вы завершаете ввод описания.
Name [Имя]	Введите символическое имя, которое вы должны назначить здесь каждой переменной.
Type [Тип]	Введите тип данных, который вы хотите назначить переменной (BOOL, INT, WORD, ARRAY и т. д.). Переменные могут иметь элементарные типы данных, составные типы данных или свои собственные определяемые пользователем типы данных.
Initial Value [Начальное значение]	Здесь вы можете вводить начальное значение, если вы не хотите, чтобы программное обеспечение использовало значение по умолчанию для введенного типа данных. Все значения должны быть совместимы с типом данных. Когда вы впервые сохраняете экземпляр определяемого пользователем типа данных (или переменную, или блок данных), начальное значение используется как фактическое значение, если вы явно не определили фактические значения переменных.
Comment [Комментарий]	Ввод комментария в этом поле помогает документировать переменные. Комментарий может иметь длину до 80 символов

27.12.4 Ввод и отображение структуры блоков данных, ссылающихся на UDT

Ввод

Когда вы ставите в соответствие блок данных определяемому пользователем типу данных, структура данных определяемого пользователем типа данных определяет структуру блока данных. Любые изменения могут производиться только в соответствующем определяемом пользователем типе данных.

1. Откройте определяемый пользователем тип данных (UDT).
2. Отредактируйте структуру определяемого пользователем типа данных.
3. Снова создайте блок данных.

Отображение

В представлении описаний блока данных вы можете только отображать то, как были описаны переменные в определяемом пользователем типе данных.

1. Откройте блок данных.
2. Отобразите представление описаний блока данных, если это представление еще не установлено.
3. Дополнительную информацию об отображаемой таблице смотрите ниже.

Представление описаний не может изменяться. Любые изменения могут производиться только в соответствующем определяемом пользователем типе данных.

Столбец	Объяснение
Address [Адрес]	Отображает адрес, который STEP 7 автоматически назначает переменной.
Name [Имя]	Символическое имя, назначенное в описании переменных типа данных пользователя.
Type [Тип]	Отображает типы данных, назначенные в описании переменных определяемого пользователем типа данных. Переменные могут иметь элементарные типы данных, составные типы данных или типы данных, определяемые пользователем.
Initial Value [Начальное значение]	Начальное значение, которое вы ввели для переменной в определяемом пользователем типе данных, если вы не хотите, чтобы программное обеспечение использовало значение по умолчанию. Когда вы сохраняете блок данных впервые, начальное значение используется как фактическое значение, если вы явно не определили фактические значения переменных.
Comment [Комментарий]	Комментарий, введенный в описание переменных определяемого пользователем типа данных, чтобы документировать элемент данных.

Примечание

В блоках данных, поставленных в соответствие определяемому пользователем типу данных, можно редактировать только фактические значения переменных. Чтобы вводить фактические значения переменных, вы должны находиться в представлении данных блоков данных.

27.12.5 Редактирование значений данных в представлении данных

Редактирование фактических значений возможно только в представлении данных блоков данных.

1. В случае необходимости переключитесь на отображение таблицы в представлении данных при помощи команды меню **View > Data View [Вид > Представление данных]**.
2. В полях столбца "Actual Value [Фактическое значение]" введите требуемые фактические значения для элементов данных. Фактические значения должны быть совместимы с типом данных элементов данных.

Любой неправильный ввод (например, введенное фактическое значение несовместимо с типом данных), сделанный во время редактирования, обнаруживается сразу и изображается красным шрифтом. Эти ошибки должны быть исправлены перед сохранением блока данных.

Примечание

Любые изменения значений данных сохраняются только тогда, когда сохраняется блок данных.

27.12.6 Возврат данных к их начальным значениям

Возврат значений данных возможен только в представлении данных блоков данных.

1. В случае необходимости переключитесь на отображение таблицы в представлении данных при помощи команды меню **View > Data View [Вид > Представление данных]**.
2. Чтобы сделать это, выберите команду меню **Edit > Initialize Data Block [Редактировать > Инициализировать блок данных]**.

Всем переменным снова присваиваются предназначенные им начальные значения, то есть фактические значения всех переменных заменяются соответствующими им начальными значениями.

Примечание

Любые изменения значений данных сохраняются только тогда, когда сохраняется блок данных

27.12.7 Сохранение блоков данных

Чтобы ввести в базу данных устройства программирования вновь созданные блоки или измененные значения данных блоков данных, вы должны сохранить соответствующий блок. Тогда данные записываются на жесткий диск устройства программирования.

Чтобы сохранить блоки на жестком диске устройства программирования:

1. Активизируйте рабочее окно блока, который вы хотите сохранить.
2. Выберите одну из следующих команд меню:
   - **File > Save [Файл > Сохранить]** сохраняет блок под прежним именем.
   - **File > Save As** [Файл > Сохранить как...] сохраняет блок под другой программой пользователя S7 или под другим именем. Введите новый путь или новое имя блока в появляющемся диалоговом окне. Для блоков данных вы не можете использовать имя DB0, потому что этот номер зарезервирован для системы.

В обоих случаях блок сохраняется только при условии, что его синтаксис не содержит ошибок. Синтаксические ошибки идентифицируются сразу, когда блок создается, и отображаются красным шрифтом. Эти ошибки нужно исправить прежде, чем блок можно будет сохранить.

Примечание

- Вы можете также сохранять блоки или исходные файлы ниже других проектов или библиотек в SIMATIC Manager (например, буксировкой).
- Вы можете сохранять в SIMATIC Manager на плате памяти только блоки или полные программы пользователя.
- Если при сохранении или компиляции больших блоков возникают проблемы, то вы должны реорганизовать проект. Чтобы сделать это, используйте в SIMATIC Manager команду меню **File > Reorganize [Файл > Реорганизовать]**. Затем снова попытайтесь сохранить или компилировать.

27.13 Как создавать исходные файлы на AWL

27.13.1 Создание исходных файлов AWL

Исходный файл должен создаваться в папке исходных файлов ниже программы S7. вы можете создавать исходные файлы в SIMATIC Manager или в окне редактора.

Создание исходных файлов в SIMATIC Manager

1. Откройте соответствующую папку "Source Files [Исходные файлы]", дважды щелкнув по ней.
2. Чтобы вставить исходный файл на AWL, выберите команду меню **Insert > S7 Software > STL Source File [Вставить > Программное обеспечение S7 > Исходный файл на AWL]**.

Создание исходных файлов в окно редактора

1. Выберите команду меню **File > New [Файл > Новый]**.
2. В диалоговом окне выберите папку исходных файлов той же самой программы S7, которая содержит программу пользователя с блоками.
3. Введите имя нового исходного файла.
4. Подтвердите посредством "OK".

Создается исходный файл под именем, которое вы ввели, и отображается в окне редактирования.

27.13.2 Редактирование исходных файлов S7

Язык программирования и редактор, с помощью которого редактируется исходный файл, можно установить в свойствах объекта для исходного файла. Это гарантирует, что при открытии исходного файла для редактирования запускаются правильный редактор и правильный язык программирования. Стандартный пакет STEP 7 поддерживает программирование в исходных файлах на AWL.

Доступны также другие языки программирования в виде дополнительных пакетов. Вы можете выбирать команду меню для вставки исходного файла только тогда, когда соответствующее дополнительное программное обеспечение загружена в ваш компьютер.

Чтобы редактировать исходный файл S7, действуйте следующим образом:

1. Откройте соответствующую папку "Source Files [Исходные файлы]", дважды щелкнув по ней.
2. Запустите требуемый редактор следующим образом:
   - Дважды щелкните по требуемому исходному файлу в правой половине окна.
   - Выделите требуемый исходный файл в правой половине окна и выберите команду меню **Edit > Open Object [Редактировать > Открыть объект]**.

27.13.3 Вставка шаблонов блоков в исходные файлы на AWL

Для программирования в исходных файлах на AWL имеются в распоряжении шаблоны для организационных блоков (OB), функциональных блоков (FB), функций (FC), блоков данных (DB), экземплярных блоков данных, блоков данных с соответствующими типами данных, определяемыми пользователем, и определяемых пользователем типов данных (UDT). Шаблоны блоков облегчают ввод блоков в исходный файл и соблюдение указаний по синтаксису и структуре.

Чтобы вставить шаблон блока, действуйте следующим образом:

1. Активизируйте окно исходного файла, в который вы хотите вставить шаблон блока.
2. Поместите в файле курсор в точке, после которой вы хотите вставить шаблон блока.
3. Выберите одну из команд меню **Insert > Block Template > OB/FB/FC/DB/Instance DB/DB Referencing UDT/UDT [Вставить > Шаблон блока > OB/FB/FC/DB/Экземплярный DB/DB, ссылающийся на UDT/UDT]**.

Шаблон блока вставляется в файл после позиции курсора.

27.13.4 Вставка содержимого других исходных файлов на AWL

Вы можете вставлять содержимое других исходных файлов в ваш исходный файл на AWL.

Действуйте следующим образом:

1. Активизируйте окно исходного файла, в который вы хотите вставить содержимое другого исходного файла.
2. Поместите в файле курсор в точке, после которой вы хотите вставить исходный файл.
3. Выберите команду меню **Insert > Object > File** [Вставить > Объект > Файл].
4. В появляющемся диалоговом окне выберите требуемый исходный файл.

Содержимое выбранного исходного файла вставляется после позиции курсора. Символы перевода строки (возврата каретки) сохраняются.

27.13.5 Вставка исходного кода из существующих блоков в исходные файлы на AWL

Вы можете вставлять в ваш исходный файл на AWL исходный код из других блоков, созданных в виде контактного плана, функционального плана или списка операторов. Это возможно для организационных блоков (OB), функциональных блоков (FB), функций (FC), блоков данных (DB) и определяемых пользователем типов данных (UDT).

Действуйте следующим образом:

1. Активизируйте окно исходного файла, в который вы хотите вставить блок.
2. Поместите в файле курсор в точке, после которой вы хотите вставить исходный код из блока.
3. Выберите команду меню **Insert > Object > Block [Вставить > Объект > Блок]**.
4. В появляющемся диалоговом окне выберите требуемый блок.

Из блока генерируется эквивалентный исходный файл. Содержимое этого исходного файла вставляется после позиции курсора.

27.13.6 Вставка внешних исходных файлов

Вы можете создать и отредактировать исходный файл с помощью какого-либо редактора ASCII, затем импортировать его в проект и компилировать его в отдельные блоки, использующие это приложение. Чтобы сделать это, вы должны импортировать исходные файлы в папку "Source Files [Исходные файлы]" программы S7, в чьей пользовательской программе S7 должны сохраняться блоки, созданные во время трансляции.

Чтобы вставить внешний исходный файл, действуйте следующим образом:

1. Выберите папку исходных файлов программы S7, в которую должны импортироваться внешние исходные файлы.
2. Выберите команду меню **Insert > External Source File [Вставить > Внешний исходный файл]**.
3. В появляющемся диалоговом окне введите исходный файл, который вы хотите импортировать.

Имя импортируемого вами исходного файла должно иметь подходящее расширение файла. STEP 7 использует расширение файла для того, чтобы определить тип исходного файла. Это означает, например, что STEP 7 создает исходный файл на AWL, когда он импортирует файл с расширением **.AWL**. Допустимые расширения файлов перечисляются в диалоговом окне под "File Type [Тип файла]".

Примечание

Вы можете также использовать команду меню **Insert > External Source File** [Вставить > Внешний исходный файл], чтобы импортировать исходные файлы, которые вы создали с помощью STEP 7 версии 1.

27.13.7 Генерирование исходных файлов на AWL из блоков

Вы можете генерировать из существующих блоков исходный файл на AWL, который вы можете редактировать с помощью любого текстового редактора. Генерируемый исходный файл создается в папке исходных файлов той же самой пользовательской программы S7, из которой были выбраны блоки.

Чтобы генерировать исходный файл из блока, действуйте следующим образом:

1. Выберите команду меню File > Generate Source File **[Файл > Генерировать исходный файл]**.
2. В диалоговом окне выберите папку исходного файла, в которой вы хотите создать новый исходный файл.
3. В текстовом поле введите имя исходного файла.
4. В диалоговом окне "Select STEP Blocks [Выбор блоков STEP]" выберите блок(и), которые вы хотите генерировать как заданный исходный файл. Выбранные блоки отображаются в правом списковом окне.
5. Подтвердите посредством "OK".

Из выбранных блоков создается один непрерывный исходный файл на AWL, который отображается в окне редактирования.

27.13.8 Импортирование исходных файлов

Чтобы импортировать исходный файл из какого-либо каталога в проект:

1. В SIMATIC Manager выберите папку исходного файла, в которую вы хотите импортировать исходный файл.
2. Выберите команду меню **Insert > External Source File** [Вставить > Внешний исходный файл].
3. В отображающемся диалоговом окне выберите целевой каталог и импортируемый исходный файл.
4. Щелкните на кнопке "Open [Открыть]".

27.13.9 Экспортирование исходных файлов

Чтобы экспортировать исходный файл из проекта в какой-либо целевой каталог:

1. В папке исходных файлов выберите исходный файл.
2. Выберите команду меню **Edit > Export Source File in the SIMATIC Manager [Редактировать > Экспортировать исходный файл в SIMATIC Manager]**.
3. В отображающемся диалоговом окне введите целевой каталог и имя файла.
4. Щелкните на кнопке "Save [Сохранить]".

Примечание

Если имя объекта не имеет расширения файла, то к имени файла добавляется расширение файла, выведенное из типа файла. Например, исходный файл на AWL "**prog**" экспортируется в файл "**prog.awl**".

Если имя объекта уже имеет допустимое расширение файла, то оно сохраняется и не изменяется. Например, исходный файл на AWL "**prog.awl**" экспортируется в файл "**prog.awl**".

Если имя объекта имеет недопустимое расширение файла (то есть точка содержится в имени), то расширение файла не добавляется.

Вы найдете список допустимых расширений файла в диалоговом окне "Export source file [Экспорт исходного файла]" под "File type [Тип файла]".

27.14 Сохранение и компиляция исходных файлов на AWL и выполнение проверки на непротиворечивость

27.14.1 Сохранение исходных файлов AWL

Вы можете сохранить исходный файл на AWL в любое время в его текущем состоянии. Программа не компилируется, и никакая проверка синтаксиса не выполняется, то есть любые ошибки также сохраняются.

Синтаксические ошибки обнаруживаются и сообщаются только тогда, когда исходный файл компилируется, или после проверки на непротиворечивость.

Чтобы сохранить исходный файл под прежним именем:

1. Активизируйте окно для исходного файла, который вы хотите сохранить.
2. Выберите команду меню **File > Save [Файл > Сохранить]**.

Чтобы сохранить исходный файл под новым именем/в другом проекте:

3. Активизируйте окно для исходного файла, который вы хотите сохранить.
4. Выберите команду меню **File > Save As [Файл > Сохранить как...]**.
5. В диалоговом окне выберите папку исходных файлов, в которой вы хотите сохранить исходный файл, и введите его новое имя.

27.14.2 Проверка на непротиворечивость в исходных файлах на AWL

При помощи команды меню **File > Consistency Check [Файл > Проверка на непротиворечивость]** вы можете выводить на экран любые синтаксические ошибки в исходном файле на AWL. В отличие от компиляции, блоки не генерируются.

Когда проверка на непротиворечивость заканчивается, отображается диалоговое окно, показывающее вам общее количество найденных ошибок

Любые найденные ошибки перечисляются индивидуально в нижней части окна со ссылкой на строку. Исправьте эти ошибки перед компиляцией исходного файла так, чтобы все блоки могли быть созданы.

27.14.3 Поиск ошибок в исходных файлах на AWL

Активное окно для исходных файлов разбито на два. В нижней половине перечисляются следующие ошибки:

- Ошибки, найденные после запуска компиляции, выполняемого при помощи команды меню **File > Compile [Файл > Компилировать]**.
- Ошибки, найденные после запуска проверки на непротиворечивость с помощью команды меню **File > Consistency Check [Файл > Проверка на непротиворечивость]**.

Чтобы найти местоположение ошибки в исходном файле, установите курсор на соответствующее сообщение об ошибке в нижней части окна. В верхней части окна автоматически высветится текстовая строка, содержащая ошибку. Сообщение об ошибке появляется также в строке состояния.

27.14.4 Компиляция исходных файлов AWL

Требования

Чтобы иметь возможность компилировать созданную вами в исходном файле программу в блоки, нужно выполнить следующие требования:

- Компилироваться могут только исходные файлы, хранимые в папке "Source Files [Исходные файлы]" ниже программы S7.
- Как и папка "Source Files [Исходные файлы]", ниже программы S7 должна лежать папка "Blocks [блоки]", в которой могут сохраняться блоки, создаваемые во время компиляции. Блоки, запрограммированные в исходном файле, создаются только тогда, когда исходный файл был откомпилирован без ошибок. Если в исходном файле запрограммирован ряд блоков, то создаются только те из них, которые не содержат ошибок. Тогда вы можете открывать эти блоки, редактировать их, загружать их в CPU и отлаживать их индивидуально.

Процедура в редакторе

1. Откройте исходный файл, который вы хотите компилировать. Исходный файл должен находиться в папке исходных файлов программы S7, в чьей пользовательской программе S7 должны храниться откомпилированные блоки.
2. Выберите команду меню **View > Display > Symbolic Representation [Вид > Отобразить > Символическое представление]**, чтобы впоследствии в откомпилированных блоках могли отображаться символические имена.
3. Выберите команду меню **File > Compile** [Файл > Компилировать].
4. Отображается диалоговое окно "Compiler Report [Отчет компилятора]", показывающее число откомпилированных строк и число найденных синтаксических ошибок.

Блоки, заданные для файла, создаются только при условии, что исходный файл был скомпилирован без ошибок. Если в исходном файле запрограммирован ряд блоков, то создаются только те из них, которые не содержат ошибок. Предупреждения об ошибках не препятствуют созданию блоков.

Любые синтаксические ошибки, обнаруженные во время компиляции, показываются в нижней части рабочего окна и должны быть исправлены прежде, чем могут быть созданы соответствующие блоки.

Процедура в SIMATIC Manager

1. Откройте соответствующую папку "Source Files [Исходные файлы]", дважды щелкнув по ней.
2. Выберите один или большее количество исходных файлов, которые вы хотите компилировать. Для закрытой папки исходных файлов вы не можете запустить выполнение компиляции так, чтобы компилировать все находящиеся в ней исходные файлы.
3. Выберите команду меню **File > Compile [Файл > Компилировать]**, чтобы запустить компиляцию. Для исходного файла, который вы выбрали, вызывается подходящий компилятор. Затем успешно скомпилированные блоки сохраняются в папке блоков ниже программы S7.
   Любые синтаксические ошибки, обнаруженные во время компиляции, отображаются в диалоговом окне и должны быть исправлены, чтобы блоки, где обнаружились ошибки, могли быть созданы.

27.15 Как работать со справочными данными

27.15.1 Поиск справочных данных

При помощи команды меню **Edit > Find** [Редактировать > Найти], вы можете искать определенные текстовые строки в активном окне. Искомая строка может отыскиваться от позиции курсора вверх или вниз, либо во всем документе.

Обратите внимание на то, что это функция чисто текстового поиска, где вы должны вводить искомую строку с точностью до символа.

27.15.2 Сортировка справочных данных

Вы можете сортировать записи списка, щелкая по заголовку столбца: столбцы, содержащие буквы (например, символические имена), сортируются в алфавитном порядке, столбцы, содержащие числа, сортируются в порядке возрастания.

Если один и тот же адрес появляется более одного раза, то может оказаться полезной работа с дальнейшими критериями сортировки.

Например, если вы хотите сортировать список перекрестных ссылок по адресам, а в пределах одинаковых адресов сортировать по блокам, то действуйте следующим образом:

1. Сначала отсортируйте по блокам, щелкнув по заголовку столбца "Block [Блок]".
2. Затем щелкните по заголовку столбца "Address [Адрес]".

Альтернатива:

1. Выберите команду меню **View > Sort** [Вид > Сортировать].
2. В появляющемся диалоговом окне выберите критерии сортировки.
3. Щелкните на кнопке "OK".

27.15.3 Фильтрация справочных данных

В случае больших программ вы можете значительно улучшить скорость отображения справочных данных, используя команду фильтрации для ограничения диапазона отображаемых данных. Вам рекомендуется определять фильтр настолько точно, насколько возможно, задавая только то, что вы действительно хотите отобразить.

При стандартной настройке фильтра по умолчанию данные отображаются сначала в виде списка перекрестных ссылок "cross-reference list". Вы можете изменять отображаемое представление справочных данных определенным образом для каждого рабочего окна, используя команду меню **View > Filter [Вид > Фильтр]**, то есть Вы можете приспосабливать содержимое списков к вашим требованиям.

В представлении "addresses without symbols [адреса без символов]" фильтр невозможен.

Если вы уже открыли рабочее окно, содержащее справочные данные, то действуйте следующим образом:

1. Выберите команду меню View > Filter [Вид > Фильтр].
2. В соответствующих вкладках представлений в диалоговом окне "Filter [Фильтр]" выберите требуемые параметры настройки, например, скрыть/ показать (hide/show) заданные столбцы. Настройка по умолчанию действует для всех отображаемых столбцов.
3. Если вы хотите принять эти параметры настройки как умолчание, то нужно активизировать триггерную кнопку "Save as standard [Сохранить как стандарт]". Вы можете активизировать/деактивировать эту опцию, щелкая на этой триггерной кнопке.
4. Подтвердите ваши параметры настройки посредством "OK" или нажатием RETURN.

Примечания к фильтрации особых представлений справочных данных

Представление	Фильтр
Структура программы	Вы можете определить следующее: отображается ли в скобках потребность локальных данных в памяти (в байтах) в расчете на путь рядом с последним блоком в каждом пути программы и отображается ли максимальная потребность локальных данных в начале каждой структуры программы.
Назначение	Вы можете выбирать области памяти, которые хотите отображать в списках назначений. Для каждой области памяти (входы, выходы, меркеры, таймеры, счетчики) вы можете определить диапазон адресов, которым вы хотите ограничить отображение.

27.15.4 Изменение представления справочных данных

Действуйте следующим образом:

1. Выделите программу пользователя, для которой вы хотите отобразить список адресов без символов.
2. В окне "Display Reference Data [Отобразить справочные данные]", выберите команду меню для отображения нужных вам справочных данных (**View > ..** [Вид > …]).

27.15.5 Переход из списка перекрестных ссылок в соответствующую точку программы

Чтобы перейти из списка перекрестных ссылок в соответствующую часть программы:

- Дважды щелкните левой кнопкой мыши по адресу.

Альтернативная процедура:

1. Выберите адрес в списке перекрестных ссылок.
2. Щелкните правой кнопкой мыши, чтобы открыть контекстно-зависимое меню.
3. Выберите команду меню ”**Go To Location** [Перейти к местоположению]

Эта команда контекстно-зависимого меню доступна также в строке меню:

Edit > Go To > Location [Редактировать > Перейти > Местоположение].

27.15.6 Переход из структуры программы в определенное место программы

Чтобы перейти из структуры программы в соответствующую часть программы:

1. Выберите блок в окне "Program Structure [Структура программы]".
2. Щелкните правой кнопкой мыши. Появляется контекстно-зависимое меню.
3. Выберите команду меню ”**Go To Block [Перейти в блок]**, чтобы открыть блок непосредственно, или выберите команду меню ”**Go To Location [Перейти к местоположению]**, чтобы открыть вызывающий блок, и расположите курсор на вызове выбранного блока.

Команду меню ”**Go To Location [Перейти к местоположению]** можно выбрать только тогда, когда для выбранного блока имеется вызывающий блок (на более высоком уровне вложенности).

Эти команды контекстно-зависимого меню доступны также в строке меню:

Edit > Go To > Block [Редактировать > Перейти > Блок] и

Edit > Go To > Location [Редактировать > Перейти > Местоположение].

Вы можете также выбирать эти команды, выделяя блок и щелкая правой кнопкой мыши, чтобы отобразить всплывающее меню.

27.15.7 Открытие рабочих окон для справочных данных, которые уже отображаются

Действуйте следующим образом:

1. Активизируйте рабочее окно программы пользователя S7, для которой вы хотите обновить отображение.
2. В окне "Displaying Reference Data [Отображение справочных данных]" выберите команду меню **Window > New Window [Окно > Новое окно]**.
3. Если вы не отменили выбор диалогового окна "Customize [Настройка]", то можете выбрать, какое представление должно отображать новое окно. В противном случае, выберите соответствующую команду в меню **View [Вид]**.

27.15.8 Открытие рабочих окон для справочных данных, которые еще не отображаются

Действуйте следующим образом:

1. В окне "Displaying Reference Data [Отображение справочных данных]" выберите команду меню **Reference Data > Open [Справочные данные > Открыть]**.
2. В диалоговом окне "Open [Открыть]", выберите программу пользователя S7, для которой вы хотите отобразить список справочных данных.
3. Если вы не отменили выбор диалогового окна "Settings [Параметры настройки]", то можете выбрать, какое представление должно отображать новое окно. В противном случае, выберите соответствующую команду в меню **View [Вид]**.

27.15.9 Отображение перекрывающегося доступа

Чтобы отобразить перекрестные ссылки для адресов, адресные области которых перекрываются, действуйте следующим образом:

1. Выберите адрес в списке перекрестных ссылок справочных данных.
2. Нажмите правую кнопку мыши и выберите во всплывающем меню команду меню **Cross References for Address [Перекрестные ссылки для адреса]**.

Тогда в другом окне отображаются перекрестные ссылки адресов, адресные области которых перекрываются с выбранным адресом.

В редакторе LAD/FBD/STL (KOP/FUP/AWL) действуйте следующим образом:

1. Выделите адрес в окне редактора.
2. Выберите команду меню **Edit > Go To > Location [Редактировать > Перейти > К местоположению]**.
3. Выберите в диалоговом окне "Go To Location [Перейти к местоположению]" опцию "Overlapping accesss to memory areas [Перекрывающийся доступ к областям памяти]".

27.15.9.1 Автоматическое обновление справочных данных при компиляции

Чтобы обновлять справочные данные каждый раз, когда вы компилируете блок, действуйте следующим образом:

1. Выберите в окне ”LAD/STL/FBD (KOP/AWL/FUP)" команду меню **Options > Customize [Параметры > Настроить]**.
2. Выберите в диалоговом окне вкладку ”Create Block [Создать блок]".
3. Выберите опцию ”Generate Reference Data [Генерировать справочные данные]" и подтвердите ваш ввод посредством ”OK".

Тогда при компиляции исходного файла или сохранении блока, созданного в режиме пошагового редактирования, автоматически генерируются справочные данные.

27.15.9.2 Смена заданного по умолчанию представления справочных данных

Справочные данные отображаются в окне в заданном по умолчанию представлении.

Чтобы сменить представление по умолчанию, действуйте следующим образом:

1. Выберите в SIMATIC Manager команду меню Options > Reference Data > Display [Параметры > Справочные данные > Отобразить].
2. Выберите "First View to Open **[Первое представление при открытии]**" в открывающемся диалоговом окне "Customize [Настройка]".
3. Подтвердите ваш ввод посредством ”OK". Открывается диалоговое окно для установленного представления.

27.16 Как проектировать сообщения

27.16.1 Редактирование сообщений, связанных с блоками

27.16.1.1 Редактирование сообщений, связанных с блоками

1. Выберите блок сообщений через его формальный параметр однократным щелчком в подробном представлении: правая половина администратора сообщений.

   Результат: Отображается секция с вкладками для стандартного сообщения.

2. Введите требуемые атрибуты и тексты во вкладках "Attributes [Атрибуты]" и "Text [Текст]".

   Результат: Вы создали стандартное сообщение, которое может отображаться на всех устройствах отображения.

3. При помощи кнопки "New Device [Новое устройство]" добавьте новое устройство отображения типа "ProTool" (Opx) или "WinCC". Для выбора доступны только те устройства отображения, на которые могут выводиться спроектированные сообщения.

   Результат: Добавляется и выбирается новое устройство, и отображается соответствующая секция с вкладками.

4. Во вкладках, относящихся к конкретным устройствам отображения, введите атрибуты и тексты для сообщения, зависящих от устройства отображения.

   Результат: Вы создали вариант сообщения, который используется в качестве сообщения только для выбранного устройства отображения.

Если вы хотите редактировать другие варианты сообщения для существующих устройств отображения:

- Выберите и откройте блок сообщений в подробном представлении, дважды щелкнув по нему:

  Результат: Автоматически выбирается первое устройство отображения, и вы теперь можете редактировать для него варианты сообщения, зависящие от устройства отображения.

27.16.1.2 Вставка нового устройства отображения

1. Выберите стандартное сообщение. Для этого выберите в администраторе сообщений одноканальный блок сообщений, субномер многоканального блока сообщений или сообщение, связанное с символом.
2. Щелкните на кнопке ”New Device” [Новое устройство]" или выполните соответствующую команду контекстного меню, получаемого посредством щелчка правой кнопки мыши.

   Результат: Появляется диалоговое окно ”Add Display Device” [Добавить устройство отображения]".
3. В диалоговом окне выберите устройство отображения, введите для него символическое имя и затем выйдите, щелкнув на кнопке "OK".

   Результат: Добавляется и выделяется устройство отображения. Теперь подробное представление и секция с вкладками устанавливаются так, чтобы вы могли создать тип сообщения, зависимый от устройства отображения.

27.16.1.3 Удаление устройства отображения

1. В подробном представлении администратора сообщений выберите устройство отображения, которое вы хотите удалить.
2. Щелкните на кнопке ”Delete [Удалить]” или выполните соответствующую команду контекстного меню, получаемого посредством щелчка правой кнопки мыши.
3. В диалоговом окне щелчком на "Yes [Да]" подтвердите, что вы хотите удалить устройство отображения.

   Результат: Удаляются устройство отображения и зависящий от устройства отображения вариант сообщения, который вы создавали для этого устройства.

27.16.1.4 Просмотр свойств устройства отображения

1. Выберите стандартное сообщение. Для этого выберите в администраторе сообщений одноканальный блок сообщений или субномер многоканального блока сообщений.
2. Щелкните на кнопке ”New Device” [Новое устройство]" или выполните соответствующую команду контекстного меню, получаемого посредством щелчка правой кнопки мыши

   Результат: Появляется диалоговое окно ”Add Display Device” [Добавить устройство отображения]".
3. В диалоговом окне выберите устройство отображения, свойства которого вы желаете просмотреть.

   Результат: В нижней части диалогового окна отображаются свойства выбранного устройства отображения "Opx" ("ProTool") или "WinCC":

   - Предельные значения для текста сообщения
   - Текстовые функции
   - Предельные значения для информационного текста
   - Атрибуты
4. Если вы желаете продолжить, то выйдите из диалогового окна, щелкнув на кнопке ”Cancel [Отменить]”.

27.16.1.5 Атрибуты блокировки

Кнопка "Disable [Блокировать]" для шаблона сообщения

Вы можете блокировать атрибуты только тогда, когда вы редактируете шаблоны сообщений. Блокированные атрибуты служат только для чтения в сообщениях, получаемых из шаблона сообщения. Блокированные данные маркируются символом ключа рядом с полем ввода.

Чтобы блокировать атрибуты, действуйте следующим образом:

1. Начните с редактирования шаблонов сообщений.
2. Перейдите во вкладку для ввода атрибутов.
3. Выберите поле ввода, которое вы хотите блокировать. Рядом с полем ввода не должно быть видимого символа ключа.
4. Щелкните на кнопке "Disable [Блокировать]".
   Результат: Рядом с полем ввода появляется символа ключа.
5. На втором этапе проектирования сообщения введите атрибуты сообщений. Атрибуты, которые вы блокировали для шаблона сообщения, в этом сообщении могут только читаться.

Чтобы разблокировать атрибуты, действуйте следующим образом:

1. Начните с редактирования шаблона сообщения.
2. Перейдите во вкладку для ввода атрибутов.
3. Выберите поле ввода, которое вы хотите разблокировать. Для блокированных атрибутов рядом с полем ввода виден символ ключа. Разблокировать можно только этот тип данных.
4. Щелкните на кнопке "Disable [Блокировать]".
   Результат: Символ ключа, расположенный рядом с полем ввода, удаляется.
5. На втором этапе проектирования сообщения введите атрибуты сообщений. Теперь атрибуты, которые вы разблокировали для шаблона сообщения, можно изменять.

27.16.1.6 Блокировка текстов

Триггерная кнопка "Locked [Блокирован]" для шаблона сообщения

Вы можете блокировать тексты только тогда, когда редактируете шаблоны сообщений. Блокированные тексты служат только для чтения в сообщениях, получаемых из шаблона сообщения. Триггерная кнопка рядом с полем ввода служит для того, чтобы показать, блокирован ли текст.

Чтобы блокировать тексты, действуйте следующим образом:

1. Начните с редактирования шаблонов сообщений.
2. Перейдите во вкладку ввода текстов.
3. Щелкните по триггерной кнопке рядом с полем ввода текста, который вы хотите блокировать.
4. На втором этапе проектирования сообщения введите текст сообщения. Тексты, которые вы блокировали для шаблона сообщения, в сообщении служат только для чтения.

Чтобы разблокировать тексты, действуйте следующим образом:

1. Начните с редактирования шаблонов сообщений.
2. Перейдите во вкладку ввода текстов.
3. Щелкните по триггерной кнопке рядом с полем ввода текста, который вы хотите разблокировать.
4. На втором этапе проектирования сообщения введите текст сообщения. Теперь тесты, которые вы разблокировали, можно изменять.

27.16.2 Редактирование сообщений, связанных с символами

27.16.2.1 Создание сообщений, связанных с символами

Для создания сообщений, связанных с символами, действуйте следующим образом:

1. Выберите в SIMATIC Manager требуемую таблицу символов в соответствующем проекте и программе S7 и откройте ее.
2. Выберите сигнал, которому вы хотите назначить сообщение, связанное с символом, и выделите всю строку. Вы можете выбирать входы (I), выходы (Q) или меркеры (M) типа данных ”BOOL”.
3. Откройте функцию проектирования сообщений при помощи команды **Edit > Special Object Properties > Message [Редактировать > Специальные свойства объекта > Сообщение]**.

   Результат: Открывается диалоговое окно для проектирования сообщений.

Заполните вкладки ”Attributes” [Атрибуты]" и ”Text” [Текст]". Во вкладке ”SCAN Attributes” [Атрибуты SCAN]" сигнал, выбранный в таблице символов, через которую вы получили доступ к функции проектирования сообщений, отображается как абсолютный адрес и символический адрес.

Примечание

Убедитесь, что в триггерной кнопке "Message [Сообщение]" установлена метка, иначе сообщение, которое вы в данный момент редактируете, будет удалено, когда вы выйдете из диалогового окна проектирования сообщений.

Если вы хотите удалить сообщение, то удалите галочку триггерной кнопке "Message [Сообщение]", щелкнув по ней.

---

4. Введите требуемое контрольное время в поле ”SCAN interval” [Интервал SCAN]. Здесь вы должны принять во внимание производительность вашего CPU, потому что время сканирования, которое вы здесь вводите, может увеличить нагрузку на цикл сканирования.
5. Вкладка ”Filter” [Фильтр] поможет вам выбрать типы адресов и типы данных из таблицы символов, которые вы хотите вводить как связанные значения. Установите здесь требуемые фильтры.
6. Теперь во вкладке ”SCAN Associated Values” [Значения, связанные со SCAN] выберите поле для связанного значения 1 и щелкните на кнопке ”Add [Добавить]”.

   Результат: Таблица символов открывается и отображается с учетом выполненной вами настройки фильтра.

7. В таблице символов выберите строку, содержащую адрес, который вы хотите добавить как связанное значение (например, M. 1.0) и щелкните на кнопке ”Add [Добавить]”.

   Результат: Выбранный адрес добавляется как связанное значение во вкладку ”SCAN Associated Values” [Значения, связанные со SCAN].

8. Если вы хотите добавить несколько значений, связанных со SCAN, то повторите шаги с 6 по 8, а затем выйдите из диалогового окна проектирования сообщений с помощью ”Save [Сохранить]".

   Результат: В отображающейся таблице символов все адреса, имеющие выделенные им сообщения, показывают «крестик» в столбце «M».

9. Если вы хотите еще создавать сообщения SCAN, то повторите шаги с 2 по 8.

   Результат: Вы создали стандартные сообщения, которые могут отображаться на всех устройствах отображения.

10. Добавьте новое устройство отображения типа "WinCC", щелкнув на кнопке "New Device [Новое устройство]".

    Результат: Вставляется новое устройство и отображается соответствующая секция с вкладками.

11. Во вкладках, относящихся к устройствам отображения, введите атрибуты и тексты для сообщения, зависящего от устройства отображения.

    Результат: Вы создали вариант сообщения, который используется как сообщение только для выбранного устройства отображения.

12. Когда вы закончите проектирование всех ваших сообщений, щелкните на кнопке ”Generate SDB” [Генерировать SDB], чтобы сохранить спроектированные данные в одном или нескольких системных блоках данных.

    Результат: Все данные, сохраненные в базе данных проектирования сообщений, записываются в один или большее количество SDB.

    Примечание: Если кнопка "Generate SDB [Генерировать SDB]" не активна, то сохраните таблицу символов или щелкните на кнопке "Save [Сохранить]" в диалоговом окне проектирования сообщений.

13. Выйдите из диалогового окна проектирования сообщений с помощью ”Save” [Сохранить].

14. В соответствующей программе S7 в SIMATIC Manager выберите папку "System Data [Системные данные]", находящуюся среди блоков ("Blocks"), которая содержит сгенерированные SDB, и загрузите папку в требуемый CPU при помощи команды меню **PLC > Download [ПЛК > Загрузить]**.

15. Передайте данные, которые вы спроектировали.

27.16.2.2 Добавление связанных значений в поле

Вы можете в таблице символов выбрать символ, чтобы вставить как связанное значение.

Чтобы добавить связанное значение, действуйте следующим образом:

1. Выберите одно из десяти полей ввода для связанных значений, щелкнув по нему.
2. Щелкните на кнопке "Add [Добавить]".
   Результат: Отображается диалоговое окно для выбора символов.
3. Выберите там символ, выделив всю строку, в которой он отображается, а затем щелкнув на кнопке "Add [Добавить]".
   Результат: Выбранный символ вставляется в поле связанного значения.

Диалоговое окно остается открытым, и вы можете вставлять дополнительные связанные значения в другие поля тем же самым способом. После добавления последнего связанного значения, выйдите из диалогового окна, щелкнув на кнопке "OK".
Результат: Связанное значение вставляется, и диалоговое окно закрывается.

Примечание

Если вы хотите отменить вставку связанного значения, то выйдите из диалогового окна, щелкнув на кнопке "Cancel [Отменить]".

27.16.2.3 Удаление связанных значений из поля

1. В окне редактирования во вкладке выберите связанное значение, которое вы хотите удалить.
   Результат: Активизируется кнопка "Delete [Удалить]".
2. Щелкните на кнопке "Delete [Удалить]".

   Результат: Связанное значение удаляется. Это связанное значение больше не отображается в поле ввода.

27.16.3 Создание и редактирование диагностических сообщений, определяемых пользователем

27.16.3.1 Создание диагностических сообщений, определяемых пользователем

1. Выберите в SIMATIC Manager требуемую программу S7 и запустите приложение для проектирования сообщений при помощи команды меню **Edit > Special Object Properties > Message [Редактировать > Специальные свойства объекта > Сообщение]**.
2. Выделите отображаемую программу S7 и щелкните на кнопке "New Message [Новое сообщение]" справа.

   Результат: Вставляется новое диагностическое сообщение, определяемое пользователем, с обозначением "WR_USMSG (<No.>)", и отображается вкладка "Identification [Идентификация]".
3. Заполните вкладки "Identification [Идентификация]" и "Text [Текст]" для нового сообщения. Введите номер сообщения, если вы не хотите использовать номер, предлагаемый системой, и введите имя сообщения (идентификатор) и текст сообщения для поступающего и уходящего сообщения.

27.16.3.2 Удаление диагностических сообщений, определяемых пользователем

Вы можете удалить выбранное сообщение. Тексты, спроектированные вами для этого сообщения, удаляются, и номер сообщения снова становится свободным, как только вы выйдете из диалогового окна для проектирования сообщений с помощью "OK".

Чтобы добавить связанное значение, продолжите следующим образом:

1. Выберите сообщение, которое вы хотите удалить, в структуре каталога или в подробном представлении администратора сообщений.
2. Щелкните на кнопке "Delete [Удалить]" или используйте соответствующую команду контекстного меню, которое появляется, когда вы щелкаете правой кнопкой мыши.
3. В диалоговом окне щелчком на "Yes [Да]" подтвердите, что вы хотите удалить сообщение.
   Результат: Сообщение удаляется и больше не отображается в структуре каталога и в подробном представлении.

27.16.3.3 Добавление связанных значений к сообщениям

Для обеспечения сообщений, связанных с блоками и символами, текущей информацией, например из процесса, вы можете добавлять связанные значения в любой точке текста сообщения.

Действуйте следующим образом:

1. Сформируйте блок информации, который начинается с символа @, содержит указатель местоположения и код формата и заканчивается другим символом @.
2. Вставьте этот блок в текст сообщения в местах, где должно отображаться связанное значение.

Указатель местоположения

Это сноска, сообщающая вам, где может быть найдена текущая информация (такая как переменная, которая может управляться и контролироваться из устройства отображения).

Код формата

Он сообщает вам формат отображения связанного значения на устройстве отображения. Команда формата вводится знаком "%". Разрешенными кодами формата являются те, что используются в соответствующем языке программирования. Кроме того, имеются коды формата, установленные для текстов сообщений.

Код формата	Описание
%ix	Шестнадцатеричное число с i разрядами
%iu	Десятичное число без знака с i разрядами
%id	Десятичное число со знаком с i разрядами
%e	Нормализованное число с плавающей точкой: Значение со знаком в форме [-]d.dddd e [знак]ddd d: отдельная цифра dddd: одна или более цифр ddd: ровно три цифры знак: + или -
%E	Как формат %e, но перед экспонентой стоит буква верхнего регистра (E вместо e).

Если код формата слишком мал, то значение все еще выводится в своей полной длине.

Если код формата слишком большой, то перед значением выводится соответствующее число пробелов.

Пример связанного значения

@1%6d@: Связанное значение 1 должно отображаться как десятичное число, имеющее максимум 6 разрядов.

27.16.3.4 Удаление связанных значений

Вы можете удалять связанные значения, удаляя из текста сообщения символьную строку, представляющую связанное значение.

Действуйте следующим образом:

1. Найдите в тексте сообщения блок информации, соответствующий связанному значению, которое вы хотите удалить.
   Блок начинается символом @, за ним идут указатель местоположения, идентифицирующий связанное значение, а также код формата; блок заканчивается другим символом @.
2. Удалите эту информацию из текста сообщения.

27.16.4 Перевод и редактирование текстов пользователя

27.16.4.1 Перевод текстов пользователя

Для перевода тексты пользователя действуйте следующим образом:

1. В SIMATIC Manager при помощи команды меню **Options > Language for Display Devices.. [**Параметры > Язык для устройств отображения …] установите языки, на которые хотите переводить тексты пользователя.
2. В диалоговом окне ”Add/Delete Languages, Set Standard Language” [Добавление/удаление языков, установка стандартного языка] выберите требуемый язык из списка доступных языков и щелкните на кнопке ”->”, чтобы установить этот язык как новый язык в проекте. Для каждого нового языка, который вы устанавливаете, в списке текстов пользователя добавляется новый столбец.
3. Повторите шаг 2 для всех требуемых языков и выйдите из диалогового окна с помощью "OK".
4. В SIMATIC Manager выберите объект, тексты пользователя которого вы хотите переводить или редактировать и выберите команду меню **Options > Translate Texts** [Параметры > Перевести тексты].

   Результат: Список текстов пользователя для выбранного объекта отображается на выбранных языках.
5. Поместите курсор в поле для ввода текста перевода. Введите текст или сделайте изменения в уже существующем тексте. Здесь доступны различные опции меню **Edit** [Редактирование] (такие как Find [Найти], Replace [Заменить] и т.д.). Обратите внимание на то, что вы можете только форматировать тексты сообщений для панелей оператора, созданные в приложении для проектирования сообщений, с помощью кнопок на панели инструментов.

6. По окончании перевода и редактирования сохраните тексты при помощи команды меню **Texts > Save [Тексты > Сохранить]**.
7. Вы можете использовать команду меню **Texts > Print [Тексты > Печатать]**, чтобы распечатать тексты.
8. Посредством команды меню **Texts > Exit [Тексты > Выйти]** выйдите из функции, когда вы перевели или отредактировали все требуемые тексты.

27.16.4.2 Экспортирование текстов пользователя

1. В SIMATIC Manager при помощи команды меню **Options > Language for Display Devices.. [Параметры > Язык для устройств отображения …]** установите языки, на которые хотите переводить тексты пользователя.
2. В SIMATIC Manager выберите объект (проект, программа S7, папка блоков, блок или таблица символов), тексты пользователя из которого вы хотите печатать, и выберите команду меню **Options > Translate Texts [Параметры > Перевести тексты]**.
   Выберите команду меню **Texts > Export [Тексты > Экспортировать]**.
3. В диалоговом окне "Export [Экспорт]", выберите папку, в которой вы хотите сохранить список.
4. Выберите тип файла и введите имя файла, под которым вы хотите сохранить экспортируемые тексты пользователя.

Примечания

Для упрощения импортирования вам следует выбирать имя файла, показывающее, из какой части проекта экспортировались тексты.

Для экспортирования текстов, которые вы хотите редактировать с помощью табличного редактора, используйте тип файла *.CSV.

27.16.4.3 Импортирование текстов пользователя

1. Выделите проект, в который вы хотите импортировать тексты, и выберите команду меню **Options > Translate Texts [Параметры > Перевести тексты]**.
2. Выберите команду меню **Texts > Import [Тексты > Импортировать]**.
3. В диалоговом окне "Import [Импорт]" выберите папку, где расположен список, который вы хотите импортировать.
4. Выберите тип файла и имя файла и подтвердите с помощью "Open [Открыть]".

27.16.5 Передача данных проектирования сообщений в программируемый контроллер

27.16.5.1 Передача данных проектирования

Когда вы передаете данные проектирования в WinCC, вас поддерживает мастер "Transfer PLC Data to Operator Station [Передача данных ПЛК станции оператора]". Действуйте следующим образом:

1. Выберите тип программируемого контроллера и станции оператора (например, WinCC), в которую вы хотите передавать данные.
2. Назначьте программу S7, в которой вы создавали проектные данные, станции оператора, на которой вы хотите отображать сообщения. Вы можете назначать несколько программ S7 одной станции оператора или несколько станций оператора одной программе S7. Это означает, что вы имеете возможность отображать сообщения на разных станциях оператора.
3. Выделите программу S7 и используйте правую кнопку мыши, чтобы выбрать команду меню **Select Network Connections [Выбрать сетевые соединения]**. Выберите сеть, которая должна использоваться для связи между программируемым контроллером и станцией оператора во время выполнения.
4. Выберите, какая программа S7 должна передаваться и какой станции оператора вы хотите передавать данные.
5. Здесь выберите требуемые параметры передачи.

   Вы можете выбирать из следующего:

   - Transfer Data [Передаваемые данные]: здесь выберите, какие данные вы хотите передавать.
   - Size of Transfer [Объем передачи]: вы можете передавать станции оператора все данные проектирования или только измененные данные проектирования. Если вы выбираете опцию ”All [Все]", то вы можете также одновременно удалить все ранее переданные данные (”Clear operator station(s) [Очистить станцию(и) оператора])".
   - Comparison [Сравнение]: выберите эту опцию, чтобы гарантировать, что во время передачи будут передаваться непротиворечивые данные. Это важно, если вы сделали изменения в период между созданием и передачей данных.

Примечание

Обратите внимание на то, что выполнение сравнения увеличит время передачи.

- Create Logs [Создать файлы регистрации]: Если вы выбираете эту опцию, то создается файл, который перечисляет, какие дескрипторы (теги) WinCC каким объектам изображений назначены, и создает журнал регистрации для передачи данных проектирования.
- Compress [Сжать]: щелкните на этой кнопке, если вы хотите удалить также физически все дескрипторы (теги), отмеченные как удаленные.
- В качестве дополнительного параметра передачи вы можете также выбрать, какую стратегию символов замены вы хотите использовать. Обратитесь здесь к примечаниям в контекстно-зависимой справке для диалогового окна.

  Здесь вы можете также устанавливать источник сообщения: HID; HID + chart name [имя схемы]; HID + chart name [имя схемы] + block name [имя блока].

6. Щелкните на кнопке ”Finish” [Конец]. Начинается передача данных.

27.17 Как проектировать управление и контроль переменных со стороны оператора

27.17.1 Назначение системных атрибутов параметрам функционального блока

Когда вы проектируете управление и контроль атрибутов с помощью AWL, KOP и FUP, вы должны сначала назначить системный атрибут ”s7_m_c" всем параметрам функционального блока, который вы хотите подготовить для управления и контроля. Действуйте следующим образом:

1. Откройте функциональный блок (FB).
2. Выберите параметр в таблице описания переменных, который вы хотите подготовить для управления и контроля.
3. При помощи правой кнопки мыши выберите команду меню **Object Properties [Свойства объекта]**. В диалоговом окне ”Parameter Properties [Свойства параметров]" введите строку ”s7_m_c" в столбце ”System Attribute [Системный атрибут]" и ”true [истина]" в столбце ”Value [Значение]" пустой строки.
4. Если требуется, введите другие системные атрибуты для параметра. Вы найдете полный список атрибутов системы в оперативной справке.
5. Выйдите из диалогового окна, щелкнув на ”OK".
6. Повторите эту процедуру для всех параметров, которые вы хотите подготовить для управления и контроля.

27.17.2 Назначение атрибутов WinCC блокам данных

Чтобы назначить атрибуты WinCC экземплярам функционального блока или совместно используемым блокам данных, действуйте следующим образом:

1. В SIMATIC Manager или редакторе LAD/STL/FBD (KOP/AWL/FUP) создайте один или несколько совместно используемых или экземплярных блоков данных, которые связаны с подготовленным функциональным блоком.
2. Выделите блок данных в SIMATIC Manager.
3. Выберите команду меню **Edit > Special Object Properties > Operator Control and Monitoring [Редактировать > Специальные свойства объекта > Управление и контроль со стороны оператора]**.
4. В диалоговом окне ”Operator Control and Monitoring [Управление и контроль со стороны оператора]" активизируйте триггерную кнопку ”Operator Control and Monitoring [Управление и контроль со стороны оператора]".
5. Выберите вкладку "General [Общие свойства]".

   Здесь имя блока данных отображается таким, каким оно появляется в WinCC (имя программы S7_номер DB или
   имя программы S7_символическое имя DB).

   В случае необходимости введите в поле ”Comment [Комментарий]" дополнительную информацию о блоке данных.
6. Теперь выберите вкладку WinCC Attributes [Атрибуты WinCC], чтобы редактировать атрибуты WinCC соответствующего блока данных.
7. В появляющейся таблице введите требуемые значения атрибутов для всех параметров функциональных блоков, используемых в управлении и контроле со стороны оператора.
8. Закройте диалоговое окно, щелкнув на кнопке ”OK".
9. Повторите шаги с 2 по 8 для каждого блока данных.

27.17.3 Изменение атрибутов WinCC для параметров блока CFC

Чтобы изменить предварительно установленные атрибуты WinCC для параметров блока CFC, действуйте следующим образом:

1. Выделите блок.
2. Выберите команду меню **Edit > Object Properties [Редактировать> Свойства объекта]**, чтобы редактировать свойства блока CFC.
3. Щелкните на кнопке ”Operator Control and Monitoring"[Управление и контроль со стороны оператора]".
4. В случае необходимости измените уже введенные значения атрибутов в таблице, отображаемой в диалоговом окне ”Operator Control and Monitoring [Управление и контроль со стороны оператора]".
5. За значениями атрибутов WinCC обратитесь к оперативной справке.
6. Закройте диалоговое окно, щелкнув на кнопке ”OK".

27.17.4 Вставка объектов станции оператора

Для каждой системы операторского управления и контроля вы должны создать объект OS [Станция оператора] в SIMATIC Manager. Действуйте следующим образом:

1. Откройте ваш проект STEP 7.
2. Выберите команду меню **Insert > WinCC Object > Operator Station [Вставить > Объект WinCC > Станция оператора]**.

Примечание

Обратите внимание на то, что количество станций оператора, для которых должны передаваться данные, влияет на продолжительность передачи.

27.17.5 Запуск программы передачи

Для запуска программы передачи у вас есть две возможности:

- Выберите команду меню **Options > PLC-OS Connection Data > Transfer** [Параметры > Данные соединения ПЛК-OS > Передача] в SIMATIC Manager

или

1. Откройте PLC-OS Engineering tool [Инструментальное средство для проектирования соединений ПЛК-OS] через **Start > Simatic > STEP 7 > PLC-OS Engineering [Пуск > Simatic > STEP 7 > Проектирование ПЛК-OS]** из стартового меню Windows.
2. Откройте ваш проект STEP 7.
3. Выберите команду меню **Project > Transfer [Проект > Передать]**.

27.17.6 Передача данных

Когда вы передаете данные проектирования в WinCC, вас поддерживает мастер "Transfer PLC Data to Operator Station [Передача данных ПЛК на станцию оператора]". Действуйте следующим образом:

1. Выберите тип программируемого контроллера и станции оператора (например, WinCC), на которую вы хотите передавать данные.
2. Назначьте программу S7, в которой вы создавали данные конфигурации, станции оператора, на которой вы хотите выполнять управление и контроль. Вы можете назначить несколько программ S7 одной станции оператора или несколько станций оператора одной программе S7. Это позволяет вам выполнять управление и контроль конкретных процессов с разных станций оператора.
3. Выберите программу S7 и используйте правую кнопку мыши, чтобы выбрать команду меню **Select Network Connections [Выбрать сетевые соединения]**. Выберите сеть, которая должна использоваться для связи между программируемым контроллером и станцией оператора во время выполнения
4. Выберите, какая программа S7 должна передаваться и какой станции оператора вы хотите передать данные.
5. Здесь выберите требуемые параметры передачи.

Вы можете выбирать из следующего:

- Transfer Data [Передаваемые данные]: здесь выберите, какие данные вы хотите передать.
- Size of Transfer [Объем передачи]: вы можете передать станции оператора все данные проектирования или только измененные данные проектирования. Если вы выбираете опцию ”All [Все]", то вы можете также одновременно удалить все ранее переданные данные (”With memory reset on OS [Со сбросом памяти в OS]").
- Update [Обновить]: выберите эту опцию, чтобы гарантировать, что во время передачи передаются непротиворечивые данные. Это важно, если вы сделали изменения в период между созданием и передачей данных.

  Примечание

  Обратите внимание на то, что выполнение обновления увеличивает время передачи.

- Create Logs [Создать файлы регистрации]: если вы выбираете эту опцию, то создается файл, перечисляющий, какие переменные WinCC каким объектам изображений назначены, или создается файл регистрации для передачи данных проектирования.
- Compress [Сжать]: Щелкните на этой кнопке, если вы хотите удалить также физически все переменные, уже отмеченные как удаленные.

  В качестве дополнительного параметра передачи вы можете также выбрать, какая стратегия символов замены должна использоваться. Обратите внимание на информацию об этом диалоговом окне, которая содержится в оперативной справке.

6. Щелкните на кнопке ”Finish” [Конец]. Теперь начинается передача данных.

27.17.7 Отображение файла регистрации передачи

Если в параметрах передачи вы выбрали "Transfer Log [Файл регистрации передачи]", то создается файл регистрации, содержащий информацию о: существующих соединениях ПЛК-OS, ошибках, произошедших во время передачи, именах переменных и т.д. Чтобы отобразить файл регистрации передачи:

- Выберите команду меню **Options > PLC-OS Connection Data > Display Log [Параметры > Данные соединений ПЛК-OS > Отобразить файл регистрации]**.

28 Как создавать online-соединения и настраивать CPU

28.1 Как устанавливать online-соединения

28.1.1 Установка соединения из проекта с конфигурированными аппаратными средствами

1. Активизируйте требуемое окно проекта
2. Выберите команду меню **View > Online [Вид > Online]"**, чтобы открыть окно проекта в режиме online.
3. Дважды щелкните по станции, чтобы просмотреть программируемые модули на станции.
4. Дважды щелкните по модулю, с которым вы хотите установить соединение.

28.1.2 Установка соединения из проекта без сконфигурированных аппаратных средств

1. Активизируйте требуемое окно проекта
2. Выберите команду меню **View > Online [Вид > Online]"**, чтобы открыть окно проекта в режиме online.
3. Выберите программу S7 или программу M7, расположенную непосредственно под проектом.
4. Выберите команду меню **Edit > Object Properties** [Редактировать > Свойства объекта]". В появляющемся диалоговом окне введите адрес MPI программируемого модуля, к которому вы хотите обратиться.
5. Закройте диалоговое окно.

28.1.3 Установка соединения без проекта

Этот тип доступа позволяет вам быстро обратиться к программируемому логическому контроллеру, например, с целью тестирования. Вы можете обращаться ко всем доступным программируемым модулям в сети.

1. Откройте окно "Accessible Nodes [Доступные узлы]", используя команду меню **PLC > Display Accessible Nodes [ПЛК > Отобразить доступные узлы]"** или щелкая по соответствующей кнопке на панели инструментов.
2. В окне "Accessible Nodes [Доступные узлы]" выберите программируемый модуль, с которым вы хотите установить соединение. Вы можете идентифицировать модуль через адрес MPI, заданный в его имени.

28.2 Как переключить режим работы

28.2.1 Переключение режима работы CPU S7

Действуйте следующим образом:

1. Установите online-соединение с программируемым контроллером, используя один из следующих методов:
   - Откройте окно проекта online и выберите модуль, либо программу S7 или M7.
   - Выберите объект ”MPI=..." в окне "Accessible Nodes [Доступные узлы]".
2. Выберите команду меню **PLC > Operating Mode** [ПЛК > Режим работы]". В диалоговом окне отобразится текущий режим.
3. Нажмите соответствующую кнопку, чтобы переключить режим работы. Кнопка деактивирована (показана более бледным тоном), если в текущей ситуации переключение в этот режим работы не разрешено.

29 Как производить загрузку и считывание

29.1 Загрузка всей программы в CPU S7

29.1.1 Загрузка с управлением проектом

1. В окне проекта выберите программу пользователя или блоки, которые вы хотите загрузить.
2. Загрузите выбранные объекты в программируемый логический контроллер, выбирая команду меню **PLC > Download [ПЛК > Загрузить]**.

Альтернативная процедура (буксировка)

1. Откройте окно offline и окно online вашего проекта.
2. Выберите в окне offline объекты, которые вы хотите загрузить, и отбуксируйте их в окно online.

29.1.2 Загрузка без управления проектом

1. Откройте окно "Accessible Nodes [Доступные узлы]", используя команду меню **PLC > Display Accessible Nodes [ПЛК > Отобразить доступные узлы]**" или щелкнув по соответствующей кнопке на панели инструментов.
2. Дважды щелкните в окне "Accessible Nodes [Доступные узлы]" по требуемому узлу ("MPI=..."), чтобы отобразить папку "Blocks [Блоки]".
3. Откройте библиотеку или проект, из которого вы хотите загрузить программу пользователя или блоки в программируемый логический контроллер. Для этого используйте команду меню **File > Open [Файл > Открыть]**".
4. В окне, открывающемся для проекта или библиотеки, выберите объекты, которые вы хотите загрузить.
5. Загрузите объекты в программируемый логический контроллер, копируя их в папку "Blocks [Блоки]" в окне "Accessible Nodes [Доступные узлы]" при помощи буксировки.

29.1.3 Перезагрузка блоков в программируемый контроллер

Вы можете заменять блоки, уже существующие в загрузочной памяти (ОЗУ) или в рабочей памяти CPU программируемого логического контроллера S7, новыми версиями (перезагружать их). При этом происходит замена существующей версии.

Процедура перезагрузки блоков S7 является такой же, как процедура загрузки. Просто появляется подсказка, запрашивающая, хотите ли вы перезаписать существующий блок.

Блок, хранимый в СППЗУ, не может быть удален, но объявляется недействительным, как только он будет перезагружен. Заменяющий блок загружается в ОЗУ. Это создает пустоты в загрузочной памяти или рабочей памяти. Если эти пустоты, в конечном счете, означают, что новые блоки больше не могут загружаться, то вам нужно сжать память.

Примечание

Если прекращается и затем восстанавливается электропитание и ОЗУ не имеет резервного батарейного питания или выполняется сброс памяти CPU, то "старые" блоки снова становятся действительными.

29.1.4 Сохранение загруженных блоков во встроенном СППЗУ

В случае CPU (таких как CPU 312), имеющих встроенный СППЗУ, вы можете сохранять блоки из ОЗУ во встроенном СППЗУ, чтобы не терять данные после выключения питания или сброса памяти.

1. Используйте команду меню **View > Online [Вид > Online]**, чтобы отобразить окно, содержащее представление online открытого проекта, или откройте окно "Accessible Nodes [Доступные узлы]", щелкая по кнопке "Accessible Nodes [Доступные узлы]" на панели инструментов или выбирая команду меню **PLC > Display Accessible Nodes [ПЛК > Отобразить доступные узлы]**.
2. Выделите программу S7 или M7 в окне проекта online или узел в окне "Accessible Nodes [Доступные узлы]".
3. Выделите в CPU папку "Blocks [Блоки]", которую вы хотите сохранить, при помощи одного из следующих методов:
   - в окне проекта online, если вы работаете с управлением проектом
   - в окне "Accessible Nodes [Доступные узлы]", если вы работаете без управления проектом.
4. Выберите команду меню **PLC > Save RAM to ROM [ПЛК > Сохранить ОЗУ в ПЗУ]**.

29.1.5 Загрузка через платы памяти СППЗУ

Требования

Для доступа к платам памяти СППЗУ, предназначенным для программируемого логического контроллера S7, вам в устройстве программирования потребуются соответствующие драйверы СППЗУ. Для доступа к платам памяти СППЗУ, предназначенным для программируемой системы управления M7, должна быть установлена Flash File System [Файловая система флэш-памяти]" (возможно только в PG 720, PG 740 и PG 760). Драйверы СППЗУ и Flash File System предлагаются как опции, когда вы устанавливаете стандартный пакет STEP 7. Если вы используете PC, то для сохранения на платах памяти СППЗУ потребуется внешний программатор.

Вы можете устанавливать драйверы и позднее. Чтобы сделать это, вызовите соответствующее диалоговое окно через **Start > Simatic > STEP 7 > Memory Card Parameter Assignment** [Пуск > Simatic > STEP 7 > Настройка параметров платы памяти]" или через Control Panel [Панель управления]" (двойной щелчок по значку "Memory Card Parameter Assignment [Настройка параметров платы памяти]").

Сохранение на плате памяти

Для сохранения блоков или программ пользователя на плате памяти действуйте следующим образом:

1. Вставьте плату памяти в слот вашего устройства программирования.
2. Откройте окно "Memory Card [Плата памяти]":
   - Щелчком по кнопке "Memory Card [Плата памяти]" на панели инструментов. В случае необходимости активизируйте панель инструментов, используя команду меню **View > Toolbar [Вид > Панель инструментов]**.
   - Альтернативно, выбирая команду меню **File > S7 Memory Card > Open [Файл > Плата памяти S7 > Открыть]**.
3. Откройте или активизируйте одно из следующих окон, отображающих блоки, которые вы хотите сохранить. Возможны следующие окна:
   - Окно проекта, представление "ONLINE"
   - Окно проекта, представление "offline"
   - Окно библиотеки
   - Окно "Accessible Nodes [Доступные узлы]".
4. Выберите папку "Blocks [Блоки]" или отдельные блоки и скопируйте их в окно "S7 memory card [Плата памяти S7]".
5. Если блок уже существует на плате памяти, то выводится сообщение об ошибке. В этом случае сотрите содержимое платы памяти и повторите шаги, начиная с 2-го.

29.2 Считывание всей программы из CPU S7

29.2.1 Считывание блоков в соответствующий проект на устройстве программирования

1. В SIMATIC Manager откройте окно online проекта, используя команду меню **View > Online [Вид > Online]**.
2. В окне online выделите папку блоков или набор блоков в папке.
3. Выберите команду меню **PLC > Upload [ПЛК > Считать]**.

Выбранные объекты передаются в базу данных проекта на устройстве программирования.

Альтернативно, вы можете копировать соответствующие объекты также из окна проекта online в окно проекта offline.

Если соответствующий проект в устройстве программирования недоступен, то в вашем распоряжении имеются следующие возможности:

Считывание блоков в другой проект в устройстве программирования

Считывание блоков в новый проект в устройстве программирования

29.2.2 Считывание блоков в другой проект в устройстве программирования

1. В SIMATIC Manager откройте окно "Accessible Nodes [Доступные узлы]", щелкнув по соответствующей кнопке панели инструментов или выбрав команду меню **PLC > Display Accessible Nodes [ПЛК > Отобразить доступные узлы]**.
2. Дважды щелкните по узлу ("MPI=...").
3. Выберите папку "Blocks [Блоки]" или отдельные блоки в папке.
4. Скопируйте выбранную папку "Blocks [Блоки]" в программу S7 или скопируйте выбранные блоки в папку "Blocks [Блоки]" в окне offline другого проекта.

29.2.3 Считывание блоков в новый проект в устройстве программирования

1. Вначале создайте новый проект.
2. Вставьте программу S7.
3. Затем откройте окно online этого проекта, используя команду меню **View > Online [Вид > Online]**.
4. Откройте в окне online программу S7 и откройте в ней папку "Blocks [Блоки]".
5. Если подключено более одного программируемого контроллера, то отображается диалоговое окно. В диалоговом окне введите адрес MPI программируемого контроллера, из которого вы хотите считать блоки.
6. Выберите команду меню **PLC > Upload [ПЛК > Считать]**.

Альтернативно, вы можете скопировать папку "Blocks [Блоки]" или набор блоков в окне online и вставить их в окно offline.

29.2.4 Редактирование считанных блоков, если программа пользователя находится в PG / PC

Для редактирования блоков из CPU действуйте следующим образом:

1. Откройте окно online проекта в SIMATIC Manager.
2. Выберите в интерактивном окне папку "Blocks [Блоки]". Отображается список загруженных блоков.
3. Теперь выбирайте блоки, открывайте и редактируйте их.
4. Выберите команду меню **File > Save [Файл > Сохранить]**, чтобы сохранить изменение offline в устройстве программирования.
5. Выберите команду меню **PLC > Download [ПЛК > Загрузить]**, чтобы загрузить измененные блоки в программируемый контроллер.

29.2.5 Редактирование считанных блоков, если программа пользователя не находится на PG / PC

Для редактирования блоков из CPU действуйте следующим образом:

1. В SIMATIC Manager щелкните по кнопке "Accessible Nodes [Доступные узлы]" на панели инструментов или выберите команду меню **PLC > Display Accessible Nodes [ПЛК > Отобразить доступные узлы]**.
2. Выберите в отображаемом списке узел (объект ”MPI=...") и откройте папку "Blocks [Блоки]", чтобы отобразить блоки.
3. Теперь вы можете открывать блоки и редактировать, просматривать или копировать их требуемым образом.
4. Выберите команду меню **File > Save As [Файл > Сохранить как...]** и введите в диалоговом окне путь для устройства программирования, где вы хотите хранить блоки.
5. Выберите команду меню **PLC > Download [ПЛК > Загрузить]**, чтобы загрузить измененные блоки в программируемый контроллер.

29.2.6 Сжатие содержимого памяти CPU S7

Способы сжатия памяти

Есть два следующих метода сжатия пользовательской памяти:

- Если при загрузке в программируемый контроллер недостаточно доступной памяти, то появляется диалоговое окно, сообщающее вам об ошибке. Вы можете сжимать память, щелкая по соответствующей кнопке в диалоговом окне.
- В качестве профилактической меры вы можете отображать показатель использования памяти (команда меню **PLC > Module Information** [ПЛК > Информация о модуле]", вкладка "Memory [Память]") и запускать функцию сжатия, если требуется.

Последовательность действий

1. Выберите программу S7 в окне "Accessible Nodes [Доступные узлы]" или в представлении проекта online.
2. Выберите команду меню **PLC > Module Information [ПЛК > Информация о модуле]**.
3. В появляющемся диалоговом окне выберите вкладку "Memory [Память]". На этой странице вкладки имеется кнопка сжатия памяти, если CPU поддерживает эту функцию.

29.3 Как удалять в программируемом контроллере

29.3.1 Выполнение сброса памяти в CPU/FM

1. Переключите CPU/FM в режим STOP следующим образом:
   - Установите переключатель режимов в положение STOP.
   - Если переключатель режимов установлен в положение RUN-P, то вы можете установить CPU/FM в состояние STOP также при помощи команды меню **PLC > Operating Mode [ПЛК > Режим работы]**. Для этого нужно выбрать программу S7 в окне проекта online или в окне "Accessible Nodes [Доступные узлы]".
2. Выберите команду меню **PLC > Clear/Reset [ПЛК > Очистить/ Сбросить]**.
3. Подтвердите сброс памяти в появляющемся диалоговом окне.

29.3.2 Стирание ОЗУ программируемого контроллера

Вы можете удалить один или большее количество блоков в режимах STOP и RUN-P. Если вы удаляете блок в режиме RUN-P, который все еще вызван, то либо CPU переключается в STOP, либо вызывается OB ошибки.

Для удаления блоков из ОЗУ действуйте следующим образом:

1. В окне проекта online или окне "Accessible Nodes [Доступные узлы]" выделите блоки, которые вы хотите удалить.
2. В SIMATIC Manager выберите команду меню **Edit > Delete** [Редактировать > Удалить] или нажмите DEL.

Чтобы удалить всю программу пользователя CPU, вы можете также выполнить сброс памяти на CPU.

29.3.3 Стирание платы памяти СППЗУ

Для стирания платы памяти СППЗУ действуйте следующим образом:

1. Вставьте плату памяти в слот вашего программатора.
2. Щелкните по кнопке слота платы памяти на панели инструментов (S7 Memory Card). Отображается окно "S7 Memory Card [Плата памяти S7]".
3. Выберите в этом окне папку "S7 Memory Card [Плата памяти S7]".
4. Выберите команду меню **Edit > Delete [Редактировать > Удалить]** или нажмите DEL

Примечание

Вы можете стирать платы памяти СППЗУ только в устройстве программирования.

В случае плат памяти для программируемых контроллеров S7 вы можете стирать только всю плату памяти, что означает удаление всех блоков. Вы не можете удалять отдельные блоки.

В случае плат памяти для программируемых систем управления M7 файловая система флэш-памяти позволяет удалять отдельные объекты.

Альтернативная процедура:

1. Вставьте плату памяти в слот вашего устройства программирования.
2. Выберите команду меню **File > S7 Memory Card > Delete [Файл > Плата памяти S7 > Удалить]**.

Этим способом вы можете стирать также платы памяти, которые вы не можете открыть.

29.3.4 Стирание встроенного СППЗУ

Встроенный СППЗУ CPU 312 стирается посредством записи в СППЗУ текущего содержимого ОЗУ, в котором были стерты все блоки пользователя.

Для стирания СППЗУ действуйте следующим образом:

1. В SIMATIC Manager откройте окно "Accessible Nodes [Доступные узлы]", используя команду меню **PLC > Display Accessible Nodes [ПЛК > Отобразить доступные узлы]**, или окно online проекта, используя команду меню **View > Online [Вид > Online]**.
2. Выберите блоки, которые вы хотите удалить.
3. Выберите команду меню **Edit > Delete [Редактировать > Удалить]** или нажмите DEL.
4. Выберите команду меню **PLC > Save RAM to ROM [ПЛК > Сохранить ОЗУ в ПЗУ]**.

30 Как производить отладку

30.1 Тестирование с помощью таблицы переменных

30.1.1 Как создать и открыть таблицу переменных

Альтернатива 1 в SIMATIC Manager

1. Выберите представление проекта offline.
2. Откройте папку блоков, в которой вы хотите сохранить таблицу переменных.
3. Выберите команду меню **Insert > S7 Block > Variable Table [Вставить > Блок S7 > Таблица переменных]**.
4. В диалоговом окне укажите имя таблицы переменных (VAT + номер).
5. Вы можете открыть таблицу переменных, дважды щелкнув по объекту.

Альтернатива 2 в SIMATIC Manager

- В окне online (представление проекта online или "Accessible Nodes [Доступные узлы]") выберите команду меню **PLC > Monitor/Modify Variables [ПЛК > Наблюдать/изменять переменные]**. Открывается окно "Monitoring and Modifying Variables [Наблюдение и изменение переменных]".

Альтернатива 3 в "Monitor/Modify Variables"

- В отображаемом окне создайте новую таблицу переменных, используя команду меню **Table > New** [Таблица > Новая].

30.1.2 Выбор формата наблюдения

Используя команду меню **View > Select Monitor Format [Вид > Выбрать формат наблюдения]**, вы можете выбрать для каждой переменной формат, в котором хотите представлять значение переменной. Вы можете также щелкать по установленному формату, пока не отобразится требуемый формат. В распоряжении имеются следующие форматы: двоичный, шестнадцатеричный, десятичный, вещественный, символьный, булевский, время, дата, время SIMATIC, время суток, счетчик, указатель.

Вы можете изменить формат наблюдения для одной строки таблицы или для нескольких строк сразу.

Изменение формата наблюдения для одной строки таблицы

1. Щелкните по соответствующей строке в столбце "Monitor Format [Формат наблюдения]".
2. Выберите требуемый формат наблюдения из отображаемого списка.

Альтернативно, вы можете расположить курсор в строке переменной, формат наблюдения которой хотите установить, и изменить формат при помощи команды меню **View > Select Monitor Format [Вид > Выбрать формат наблюдения]**.

Изменение формата наблюдения сразу для нескольких строк таблицы

1. Выделите в таблице область, для которой вы хотите изменить формат наблюдения, перемещая курсор через эту область таблицы при нажатой левой кнопке мыши.
2. Выберите формат при помощи команды меню **View > Select Monitor Format [Вид > Выбрать формат наблюдения]**. Изменяются только те строки в выделенной области таблицы, для которых изменение формата разрешено.

На форматы наблюдения, которые вы можете выбирать, влияет ввод задаваемых значений. Формат, используемый при задании значений, применяется также при отображении наблюдаемых значений.

30.1.3 Включение и отключение отображения столбцов в таблицах переменных

Если вы не выключили отображение каких-либо столбцов, то таблица переменных (VAT) отображается со всеми своими столбцами. Вы можете показывать или скрывать отдельные столбцы таблицы, используя команды меню "View [Вид]".

Столбец таблицы переменных	Команда меню для включения/ выключения отображения столбца
Address [Адрес]	(всегда виден)
Symbol [Символ]	View > Symbol [Вид > Символ]
Symbol Comment [Комментарий к символу] (этот столбец не отображается, когда вы создаете новую таблицу переменных)	View > Symbol Comment [Вид > Комментарий к символу]
Monitor Format [Формат наблюдения]	View > Monitor Format [Вид > Формат наблюдения]
Monitor Value [Наблюдаемое значение]	View > Monitor Value [Вид > Наблюдаемое значение]
Modify Value [Задаваемое значение]	View > Modify Value [Вид > Задаваемое значение]

Отображаются только столбцы, отмеченные галочкой в меню "View [Вид]".

Установка ширины столбца

Чтобы оптимально настроить размер/отображение таблицы переменных (VAT):

- Установите ширину столбца при помощи мыши:
  - Расположите курсор на вертикальной линии, разделяющей столбцы в строке заголовков.
  - Нажмите левую кнопку мыши.
  - Перемещайте разделительную линию по горизонтали до тех пор, пока столбец не примет требуемую ширину.
  - Отпустите кнопку мыши.
- Автоматическая настройка при вводе: Когда вы изменяете формат наблюдения, ширина столбца устанавливается автоматически.
- При использовании команды меню **View > Optimize Column Widths [Вид > Оптимизировать ширину столбцов]** или **View > Select Monitor Format [Вид > Выбрать формат наблюдения]** все столбцы таблицы переменных настраиваются на свою оптимальную ширину.

30.1.4 Вырезание выделенных областей в буфер обмена

1. Выделите:
   одну или более законченных строк, нажав левую кнопку мыши и перемещая курсор вверх или вниз;
   адрес, символ или задаваемое значение, нажав левую кнопку мыши и перемещая курсор слева направо.
2. Вырежьте эту область в буфер обмена, используя команду меню **Edit > Cut [Редактировать > Вырезать]**. Эта область остается в буфере обмена до тех пор, пока она не будет перезаписана.

30.1.5 Вставка областей из буфера обмена в таблицу переменных

1. Расположите курсор в строке таблицы переменных (VAT), в которую хотите вставить область из буфера обмена.
2. Вставьте область из буфера обмена при помощи команды меню **Edit > Paste [Редактировать > Вставить]**.

30.1.6 Копирование выбранных областей в буфер обмена

1. Выделите:
   одну или более законченных строк, нажав левую кнопку мыши и перемещая курсор вверх или вниз;
   адрес, символ или задаваемое значение, нажав левую кнопку мыши и перемещая курсор слева направо.
2. Скопируйте эту область в буфер обмена, используя команду меню **Edit > Copy** [Редактировать > Копировать].

30.1.7 Копирование из таблицы символов в таблицу переменных

1. Откройте таблицу символов (если она еще не открыта).
2. Выделите требуемые символы в столбце "Symbol [Символ]" или столбце "Address [Адрес]".
3. Выберите команду меню **Edit > Copy [Редактировать > Копировать]** или щелкните по соответствующей кнопке на панели инструментов.
4. Откройте таблицу переменных (если она еще не открыта).
5. Расположите курсор в пустой строке или в начале строки таблицы переменных.
6. Выберите команду меню **Edit > Paste [Редактировать > Вставить]** или щелкните по соответствующей кнопке на панели инструментов.

Данные для выбранных символов вводятся в таблицу переменных.

30.1.8 Вставка непрерывной области адресов в таблицу переменных

1. Откройте таблицу переменных.
2. Расположите курсор в строке, после которой хотите вставить непрерывный интервал адресов.
3. Выберите команду меню **Insert > Range of Variables [Вставить > Диапазон переменных]**. Появляется диалоговое окно "Insert Range of Variables [Вставить диапазон переменных]".
4. Введите начальный адрес в поле "From Address [С адреса]".
5. Введите количество вставляемых строк в поле "Number [Количество]".
6. Выберите требуемый формат наблюдения из отображаемого списка.
7. Щелкните по кнопке "OK".

Область переменных вставляется в таблицу переменных.

30.1.9 Наблюдение переменных с определенным видом запуска

1. При помощи команды меню **Table > Open** [Таблица > Открыть] откройте таблицу переменных (VAT), содержащую переменные, которые вы хотите наблюдать, или активизируйте окно, содержащее требуемую таблицу переменных.
2. Выберите команду меню **PLC > Connect To > ... [ПЛК > Соединить с > ...]**, чтобы установить соединение с требуемым CPU, так чтобы вы могли наблюдать переменные активной таблицы переменных.
3. При помощи команды **Variable > Trigger [Переменная > Запуск]** определите момент и частоту запуска наблюдения переменных.

Примечание

Вы **не можете** определять тип запуска в то время, когда выполняется функция Monitor [Наблюдение] или Modify [Изменение]. В случае необходимости остановите функцию наблюдения при помощи команды меню **Variable > Monitor** [Переменная > Наблюдать]. Если около команды меню Monitor [Наблюдать] нет видимой метки, то наблюдение деактивировано.

---

4. В диалоговом окне определите момент запуска и частоту запуска наблюдения переменных.
5. Выберите требуемый момент запуска и частоту запуска, щелкая по соответствующим кнопкам выбора.
6. Запустите функцию наблюдения при помощи команды меню **Variable > Monitor** [Переменная > Наблюдать]. Метка около команды показывает, что наблюдение началось.
7. Вы можете остановить функцию Monitor [Наблюдение], снова выбирая команду меню **Variable > Monitor** [Переменная > Наблюдать]. Чтобы определить новый тип запуска, начните опять с шага 3.

30.1.10 Однократное и немедленное наблюдение переменных

Действуйте следующим образом:

1. При помощи команды меню **Table > Open [Таблица > Открыть]** откройте таблицу переменных (VAT), содержащую переменные, которые вы хотите наблюдать, или активизируйте окно, содержащее требуемую таблицу переменных.
2. Выберите команду меню **PLC > Connect To > ... [ПЛК > Соединить с > ...]**, чтобы установить соединение с требуемым CPU, так чтобы вы могли наблюдать переменные активной таблицы переменных.
3. Выберите команду меню **Variable > Update Monitor Values [Переменная > Обновить наблюдаемые значения]**, чтобы отобразить значения переменных однократно и немедленно.

30.1.11 Изменение переменных с определенным видом запуска

1. При помощи команды меню **Table > Open [Таблица > Открыть]** откройте таблицу переменных (VAT), содержащую переменные, которые вы хотите изменить, или активизируйте окно, содержащее требуемую таблицу переменных.
2. Выберите команду меню **PLC > Connect To > ... [ПЛК > Соединить с > ...]**, чтобы установить соединение с требуемым CPU, так чтобы вы могли изменять значения переменных активной таблицы переменных.
3. Определите момент запуска и частоту запуска изменения переменных, используя команду **Variable > Trigger [Переменная > Запуск]**.

Примечание

Вы **не можете** определять тип запуска в то время, когда выполняется функция Monitor [Наблюдение] или Modify [Изменение]. В случае необходимости остановите функцию изменения при помощи команды меню **Variable > Modify** [Переменная > Изменить]. Если около команды меню Modify [Изменить] нет видимой метки, то изменение деактивировано.

---

4. Введите фиксированные значения для переменных, которые вы хотите изменить, в столбец "Modify Value [Задаваемое значение]" таблицы.
5. Запустите функцию изменения с помощью команды меню **Variable > Modify [Переменная > Изменить]**. Метка рядом с этой командой показывает, что изменение началось.
6. Если вы хотите назначить новые значения, определить новый тип запуска или остановить изменение переменных, то деактивируйте функцию изменения, выбирая снова команду меню **Variable > Modify [Переменная > Изменить]**.
   Чтобы определить новый тип запуска, начните снова с шага 3.
   Чтобы назначить новые значения, начните снова с шага 4.

30.1.12 Однократное и немедленное изменение переменных

Действуйте следующим образом:

1. При помощи команды меню **Table > Open [Таблица > Открыть]** откройте таблицу переменных (VAT), содержащую переменные, которые вы хотите изменить, или активизируйте окно, содержащее требуемую таблицу переменных.
2. Выберите команду меню **PLC > Connect To > ... [ПЛК > Соединить с > ...]**, чтобы установить соединение с требуемым CPU, так чтобы вы могли изменить переменные активной таблицы переменных.

3. Введите фиксированные значения для переменных, которые вы хотите изменить, в столбец "Modify Value [Задаваемое значение]".
4. Выберите команду меню **Variable > Activate Modify Values [Переменная > Активизировать задаваемые значения]**.

30.1.13 Изменение: Инициализируйте CPU в режиме STOP предварительно установленными значениями

Действуйте следующим образом:

1. Выберите команду меню **PLC > Connect To > ... [ПЛК > Соединить с > ...]**, чтобы установить соединение с требуемым CPU.
2. Откройте диалоговое окно "Operating Mode [Режим работы]" с помощью команды меню **PLC > Operating Mode [ПЛК > Режим работы]** и переключите CPU в режим STOP.
3. Введите требуемые задаваемые значения переменных в таблицу переменных.
4. Активизируйте задаваемые значения, используя команду меню **Variable > Activate Modify Values** [Переменная > Активизировать задаваемые значения].
5. Откройте диалоговое окно "Operating Mode [Режим работы]" с помощью команды меню **PLC > Operating Mode** [ПЛК > Режим работы] и переключите CPU в режим RUN.

30.1.14 Изменение периферийных выходов, когда CPU находится в состоянии STOP

Функция "enable peripheral outputs [разблокировка периферийных выходов]" деактивирует блокировку вывода на периферийные выходы (PQ). Это дает вам возможность изменять периферийный выход, когда CPU находится в состоянии STOP.

Действуйте следующим образом:

1. При помощи команды меню **Table > Open [Таблица > Открыть]** откройте таблицу переменных (VAT), содержащую выходы, которые вы хотите изменить, или активизируйте окно для соответствующей таблицы переменных.
2. Выберите команду меню **PLC > Connect To > ... [ПЛК > Соединить с > ...]**, чтобы установить соединение с требуемым CPU, так что вы сможете изменить выходы в активной таблице переменных.
3. Откройте диалоговое окно "Operating Mode [Режим работы] с помощью команды меню **PLC > Operating Mode [ПЛК > Режим работы]** и переключите CPU в состояние STOP.
4. Введите для периферийных выходов, которые вы хотите изменить, соответствующие значения в столбец ”Modify Value [Задаваемое значение]".
   Примеры: PQB 7 задаваемое значение: 2#00010011
   PQW 2 W#16#0027
   PQD 4 DW#16#00000001
5. Используя команду меню **Variable > Enable Peripheral Output [Переменная > Разблокировать периферийный выход]**, включите режим "Enable Peripheral Output [Разблокировать периферийный выход]".
6. Используйте команду меню **Variable > Activate Modify Values [Переменная > Активизировать задаваемые значения]**, чтобы изменить периферийные выходы.
7. Режим "Enable Peripheral Output [Разблокировать периферийный выход]" остается активным до тех пор, пока вы снова не выберете команду меню **Variable > Enable Peripheral Output [Переменная > Разблокировать периферийный выход]**, чтобы выключить эту функцию.
8. Чтобы назначить новые значения, начните снова с шага 4

Примечание

Команда меню **Variable > Enable Peripheral Output [Переменная > Разблокировать периферийный вывод]** уместна только в режиме STOP.

Выход из режима "Enable Peripheral Output **[Переменная > Разблокировать периферийный выход]**" происходит после следующих событий:

- CPU изменяет свой режим работы (отображается сообщение)
- снова вызывается команда меню **Variable > Enable Peripheral Output [Переменная > Разблокировать периферийный выход]** или нажимается клавиша ESC (сообщение не отображается).

Прерывание выполнения с помощью ESC

Если вы нажимаете ESC, в то время, когда функция "Enable Peripheral Output [Разблокировать периферийный выход]" активна, то эта функция заканчивается без запроса.

30.1.15 Отображение окна принудительно установленных значений

1. Используя команду меню **Table > Open [Таблица > Открыть]**, откройте таблицу переменных (VAT) или активизируйте окно, содержащее соответствующую таблицу переменных.
2. Используйте команду меню **PLC > Connect To > ... [ПЛК > Соединить с > ...]**, чтобы установить соединение с требуемым CPU.
3. Используйте команду меню **Variable > Display Force Values [Переменная > Отобразить принудительно установленные значения]**, чтобы открыть окно "Force Values [Принудительно установленные значения]", в котором отображается текущее состояние выбранного CPU. Результат:
   - Команды меню принудительной установки можно выбирать только тогда, когда окно "Force Values [Принудительно установленные значения]" активно.
   - Это окно пустое, если в текущий момент времени нет активного задания на принудительную установку.
   - Если задание на принудительную установку уже активно, то переменные вместе с соответствующими принудительно установленными значениями отображаются жирным шрифтом. Если вы не запускали какое-либо из этих существующих заданий на принудительную установку, то свяжитесь с тем, кто уже сделал это, прежде чем продолжать дальше.

30.1.16 Настройка задания на принудительную установку

1. В столбце "Address [адрес]" введите переменные, которые вы хотите принудительно установить.
2. В столбце "Force Value [Принудительно установленное значение]" введите значения, которые вы хотите назначить переменным.
3. Запустите принудительную установку с помощью команды меню **Variable > Force** [Переменная > Принудительно установить]. Результат:
   - Если в текущий момент времени нет активного задания на принудительную установку, то переменным назначаются принудительно установленные значения.
   - Если задание на принудительную установку уже активно, то вы должны решить, хотите ли вы заменить существующее задание на принудительную установку.
     Если вы не запустили какое-либо из этих существующих заданий на принудительную установку, то свяжитесь с тем, кто уже начал, прежде чем заменить задание.

30.1.17 Удаление задания на принудительную установку

1. Вы можете завершить задание на принудительную установку с помощью команды меню **Variable > Stop Forcing [Переменная > Прекратить принудительную установку]**. Если Вы не запускали какое-либо из этих существующих заданий на принудительную установку, то свяжитесь с тем, кто запустил, прежде чем его завершить. Закрытие окна "Force Values [Принудительно установленные значения]" или выход из приложения "Monitoring and Modifying Variables [Наблюдение и изменение переменных]" не удаляют принудительно установленные значения в CPU.
2. Чтобы назначить новые значения, начните снова с пункта "Настройка задания на принудительную установку".

30.2 Как тестировать в режиме Статус программы

30.2.1 Открытие блока online

Действуйте следующим образом:

При управлении проектом

1. Откройте окно проекта online в SIMATIC Manager.
2. Выберите в окне online папку "Blocks [Блоки]". Отображается список загруженных блоков.
3. Дважды щелкните по блоку, который хотите открыть.

Без управления проектом

4. В SIMATIC Manager щелкните по кнопке "Accessible Nodes [Доступные узлы]" на панели инструментов или выберите команду меню **PLC > Display Accessible Nodes [ПЛК > Отобразить доступные узлы]**.
5. Выберите из отображаемого списка узел (объект ”MPI=...") и откройте папку "Blocks [Блоки]", чтобы отобразить блоки.
6. Дважды щелкните по блоку, который хотите открыть.

30.2.2 Установка отображения статуса программы

Вы можете самостоятельно установить отображение статуса программы в списке операторов, функциональном плане или в контактном плане.

Для установки такого отображения действуйте следующим образом:

1. Выберите команду меню **Options > Customize** [Параметры > Настроить].
2. В диалоговом окне "Customize [Настройка]" выберите вкладку "STL (AWL)" или вкладку "LAD/FBD (KOP/FUP)".
3. Выберите требуемые опции для тестирования программы. Вы можете отображать следующие поля состояния.

Активизировать ...	**... чтобы отобразить**
Status bit [бит состояния]	Бит состояния; бит 2 слова состояния
RLO (VKE)	Бит 1 слова состояния; показывает результат логической операции или математического сравнения
Standard status [стандартное состояние]	Содержимое слова таймера, слова счетчика или аккумулятора 1, если соответствующие команды появляются в операторе
Address register 1/2 [адресный регистр 1/2]	Содержимое соответствующего адресного регистра при регистровой косвенной адресации (внутрисегментной или межсегментной)
Akku2	Содержимое аккумулятора 2
DB register 1/2 [регистр DB 1/2]	Содержимое регистра блоков данных, первого и/или второго открытого блока данных
Indirect [косвенная]	Косвенная ссылка через память; ссылка через указатель (адрес), ссылка не через содержимое адреса; только для косвенной адресации через память, невозможно при регистровой косвенной адресации
Status word [слово состояния]	Все биты состояния слова состояния

30.2.3 Установка среды вызова для блока

Вы можете задать точные условия для регистрации статуса программы, устанавливая среду вызова. После этого статус программы регистрируется только тогда, когда выполняются установленные условия запуска.

Чтобы настроить отображение, действуйте следующим образом

1. Выберите команду меню Debug > Call Environment [Отладка > Среда вызова].
2. Установите в диалоговом окне условия запуска и подтвердите их посредством "OK".

Опция	Значение
No condition [безусловно]	Среда вызова тестируемого блока не имеет значения. Однако если вы вызываете один и тот же блок в различных точках программы, то вы не можете различить, для которого вызова отображается состояние.
Call path [путь вызова]	Здесь вы можете задать путь вызова, по которому тестируемый блок должен вызываться, чтобы активизировать регистрацию состояния. Вы можете вводить последние три уровня вызова перед достижением тестируемого блока.
Open data blocks [открытые блоки данных]	Здесь внешние условия вызова задаются указанием имени одного или двух блоков данных. Состояние регистрируется, если тестируемый блок вызывался с указанными блоками данных.

30.2.4 Установка режима для тестирования

Предпосылка

1. Тестируемый логический блок должен быть открыт online.
2. Установлена среда вызова блока (команда меню **Debug > Call Environment [Отладка > Среда вызова]**).

Последовательность действий

1. Отобразите установленную среду тестирования, используя команду меню **Debug > Operation [Отладка > Режим]**.
2. Выберите требуемый режим работы. Вы можете выбирать между режимом тестирования (test operation) и режимом обработки (process operation).

Режим работы	Объяснение
Режим тестирования	Возможны все функции тестирования без ограничения. Может существенно увеличиться время цикла сканирования CPU, например, вследствие того, что состояние команд в программных циклах регистрируется в каждом цикле.
Режим обработки	Тестовая функция "Статус программы" ограничивается таким образом, чтобы обеспечивалась минимально возможная нагрузка на время цикла сканирования. Например, это означает, что никакие условия вызова не разрешены. Отображение состояния запрограммированного цикла прерывается в точке возврата. Тестовые функции HOLD и пошаговое выполнение программы невозможны.

Примечание

Если режим работы устанавливался тогда, когда вы назначали параметры CPU, то вы можете изменить режим только путем изменения параметров. В противном случае вы можете изменять режим в отображаемом диалоговом окне.

30.2.5 Изменение переменных в статусе программы

Требование: Открыт блок online.

Действия, описанные ниже, оказывают такое действие, что выбранные переменные изменяются однократно и немедленно.

Изменение переменных типа BOOL

1. Выберите адрес, который вы хотите изменить.
2. Выберите команду меню **Debug > Modify Address to 1 [Отладка > Установить адрес в 1]** или **Debug > Modify Address to 0 [Отладка > Установить адрес в 0]**.

Изменение не булевских переменных

1. Выберите адрес, который вы хотите изменить.
2. Выберите команду меню **Debug > Modify [Отладка > Изменить]**.
3. В диалоговом окне введите значение, которое переменная должна принять (modify value [задаваемое значение]).
4. Закройте диалоговое окно.

Альтернативная процедура

1. Расположите курсор на адресе, который вы хотите изменить.
2. Нажмите правую кнопку мыши и выберите из всплывающего меню соответствующую команду изменения.

30.2.6 Активизация и деактивизация теста с помощью статуса программы

1. Запустите регистрацию состояния программы при помощи команды меню **Debug > Monitor [Отладка > Наблюдать]** (около команды меню появляется метка).
2. Проанализируйте отображаемый в форме таблицы статус программы на AWL блока.
3. Отображение статуса программы можно скрыть, снова выбирая команду меню **Debug > Monitor [Отладка > Наблюдать]** (так, чтобы метка исчезла).

30.3 Этапы тестирования с использованием контрольных точек

30.3.1 Тестирование с использованием контрольных точек

Перед началом тестирования обеспечьте, чтобы CPU был в режиме RUN или RUN-P и чтобы тестируемый блок был сохранен и загружен в CPU.

1. Откройте блок, который будет тестироваться, online.
2. Отобразите установленную среду тестирования, используя команду меню **Debug > Operation [Отладка > Режим]**. Если режим работы устанавливался тогда, когда вы назначали параметры CPU, то вы можете изменить режим только путем изменения параметров. В противном случае вы можете изменять режим в отображаемом диалоговом окне.
   - Режим тестирования (Test operation)
   - Режим обработки (Process operation)
3. Активизируйте панель контрольных точек, используя команду меню **View > Breakpoint Bar [Вид > Панель контрольных точек]**.
4. Расположите курсор в строке команды, в которой вы хотите установить контрольную точку.
5. Установите контрольную точку, используя команду меню **Debug > Set Breakpoint [Отладка > Установить контрольную точку]** или соответствующую кнопку на панели контрольных точек. Операторная строка отмечается пустым кружком.
6. Активизируйте контрольную точку при помощи команды меню **Debug > Breakpoints Active [Отладка > Контрольные точки активны]**. Тогда контрольная точка отмечается заполненным кружком.
7. Переключите программируемый контроллер в режим RUN-P.
8. Когда в программе встречается контрольная точка, программируемый контроллер входит в режим HOLD. Контрольная точка отмечается стрелкой. В окне, которое может быть расположено на экране где угодно, отображается содержимое регистров.
9. Чтобы продолжить выполнение программы до следующей контрольной точки, выберите команду меню **Debug > Resume [Отладка > Возобновить]** или тестируйте в пошаговом режиме, используя команду **Debug > Execute Next Statement [Отладка > Выполнить следующий оператор]**.
10. Вы можете использовать команду меню **Debug > Delete Breakpoint [Отладка > Удалить контрольную точку]**, чтобы удалять контрольные точки индивидуально, либо вы можете удалить все контрольные точки, используя команду меню Debug > Delete All Breakpoints **[Отладка > Удалить все контрольные точки]**.

Примечание

Чтобы просматривать последовательность вызовов, выберите в SIMATIC Manager команду меню **PLC > Module Information [ПЛК > Информация о модулях]**.

Чтобы отобразить следующую контрольную точку, выберите команду меню **Debug > Show Next Breakpoint [Отладка > Показать следующую контрольную точку]**. Курсор переходит на следующую выбранную контрольную точку без обработки блока.

При загрузке в программируемый контроллер блоки, имеющие контрольные точки, в программируемом контроллере отвергаются. Вы можете загружать их только после того, как удалите контрольные точки.

30.3.2 Поиск и удаление контрольных точек

Если вы установили в вашей программе S7 максимальное количество контрольных точек и хотите вставить другую контрольную точку, то вы должны вначале удалить одну из них:

1. Проведите в вашей программе S7 сквозной поиск, используя команду меню Debug > Show Next Breakpoint [Отладка > Показать следующую контрольную точку].
2. Удалите контрольную точку, которая вам больше не требуется, используя команду меню **Debug > Delete Breakpoint [Отладка > Удалить контрольную точку]**.

30.3.3 Тестирование в пошаговом режиме

Действуйте следующим образом:

1. Установите контрольную точку перед строкой команды, начиная с которой вы хотите тестировать вашу программу в пошаговом режиме.
2. Выберите команду меню **Debug > Breakpoints Active [Отладка > Контрольные точки активны]**.
3. Выполните программу до этой контрольной точки
4. Чтобы выполнить следующий оператор, выберите команду меню **Debug > Execute Next Statement [Отладка > Выполнить следующий оператор]** или используйте соответствующую кнопку на панели инструментов.
   - Если следующим оператором является вызов блока, то вызов обрабатывается, и программа переходит на следующий оператор после вызова блока.
   - Используйте команду меню **Debug > Execute Call [Отладка > Выполнить вызов]**, чтобы перейти к блоку. Там вы можете либо продолжать тестирование в пошаговом режиме, либо установить контрольные точки. В конце блока программа возвращается к следующему оператору после вызова блока.

Примечание

Чтобы просматривать последовательность вызовов, выберите в SIMATIC Manager команду меню **PLC > Module Information [ПЛК > Информация о модуле]**.

30.3.4 Прекращение тестирования с помощью контрольных точек

Чтобы возвратиться к нормальной обработке программы, действуйте следующим образом:

1. Выберите либо команду меню Debug > Breakpoints Active [Отладка > Контрольные точки активны] (метка перед командой меню исчезает), либо команду меню Debug > Delete All Breakpoints [Отладка > Удалить все контрольные точки].
2. Выберите команду меню **Debug > Resume [Отладка > Возобновить]**.

   Результат: Обработка программы возобновляется, и CPU переходит в режим RUN.

31 Работа с диагностикой

31.1 Установка отображения (быстрый обзор или диагностический обзор)

При запуске приложения "Diagnosing Hardware [Диагностика аппаратных средств]" вы можете определить, должен ли отображаться быстрый обзор с краткой информацией или диагностический обзор HW Config с подробной информацией.

Отображение диагностического обзора занимает больше времени, потому что STEP 7 должен найти и отобразить данные. Поэтому при запуске приложения "Diagnosing Hardware [Диагностика аппаратных средств]" обычно выгодней отображать быстрый обзор (установлено по умолчанию). Потом в случае необходимости вы можете открывать диагностический обзор при помощи кнопки "Open Station Online [Открыть станцию online]".

Чтобы выбрать отображение, действуйте следующим образом:

1. Выберите в SIMATIC Manager команду меню **Options > Customize** [Параметры > Настроить].
2. Выберите вкладку "View [Вид]".
3. Активизируйте или деактивизируйте опцию "Display quick view when diagnosing hardware [Отобразить быстрый обзор при диагностировании аппаратных средств]".
4. Щелкните по кнопке "OK".

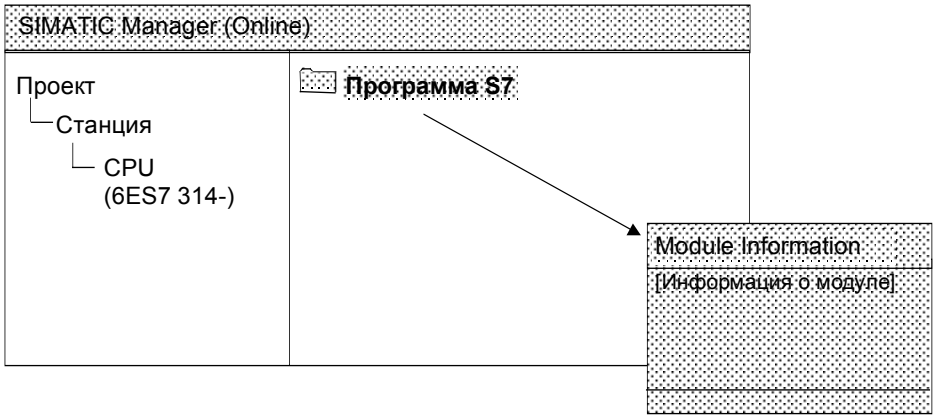
В зависимости от конфигурации станции, STEP 7 может затрачивать значительно больше времени на отображение диагностического обзора по сравнению с быстрым обзором.

31.2 Вызов информации о модулях

31.2.1 Вызов информации о программируемом модуле

В окне проекта в SIMATIC Manager

1. Откройте проект.
2. Выделите станцию и откройте ее двойным щелчком.
3. Выделите модуль или папку "S7 Program [Программа S7]" в станции.
4. Выберите команду меню **PLC > Module Information** [ПЛК > Информация о модуле].



В окне "Accessible Nodes" в SIMATIC Manager

Выполните следующие шаги:

1. Откройте в SIMATIC Manager окно "Accessible Nodes [Доступные узлы]", используя команду меню **PLC > Display Accessible Nodes** [ПЛК > Отобразить доступные узлы].
2. Выберите узел в окне "Accessible Nodes [Доступные узлы]".

Примечание

В окне "Accessible Nodes [Доступные узлы]" модули всегда видны только со своим собственным адресом узла (адресом MPI или PROFIBUS).

---

3. Затем выберите команду меню **PLC > Module Information [ПЛК > Информация о модуле]**.

Результат

В обоих случаях отображается диалоговое окно "Module Information [Информация о модуле]". В зависимости от возможностей диагностики модуля в диалоговом окне "Module Information [Информация о модуле]" отображается переменное количество вкладок. Вкладка “General [Общая]" отображается для каждого модуля.

31.2.2 Запрос информации для любого модуля

1. Вызовите диагностический обзор HW Config (вызов диагностического обзора аппаратной конфигурации).
   Диагностический обзор показывает конфигурацию станции так, как она определяется из модулей (например, CPU). Состояние модулей обозначается посредством символов. Неисправные модули и отсутствующие сконфигурированные модули перечисляются в отдельном диалоговом окне. Из этого диалогового окна вы можете перейти непосредственно к одному из выбранных модулей (кнопка ”Go To [Перейти]”).

Примечание

Для получения улучшенного обзора станций со многими модулями вы можете выбрать команду меню **PLC > Display Faulty Modules [ПЛК > Отобразить неисправные модули]**. Тогда в диалоговом окне отображается список неисправных модулей. Выбирая требуемый модуль и щелкая по кнопке "Module Information [Информация о модуле]", вы можете открыть диалоговое окно "Module Information [Информация о модуле]" с закладками.

2. Выделите в конфигурационной таблице модуль и выберите команду меню **PLC > Module Information [ПЛК > Информация о модуле]** или дважды щелкните по соответствующему модулю.

Результат: В обоих случаях отображается диалоговое окно "Module Information [Информация о модуле]". В зависимости от возможностей диагностики модуля в диалоговом окне "Module Information [Информация о модуле]" отображается переменное количество вкладок. Вкладка “General [Общая]" отображается для каждого модуля.

31.3 Открытие блока, соответствующего записи в диагностическом буфере или стеке

31.3.1 Открытие блока, соответствующего записи в диагностическом буфере

С помощью записей диагностического буфера, ссылающихся на местоположение ошибки (тип блока, номер блока, относительный адрес), вы можете открыть блок, вызвавший событие, чтобы устранить причину ошибки.

1. Выберите в верхнем списке диагностическое событие.
2. Щелкните по кнопке "Open Block [Открыть блок]". Блок открывается в соответствующем редакторе (например, Statement List [Список операторов]) с курсором, указывающим на точку в программе, которая вызвала ошибку.
3. Исправьте ошибку в блоке.

Примечание

Диагностический буфер хранит все диагностические события в пределах своей максимальной емкости. Все события в буфере сохраняются, даже если загружается другая программа пользователя.

Поэтому возможно, что более старые входные сообщения буфера диагностики могут ссылаться на блоки, больше не присутствующие в CPU. В наихудшем случае в CPU, возможно, имеется новый блок с тем же самым именем, который, однако, не порождал диагностического сообщения.

Изредка могут встречаться следующие ситуации:

Диагностическое событие более старое, чем дата последнего изменения блока:

- Диалоговое окно ”Open Block [Открыть блок]" появляется вместе с сообщением о том, что блок был изменен. Это может также означать, что блок просто является блоком с тем же самым именем, но относящимся к другой программе.
- Вы все же можете открыть этот блок online в CPU и редактировать его в случае необходимости или
- Вы можете выбрать этот блок offline в подходящей программе и редактировать его offline.

Блок, породивший событие, больше не находится в CPU:

- Диалоговое окно ”Open Block [Открыть блок]" появляется вместе с сообщением о том, что упомянутый блок не существует в CPU. Этот блок был удален после момента ввода диагностического события.

Вы можете выбрать этот блок offline в подходящей программе и редактировать его offline.

31.3.2 Открытие блока из списка B-стека

Действуйте следующим образом:

1. Щелкните по кнопке "Open Block [Открыть блок]". Блок открывается в редакторе программ. Курсор указывает на место в программе, где возобновится обработка после перехода в вызванный блок.
2. Сделайте свои изменения.

31.3.3 Открытие блока из списка I-стека

Действуйте следующим образом:

1. Щелкните по кнопке "Open Block [Открыть блок]". Блок открывается в редакторе текстов программ. Курсор указывает на место в программе, которое вызвало ошибку.
2. Сделайте свои изменения.

32 Как распечатывать и архивировать

32.1 Как печатать

32.1.1 Печать блоков и исходных файлов на AWL

Чтобы напечатать содержимое блоков или исходных файлов на AWL, действуйте следующим образом:

1. В окне проекта SIMATIC Manager откройте блок или исходный файл, содержимое которого вы хотите напечатать, дважды щелкнув по нему.
2. В окне "LAD/FBD/STL (KOP/FUP/AWL)" выберите команду меню **File > Print [Файл > Печатать]**.
3. Отображается диалоговое окно "Print [Печать]". В диалоговом окне выполните нужную вам настройку параметров.
4. Щелкните по кнопке "OK".

Печатается содержимое блока или исходного файла.

Примечание

Представление, установленное для отображения в окне редактора, учитывается при печати. Логические блоки печатаются на отображаемом в текущий момент времени языке представления (KOP, FUP или AWL), блоки данных печатаются в текущем представлении (представление описаний или представление данных).

32.1.2 Печать информации о модуле

Чтобы напечатать информацию о модуле, действуйте следующим образом:

1. Выберите в SIMATIC Manager команду меню **View > Online [Вид > online]**.
2. Отображается окно проекта online в SIMATIC Manager. В этом окне проекта щелкните по объекту "станция" (например, SIMATIC 300).
3. Выберите команду меню **PLC > Module Information [ПЛК > Информация о модуле]**.
4. Отображается диалоговое окно "Module Information [Информация о модуле]". Выберите требуемую вкладку (например, "Diagnostic Buffer [Диагностический буфер]").
5. Щелкните по кнопке "Print [Печатать]".
6. Отображается диалоговое окно "Print [Печать]". В диалоговом окне выполните нужную вам настройку параметров.
7. Щелкните по кнопке "OK".

Печатается вкладка, отображенная в текущий момент времени в диалоговом окне "Module Information [Информация о модуле]".

32.1.3 Печать таблицы глобальных данных

Чтобы напечатать таблицу глобальных данных, действуйте следующим образом:

1. Откройте сетевой объект (например, MPI) в окне проекта SIMATIC Manager.
2. Отображается окно "NetPro: Configuring Networks [NetPro: конфигурирование сетей]" с конфигурацией сети и таблицей соединений. Выберите в конфигурации сети требуемое сетевое соединение (например, CPU). Сконфигурированные данные вводятся в таблицу глобальных данных.
3. Выберите команду меню **Network > Print [Сеть > Печатать]**.
4. Отображается диалоговое окно "Customize [Настройка]". В диалоговом окне выполните нужную вам настройку параметров. (Вы можете получить описание параметров, нажимая клавишу F1, пока курсор находится в диалоговом окне.)
5. Щелкните по кнопке "OK".

Печатается таблица глобальных данных.

32.1.4 Печать конфигурационной таблицы

Чтобы напечатать конфигурационную таблицу, действуйте следующим образом:

1. В окне проекта SIMATIC Manager дважды щелкните по папке объектов для станции, например, "SIMATIC 300(1)".
2. Откройте объект "Hardware [аппаратные средства]", дважды щелкнув на нем.
3. Выберите команду меню **Station > Print** [Станция > Печатать] в окне "HW Config - Configuring Hardware [HW Config - Конфигурирование аппаратных средств]".
4. Отображается диалоговое окно "Print [Печать]. В диалоговом окне выполните нужную вам настройку параметров. (Вы можете получить описание параметров, нажимая клавишу F1, пока курсор находится в диалоговом окне.)
5. Щелкните по кнопке "OK".

Печатается конфигурационная таблица.

32.1.5 Печать сообщений

Чтобы напечатать сообщения (например, для отчета о сообщениях), действуйте следующим образом:

1. В окне проекта SIMATIC Manager щелкните по объекту, для которого вы хотите печатать сообщения, и выберите команду меню **File > Print > Messages [Файл > Печатать > Сообщения]**.
2. В диалоговом окне "Print Messages [Напечатать сообщения]" выберите требуемые типы сообщений.
3. Щелкните по кнопке "OK".

Печатаются сообщения.

32.1.6 Печать списков текстов пользователя

Тексты пользователя – это все тексты и сообщения, которые могут выводиться на устройствах отображения. Чтобы напечатать эти тексты, действуйте следующим образом:

1. В SIMATIC Manager выберите объект (проект, программу S7, папку блоков, блок или таблицу символов), чьи тексты пользователя вы хотите напечатать, и выберите команду меню **Options > Translate Texts [Параметры > Перевести тексты]**.
2. В окне "Translating Texts [Перевод текстов]" выберите команду меню **File > Print [Файл > Печатать]** или щелкните по соответствующей кнопке на панели инструментов. Открывается диалоговое окно ”Print [Печать]".
3. Установите свои параметры для печати.
4. Запустите процесс печати с помощью "OK".

Список текстов пользователя распечатывается на заданном по умолчанию принтере. Если распечатка содержит более одной страницы, то после номера страницы в нижнем правом углу страницы печатаются две точки. Последняя страница не имеет этих точек, что указывает на отсутствие последующих страниц.

32.1.7 Печать дерева объектов

Чтобы напечатать дерево объектов, действуйте следующим образом:

1. В окне проекта SIMATIC Manager щелкните по папке объектов.
2. Выберите команду меню **File > Print > Object List [Файл > Печатать > Список объектов]**.
3. Появляется диалоговое окно "Print Object List [Печать списка объектов]". Выберите опцию "Tree window [Окно дерева]".
4. Выберите опцию "All [Все]", если вы хотите напечатать структуру всего дерева, или опцию "Selection [Выбор]", если вы хотите печатать дерево объектов, идущее вниз от выбранной папки объектов.
5. В диалоговом окне выполните нужную вам настройку параметров.
6. Щелкните по кнопке "OK".

Печатается дерево объектов.

32.1.8 Печать объектов

Чтобы печатать объекты, действуйте следующим образом:

1. В окне проекта SIMATIC Manager щелкните по объекту, который вы хотите напечатать.
2. Выберите команду меню **File > Print > Object** [Файл > Печатать > Объект].
3. Печатается содержимое объекта.

32.1.9 Печать списка объектов

Вы можете печатать списки объектов двумя разными способами.

Печать списков объектов через папку объектов

Действуйте следующим образом:

1. В окне проекта SIMATIC Manager щелкните по папке объектов, объекты которой вы хотите напечатать в списке (например, блоки).
2. Выберите команду меню **File > Print > Object List [Файл > Печатать > Список объектов]**.
3. Появляется диалоговое окно "Print Object List [Печать списка объектов]". В диалоговом окне выполните нужную вам настройку параметров.
4. Щелкните по кнопке "OK".

Печатается список объектов.

Печать списков объектов через объект

Действуйте следующим образом:

1. В окне проекта SIMATIC Manager щелкните по объекту (например, FB1), для которого вы хотите печатать список всех объектов того же самого типа (например, блоки).
2. Выберите команду меню **File > Print > Object List [Файл > Печатать > Список объектов]**.
3. Появляется диалоговое окно "Print Object List [Печать списка объектов]". Выберите опцию "All [Все]".
4. В диалоговом окне выполните нужную вам настройку параметров.
5. Щелкните по кнопке "OK".

Печатается список объектов.

32.1.10 Печать справочных данных

Чтобы напечатать справочные данные, действуйте следующим образом:

1. В окне проекта SIMATIC Manager щелкните по папке "Blocks [Блоки]".
2. Выберите команду меню **Options > Reference Data > Display [Параметры > Справочные данные > Отобразить]** или команду меню **Options > Reference Data > Filter [Параметры > Справочные данные > Фильтровать]**, если вы хотите отфильтровать справочные данные перед тем, как печатать. Во втором случае выполните в диалоговом окне "Filter [Фильтр]" нужную вам настройку параметров и щелкните по кнопке "OK". (Вы можете получить описание параметров, нажимая клавишу F1, пока курсор находится в диалоговом окне.)
3. Открывается окно "Displaying Reference Data [Отображение справочных данных]" со списком справочных данных, определенных с использованием фильтра. Выберите команду меню **Reference Data > Print [Справочные данные > Печатать]**.
4. Отображается диалоговое окно "Print [Печать]". В диалоговом окне выполните нужную вам настройку параметров.
5. Щелкните по кнопке "OK".

Когда вы печатаете справочные данные, то печатается список в активном окне. При печати учитывается настройка параметров фильтра, выполненная вами для отображения в этом активном окне.

32.1.11 Печать таблицы символов

Чтобы напечатать таблицу символов, действуйте следующим образом:

1. В окне проекта SIMATIC Manager откройте объект "Symbols [Символы]", дважды щелкнув по нему. В окне "Symbol Editor [Редактор символов]" выберите команду меню **Symbol Table > Print [Таблица символов > Печатать]**.
2. Отображается диалоговое окно "Print [Печать]". В диалоговом окне выполните нужную вам настройку параметров.
   Если вы выберете опцию "Optimized printout [Оптимизированная распечатка]", то длинные строки не усекаются, а продолжаются на следующей строке, а размер шрифта подгоняется под формат страницы.
   Вы можете также выбирать между следующими вариантами:
   - Чередующееся выделение строк.
     Строки таблицы легче читать, потому что они изображаются различными оттенками серого цвета.
   - Использование разделителей между строками.
     Строки таблицы легче читать, потому что они разделены линиями.
3. Щелкните по кнопке "OK".

Печатается таблица символов.

32.1.12 Печать таблицы переменных

Чтобы напечатать таблицу переменных, действуйте следующим образом:

1. В окне проекта SIMATIC Manager откройте объект "VAT", дважды щелкнув по нему.
2. В окне "Monitoring and Modifying Variables [Наблюдение и изменение переменных]" выберите команду меню **Table > Print [Таблица > Печатать]**.
3. Отображается диалоговое окно "Print [Печать]". В диалоговом окне выполните нужную вам настройку параметров.
4. Щелкните по кнопке "OK".

Печатается таблица переменных.

32.1.13 Печать таблицы соединений

Чтобы напечатать таблицу соединений, действуйте следующим образом:

1. В окне проекта SIMATIC Manager откройте объект "Connections [Соединения]", дважды щелкнув по нему.
2. Отображается окно "NetPro: Configuring Networks [NetPro: конфигурирование сетей]" с конфигурацией сети и таблицей соединений. Выберите в конфигурации сети требуемый программируемый модуль (например, CPU). В таблицу соединений вводятся соответствующие данные.
3. Выберите команду меню **Network > Print [Сеть > Печатать]**.
4. Отображается диалоговое окно "Customize [Настройка]". В диалоговом окне выполните нужную вам настройку параметров. (Вы можете получить описание параметров, нажимая клавишу F1, пока курсор находится в диалоговом окне.)
5. Щелкните по кнопке "OK".

Печатается таблица соединений.

32.2 Архивирование/Извлечение

32.2.1 Установка предпочитаемой вами программы архивирования

Чтобы установить программу архивирования, действуйте следующим образом:

1. Выберите в SIMATIC Manager команду меню **Options > Customize [Параметры > Настроить]**. Отображается диалоговое окно.
2. На странице с закладками ”Archive [Архивирование]" выберите предпочитаемую вами программу архивирования.

По умолчанию действует программа архивирования *PKZIP 2.50*.

32.2.2 Установка маршрута поиска для программ архивирования

Стандартная конфигурация STEP 7 предполагает, что программы архивирования установлены в маршруте поиска DOS. Если программы архивирования установлены в другом месте, то действуйте следующим образом:

1. Выберите в SIMATIC Manager команду меню **Options > Customize [Параметры > Настроить]**. Отображается диалоговое окно.
2. Используя кнопку ”Configure [Конфигурировать]" на странице с закладками ”Archive [Архивирование]", откройте диалоговое окно ”Configure Archive [Конфигурирование архива]".
3. Введите имя маршрута программы архивирования в поле "Program path [Маршрут программы]" или выберите маршрут с помощью кнопки "Browse [Просмотреть]".
4. Закройте диалоговые окна с помощью ”OK".

32.2.3 Установка задаваемого по умолчанию целевого каталога

Вы можете установить целевой каталог, в котором хранится созданный архив, и каталог, в котором хранится извлеченный архив. Это экономит время на вводе каталогов во время архивирования или извлечения.

Чтобы установить эти каталоги, действуйте следующим образом:

1. Выберите в SIMATIC Manager команду меню **Options > Customize [Параметры > Настроить]**.
2. В диалоговом окне выберите вкладку ”Archive [Архивирование]".
3. Активизируйте опцию ”Target Directory for Archiving [Целевой каталог для архивирования]" и/или ”Target Directory for Retrieving [Целевой каталог для извлечения]".
4. Введите путь в соответствующее текстовое поле или выберите каталог через кнопку "Browse [Просмотреть]".
5. Закройте диалоговое окно с помощью "OK".

32.2.4 Архивирование

Чтобы заархивировать проект или библиотеку, действуйте следующим образом:

1. Убедитесь, что не открыты никакие окна проектов, показывающие проект, который вы хотите архивировать, или что библиотека, которую вы хотите архивировать, закрыта.
2. Используйте команду меню **File > Archive [Файл > Архивировать]**.
3. В диалоговом окне "Archive [Архивирование]" выберите проект или библиотеку, которую вы хотите архивировать, и подтвердите ваш ввод.
4. В появляющемся диалоговом окне установите путь к целевому каталогу для архивного файла, имя файла и тип файла. Тип файла используется для того, чтобы определить, какая программа архивирования требуется (например, "ZIP" для программы архивирования PKZIP).
5. Можно открыть другое диалоговое окно, в котором вы можете настроить большее количество параметров архива. Затем открывается окно DOS , в котором можно наблюдать выполнение процесса архивирования. Проект/ библиотека сжимается и сохраняется в целевом каталоге.

32.2.5 Извлечение

Чтобы извлечь проект или библиотеку из архива, действуйте следующим образом:

1. Выберите команду меню File > Retrieve [Файл > Извлечь].
2. В появляющемся диалоговом окне выберите архивный файл, который содержит сжатый проект или библиотеку, и подтвердите ваш ввод.
3. В следующем диалоговом окне выберите целевой каталог, в который должны извлекаться данные.

Затем открывается окно DOS , в котором можно наблюдать выполнение процесса извлечения.

33 Когда несколько пользователей редактируют один и тот же проект

33.1 Установка конфигурации рабочей станции

Чтобы работать над проектом из разных рабочих станций STEP 7, вам нужно на каждой рабочей станции выполнить следующие установки параметров.

1. На панели запуска Windows выберите команду меню Start > Simatic > STEP 7 > Configure SIMATIC Workstation [Пуск > Simatic > STEP 7 > Конфигурировать рабочую станцию SIMATIC].
2. Выберите опцию "Multi-terminal system [Многотерминальная система]" и сетевой протокол, который вы хотите использовать.

33.2 Объединение нескольких программ S7 в одну

STEP 7 не поддерживает объединение программ S7 на рабочих станциях, не соединенных в сеть. Единственный способ объединения программ S7 в этом случае состоит в том, чтобы копировать отдельные блоки или исходные файлы. Совместно используемые данные для проекта, такие как таблица символов или таблица переменных, нужно редактировать вручную после копирования.

1. В программе S7 скопируйте блоки и исходные файлы в соответствующие им папки.
2. Экспортируйте таблицы символов отдельных программ S7 в формат ASCII и импортируйте их в таблицу символов объединенной программы S7.
3. Проверьте, не использовались ли символы более одного раза.

   Совет: Вы можете также интегрировать короткие таблицы символов, используя буфер обмена (копирование и вставка).
4. Скопируйте таблицы переменных, которые вы хотите использовать, или объедините разные таблицы переменных в новую таблицу переменных, используя буфер обмена (копирование и вставка).

33.3 Копирование программ S7 с атрибутами сообщений

Если вы назначили блокам атрибуты сообщений, то обратите внимание на то, что при копировании программ S7 диапазоны номеров сообщений могут перекрываться. Во избежание конфликтов:

- Используйте команду меню **Edit > Special Object Properties > Message Numbers [Редактировать > Специальные свойства объекта > Номера сообщений]**, чтобы выделить каждой программе S7 фиксированный диапазон номеров сообщений.
- Когда вы копируете программы S7, убедитесь, что они не перезаписывают другие программы S7.
- Обратите внимание на то, что только шаблоны сообщений (FB) можно программировать отдельно от программы S7.

A Приложение

A.1 Режимы работы

A.1.1 Режимы работы и переключения режимов

Режимы работы

Режимы работы описывают поведение CPU в текущий момент времени. Знание режимов работы CPU полезно при программировании запуска, тестировании контроллера и поиске неисправностей.

CPU 7-300 и S7-400 могут принимать следующие режимы работы:

- STOP [останов]"
- STARTUP [запуск]"
- RUN [выполнение (программы)]"
- HOLD [приостановка]"

CPU в состоянии STOP проверяет, существуют ли фактически все сконфигурированные модули или модули, заданные адресацией по умолчанию, и устанавливает входы/выходы в предварительно определенное начальное состояние. В состоянии STOP программа пользователя не выполняется.

В режиме STARTUP различают типы запуска "теплый рестарт", "холодный рестарт" и "горячий рестарт".

- При теплом рестарте обработка программы запускается с начала программы при исходных установках для системных данных и областей адресов пользователя (не сохраняемые таймеры, счетчики и память с побитовым доступом (меркеры) сбрасываются).
- При холодном рестарте считывается таблица входов образа процесса, и программа пользователя на языке STEP 7 обрабатывается, начиная с первой команды в OB1 (применяется также при теплом рестарте).

  Любые блоки данных, созданные SFC в рабочей памяти, удаляются; остальные блоки данных имеют предварительно установленное значение из загрузочной памяти.

  Образ процесса, все таймеры, счетчики и память с побитовым доступом сбрасываются независимо от того, назначались ли они как сохраняемые или нет.
- При горячем рестарте выполнение программы возобновляется в точке прерывания (таймеры, счетчики и память с побитовым доступом не сбрасывается). Горячий рестарт возможен только в CPU S7-400.

CPU в режиме RUN выполняет программу пользователя, обновляет входы и выходы, обслуживает прерывания и обрабатывает сообщения об ошибках.

В режиме HOLD обработка программы пользователя приостанавливается, и вы можете тестировать пользовательскую программу шаг за шагом. Режим HOLD возможен только тогда, когда вы тестируете, используя устройство программирования.

Во всех этих режимах CPU может обмениваться данными через многоточечный интерфейс (MPI).

Другие режимы работы

Если CPU не готов к работе, то он находится в одном из следующих режимов:

- «Выключен», иными словами, выключен источник питания.
- «Неисправен», иными словами, произошел отказ. Чтобы проверить, действительно ли CPU неисправен, переключите CPU в состояние STOP, выключите и снова включите источник питания. Если CPU запускается, то выведите на экран диагностический буфер, чтобы проанализировать проблему. Если CPU не запускается, то его нужно заменить.

Переключение режимов работы

Следующий рисунок показывает режимы работы и переключение режимов CPU в S7-300 и S7-400:

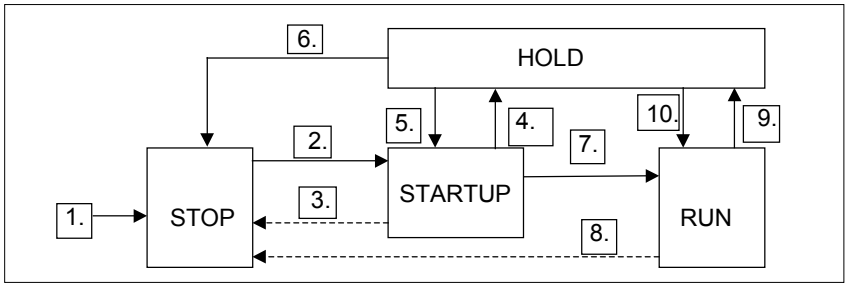


Таблица показывает условия, при которых режимы работы могут изменяться.

Переключение	Описание
1.	После того, как вы включили источник питания, CPU находится в состоянии STOP
2.	CPU переключается в режим STARTUP [запуск]": • После того, как CPU переключается в режим RUN или RUN-P с помощью переключателя режимов работы или устройства программирования. • После запуска, автоматически вызванного включением питания. • Если выполняется коммуникационная функция RESUME или START. В обоих случаях переключатель режимов работы должен устанавливаться в RUN или RUN-P.
3.	CPU переключается обратно в состояние STOP, если: • Во время запуска обнаруживается ошибка. • CPU переключается в STOP переключателем режимов работы или в устройстве программирования. • В OB запуска выполняется команда останова. • Выполняется коммуникационная функция STOP.
4.	CPU переключается в режим HOLD, когда в программе запуска достигается контрольная точка.
5.	CPU переключается в режим STARTUP, если в программе запуска была установлена контрольная точка и выполнена команда "EXIT HOLD" (тестовые функции).
6.	CPU переключается обратно в состояние STOP, если: • CPU переключается в состояние STOP с помощью переключателя режимов работы или в устройстве программирования. • Выполняется коммуникационная команда STOP.
7.	Если запуск успешный, то CPU переключается в режим RUN.
8.	CPU переключается обратно в состояние STOP, если: • В режиме RUN обнаружена ошибка, а соответствующий OB не загружен. • CPU переключается в состояние STOP с помощью переключателя режимов работы или в устройстве программирования. • В программе пользователя выполняется команда STOP. • Выполняется коммуникационная функция STOP.
9.	CPU переключается в режим HOLD, когда в программе пользователя достигается контрольная точка.
10.	CPU переключается в режим RUN, если была установлена контрольная точка и выполняется команда "EXIT HOLD".

Приоритет режима работы

Если одновременно запрашиваются несколько режимов работы, то выбирается режим работы с высшим приоритетом. Например, если переключатель режимов работы установлен в положение RUN и вы пытаетесь установить CPU в состояние STOP через устройство программирования, то CPU переключится в состояние STOP, потому что этот режим имеет высший приоритет.

Приоритет	Режим
высший	STOP
	HOLD
	STARTUP
низший	RUN

A.1.2 Состояние STOP

В состоянии STOP программа пользователя не выполняется. Все выходы устанавливаются на замещающие значения так, чтобы управляемый процесс находился в безопасном состоянии. CPU выполняет следующие проверки:

- Имеются ли какие-либо проблемы в аппаратных средствах (например, недоступные модули)?
- Должна ли применяться к CPU настройка по умолчанию или имеются наборы параметров?
- Удовлетворяются ли условия запрограммированного поведения при пуске?
- Имеются ли какие-либо проблемы в системном программном обеспечении?

CPU в состоянии STOP также может принимать глобальные данные, и возможна пассивная односторонняя связь с использованием коммуникационных SFB для конфигурированных соединений и коммуникационных SFC для не конфигурированных соединений.

Сброс памяти

Память CPU может быть сброшена в состоянии STOP. Память можно сбросить вручную, используя переключатель режимов работы (MRES), или из устройства программирования (например, перед загрузкой программы пользователя).

Сброс памяти CPU возвращает CPU в исходное состояние следующим образом:

- Происходит очистка рабочей памяти и загрузочной оперативной памяти (RAM) от всей программы пользователя и очистка всех адресных областей.
- Параметры системы и CPU и параметры модулей устанавливаются на значения, заданные по умолчанию. Параметры MPI, установленные до сброса памяти, сохраняются.
- Если подключена плата памяти (Flash EPROM), то CPU копирует программу пользователя из платы памяти в рабочую память (включая параметры CPU и модулей, если соответствующие конфигурационные данные находятся также на плате памяти).

Диагностический буфер, параметры MPI, время и счетчики рабочего времени не сбрасываются.

A.1.3 Режим STARTUP

Прежде чем CPU сможет начать обработку программы пользователя, должна быть выполнена программа запуска. Программируя OB запуска в вашей программе запуска, вы можете задать определенные параметры настройки для вашей циклической программы.

Имеются три типа запуска: теплый рестарт, холодный рестарт и горячий рестарт. Горячий рестарт возможен только в CPU S7-400. Он должен задаваться явным образом в наборе параметров CPU с помощью STEP 7.

Особенности режима STARTUP:

- Обрабатывается программа OB запуска (OB100 для теплого рестарта, OB101 для горячего рестарта, OB102 для холодного рестарта).
- Невозможно выполнение программы, управляемое временем или прерываниями.
- Таймеры обновляются.
- Счетчики рабочего времени начинают функционировать.
- Цифровые выходы в сигнальных модулях заблокированы (могут устанавливаться посредством прямого доступа).

Теплый рестарт

Теплый рестарт всегда разрешен, если система не затребовала сброс памяти. Теплый рестарт – это единственно возможный выбор после:

- сброса памяти
- загрузки программы пользователя, когда CPU находится в состоянии STOP
- переполнения стека прерываний (I-стек) или стека блоков (B-стек)
- прерывания теплого рестарта (из-за прекращения подачи электропитания или изменения положения переключателя режимов)
- превышения выбранного допуска времени при прерывании перед горячим рестартом.

Ручной теплый рестарт

Ручной теплый рестарт может выполняться

- посредством переключателя режимов

  (переключатель CRST/WRST, если он доступен, должен быть установлен в положение CRST);

- посредством соответствующей команды в устройстве программирования или посредством коммуникационных функций

  (если переключатель режимов установлен в положение RUN или RUN - P).

Автоматический теплый рестарт

Автоматический теплый рестарт может выполняться вслед за включением электропитания в следующих ситуациях:

- CPU был не в состоянии STOP, когда прекратилась подача электропитания.
- Переключатель режимов установлен в положение RUN или RUN-P.
- Не запрограммирован автоматический горячий рестарт вслед за включением питания
- Работа CPU была прервана прекращением подачи электропитания во время теплого рестарта (независимо от запрограммированного типа рестарта).

Переключатель CRST/WRST не влияет на автоматический теплый рестарт.

Автоматический теплый рестарт при отсутствии резервного батарейного питания

Если вы эксплуатируете ваш CPU без резервного батарейного питания (когда необходимо функционирование без обслуживания), то память CPU автоматически сбрасывается и выполняется теплый рестарт после того, как включается питание, или тогда, когда питание восстанавливается после прекращения его подачи. Программу пользователя нужно размещать на плате памяти (флэш-СППЗУ).

Горячий рестарт

CPU S7-400 после прекращения подачи питания в режиме RUN и последующего его восстановления проходят процедуру инициализации, а затем автоматически выполняют горячий рестарт. Во время горячего рестарта выполнение программы пользователя возобновляется в точке, где оно было прервано. Участок программы пользователя, который не был выполнен перед отказом питания, известен как «остаток цикла». Остаток цикла может содержать также разделы программы, управляемые временем и прерываниями.

Горячий рестарт разрешается только тогда, когда программа пользователя не изменялась в состоянии STOP (например, посредством перезагрузки измененного блока) и нет иных оснований в пользу теплого рестарта. Возможен как ручной, так и автоматический горячий рестарт.

Ручной горячий рестарт

Ручной горячий рестарт возможен только в сочетании с соответствующими установками параметров в наборе параметров CPU и только тогда, когда состояние STOP последовал по одной из следующих причин:

- Переключатель режимов был переведен из RUN в STOP.
- Команды STOP, запрограммированные пользователем, или STOP после вызова незагруженных OB.
- Состояние STOP стал результатом выполнения команды из устройства программирования или коммуникационной функции.

Ручной горячий рестарт может выполняться

- посредством переключателя режимов;

  переключатель CRST/WRST должен быть установлен в положение WRST;
- посредством соответствующей команды в устройстве программирования или посредством коммуникационных функций (переключатель режимов установлен в положение RUN или RUN-P);
- когда ручной горячий рестарт, задан в наборе параметров CPU.

Автоматический горячий рестарт

Автоматический горячий рестарт может выполняться вслед за включением электропитания в следующих ситуациях:

- CPU не был в состоянии STOP или HOLD, когда прекратилась подача электропитания.
- Переключатель режимов установлен в положение RUN или RUN-P.
- Автоматический горячий рестарт, следующий за включением питания, задан в наборе параметров CPU.

Переключатель CRST/WRST не влияет на автоматический горячий рестарт.

Области данных, сохраняемые при выключении электропитания

CPU в S7-300 и S7-400 по-разному реагируют на включение электропитания после прекращения его подачи.

CPU в S7-300 (за исключением CPU 318) способны только к теплому рестарту. Однако вы можете при помощи STEP 7 определить биты памяти, таймеры, счетчики и области в блоках данных как сохраняемые, чтобы избежать потери данных, вызываемой прекращением подачи электропитания. Когда электропитание восстанавливается, выполняется автоматический теплый рестарт из памяти.

CPU в S7-400 реагируют на восстановление электропитания в зависимости от настройки параметров либо посредством теплого рестарта (после включения электропитания на фоне сохраняемых или не сохраняемых данных), либо посредством горячего рестарта (возможен только после включения электропитания на фоне сохраняемых данных).

Следующая таблица показывает данные, которые сохраняются в CPU в S7-300 и S7-400 во время теплого рестарта, холодного рестарта или горячего рестарта.

X	означает:	данные сохраняются
VC	означает:	логический блок сохраняется в СППЗУ, любые перезагруженные логические блоки теряются
VX	означает:	блок данных сохраняется только, если он в СППЗУ, сохраняемые данные извлекаются из NV-RAM (энергонезависимого ОЗУ) (загруженные или созданные блоки данных в ОЗУ теряются)
0	означает:	данные сбрасываются или стираются (содержимое DB)
V	означает:	данные устанавливаются на инициализирующие значения, извлекаемые из памяти СППЗУ
---	означает:	невозможно, так как NV-RAM недоступна.

			СППЗУ (плата памяти или встроенное)						
		CPU с резервной батареей				**CPU без резервной батареи**			
Данные	Блоки в загрузочной памяти	DB в рабочей памяти	Биты памяти, таймеры, счетчики (определены как сохраняемые)	Биты памяти, таймеры, счетчики (определены как несохраняемые)	Блоки в загрузочной памяти	DB в рабочей памяти (определены как сохраняемые)	DB в рабочей памяти (определены как несохраняемые)	Биты памяти, таймеры, счетчики (определены как сохраняемые)	Биты памяти, таймеры, счетчики (определены как несохраняемые)
Теплый рестарт в S7-300	X	X	X	0	VC	VX	V	X	0
Теплый рестарт в S7-400	X	X	X	0	VC	---	V	0	0
Холодный рестарт в S7-300	X	X	0	0	VC	V	V	0	0
Холодный рестарт в S7-400	X	X	0	0	VC	---	V	0	0
Горячий рестарт в S7-400	X	X	X	X		Разрешен	только рестарт	теплый	

Действия при запуске

Следующая таблица показывает, какие действия выполняются CPU во время запуска:

Действия в порядке выполнения	При теплом рестарте	При холодном рестарте	При горячем рестарте
Очистка I-стека /B-стека	X	X	0
Очистка не сохраняемых битов памяти, таймеров, счетчиков	X	0	0
Очистка всех битов памяти, таймеров, счетчиков	0	X	0
Очистка таблицы выходов образа процесса	X	X	выбираемая
Очистка выходов модулей цифровых сигналов	X	X	выбираемая
Сброс аппаратных прерываний	X	X	0
Сброс диагностических прерываний	X	X	X
Обновление списка состояний системы (SZL)	X	X	X
Оценка параметров модулей и их передача модулям или передача значений по умолчанию	X	X	X
Выполнение подходящего OB запуска	X	X	X
Выполнение остатка цикла (часть программы пользователя, не выполненная вследствие выключения электропитания)	0	0	X
Обновление таблицы входов образа процесса	X	X	X
Разблокировка цифровых выходов (отмена сигнала OD) после переключения в RUN	X	X	X

X означает: выполняется
0 означает: не выполняется

Прерывание процесса запуска

Если во время запуска происходит ошибка, то запуск прерывается и CPU переключается в состояние STOP или остается в этом режиме.

Прерванный теплый рестарт можно повторить. После прерывания рестарта возможен как теплый рестарт, так и горячий рестарт.

Запуск (теплый рестарт или горячий рестарт) не выполняется или прерывается в следующих ситуациях:

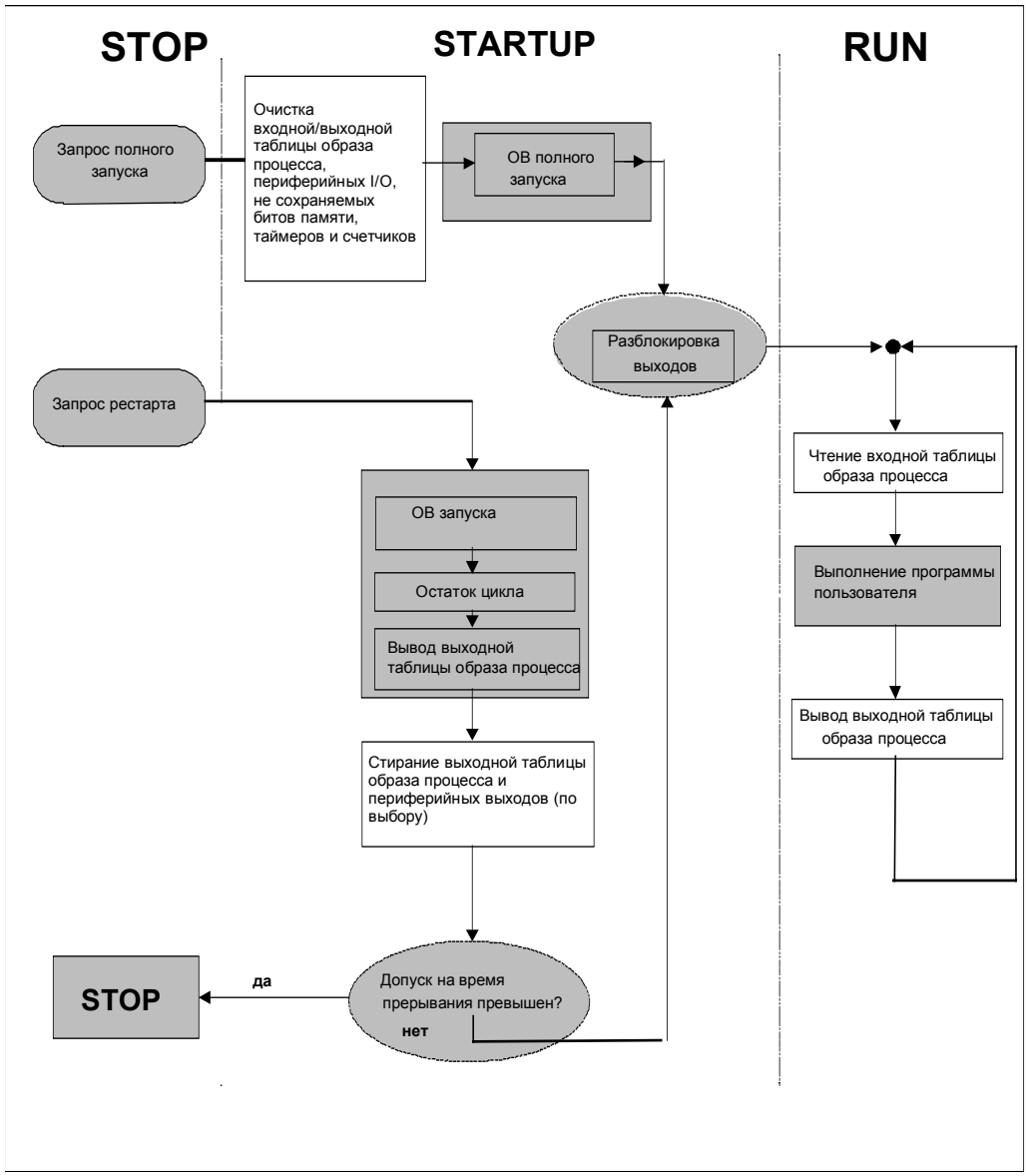
- Переключатель режимов работы CPU установлен в положение STOP.
- Затребован сброс памяти.
- Подключена плата памяти с кодом приложения, не разрешенным в STEP 7 (например, STEP 5).
- В однопроцессорный режим включено более одного CPU .
- Программа пользователя содержит OB, который не распознается CPU или заблокирован.
- После включения электропитания CPU обнаруживает, что не все модули, перечисленные в конфигурационной таблице, созданной с помощью STEP 7, действительно вставлены (различие между предварительно заданным и фактическим назначением параметров не разрешается).
- Во время оценивания параметров модулей происходят ошибки.

Горячий рестарт не выполняется или прерывается в следующих ситуациях:

- Память CPU была сброшена (после сброса памяти возможен только теплый рестарт).
- Был превышен допуск на время прерывания (это время между моментом выхода из режима RUN до момента завершения выполнения OB запуска, включая остаток цикла).
- Была изменена конфигурация модулей (например, заменен модуль).
- Назначение параметров разрешает только теплый рестарт.
- Если блоки загружались, удалялись или изменялись в то время, когда CPU был в состоянии STOP.

Последовательность действий

Следующий рисунок показывает действия CPU в режимах STARTUP и RUN:



A.1.4 Режим RUN

CPU в режиме RUN выполняет циклическую программу, а также программу, управляемую временем, и программу, управляемую прерываниями, следующим образом:

- С входов считывается образ процесса.
- Выполняется программа пользователя.
- Выводится таблица выходов образа процесса.

Активный обмен данными между CPU с использованием связи через глобальные данные (таблица глобальных данных), с использованием коммуникационных SFB для конфигурированных соединений и с использованием коммуникационных SFC для не конфигурированных соединений возможен только в режиме RUN.

Следующая таблица показывает пример, когда возможен обмен данными в разных режимах работы:

Тип связи	Режим CPU 1	Направление передачи	Режим CPU 2
Связь с помощью глобальных данных	RUN	□	RUN
	RUN	□	STOP/HOLD
	STOP	□	RUN
	STOP	X	STOP
	HOLD	X	STOP/HOLD
Односторонняя передача	RUN	□	RUN
с помощью коммуникационных SFB	RUN	□	STOP/HOLD
Двусторонняя передача с помощью коммуникационных SFB	RUN	□	RUN
Односторонняя передача	RUN	□	RUN
с помощью коммуникационных SFC	RUN	□	STOP/HOLD
Двусторонняя передача с помощью коммуникационных SFC	RUN	□	RUN

□ означает: передача данных возможна в двух направлениях
□ означает: передача данных возможна только в одном направлении
X означает: передача данных невозможна

A.1.5 Режим HOLD

Режим HOLD – это специальный режим. Он используется только в целях тестирования во время запуска или в режиме RUN. Режим HOLD означает следующее:

- Все таймеры «замораживаются»: таймеры и счетчики рабочего времени не обрабатываются, контроль времени останавливается, основные тактовые импульсы уровней, управляемых временем, задерживаются.
- Часы реального времени работают.
- Выходы заблокированы, но их можно разблокировать явным образом для целей тестирования.
- Входы и выходы можно устанавливать и сбрасывать.
- Если в CPU с резервным батарейным питанием в режиме HOLD прекращается подача электропитания, то после восстановления электропитания CPU переключается в режим останова, но не выполняет автоматический горячий или теплый рестарт. CPU без резервного батарейного питания после восстановления электропитания выполняют автоматический теплый рестарт.
- Глобальные данные могут приниматься, и возможна пассивная односторонняя передача данных с использованием коммуникационных SFB для конфигурированных соединений и коммуникационных SFC для не конфигурированных соединений (см. также таблицу в разделе «Режим RUN»).

A.2 Области памяти CPU S7

A.2.1 Распределение памяти

Память CPU S7 можно разбить на три области (см. рисунок ниже):

- Загрузочная память используется для программ пользователя без назначений символических адресов или комментариев (они остаются в памяти устройства программирования). Загрузочная память может быть вида RAM (ОЗУ) или EPROM (СППЗУ)
- Блоки, не отмеченные как необходимые для запуска, будут храниться только в загрузочной памяти.
- Рабочая память (встроенное ОЗУ) содержит части программы S7, существенные для выполнения вашей программы. Программа выполняется только в областях рабочей памяти и системной памяти.
- Системная память (ОЗУ) содержит элементы памяти, предоставляемые каждым CPU программе пользователя, такие как таблицы входов и выходов образа процесса, память с побитовым доступом (меркеры), таймеры и счетчики. Системная память содержит также стек блоков и стек прерываний.
- В дополнение к вышеупомянутым областям данных, системная память CPU предоставляет также временную память (локальный стек данных), содержащую временные данные блока, когда он вызывается. Эти данные остаются действительными только до тех пор, пока блок активен.

A.2.2 Загрузочная память и рабочая память

Когда вы загружаете программу пользователя из устройства программирования в CPU, то в загрузочную и рабочую память CPU передаются только логические блоки и блоки данных.

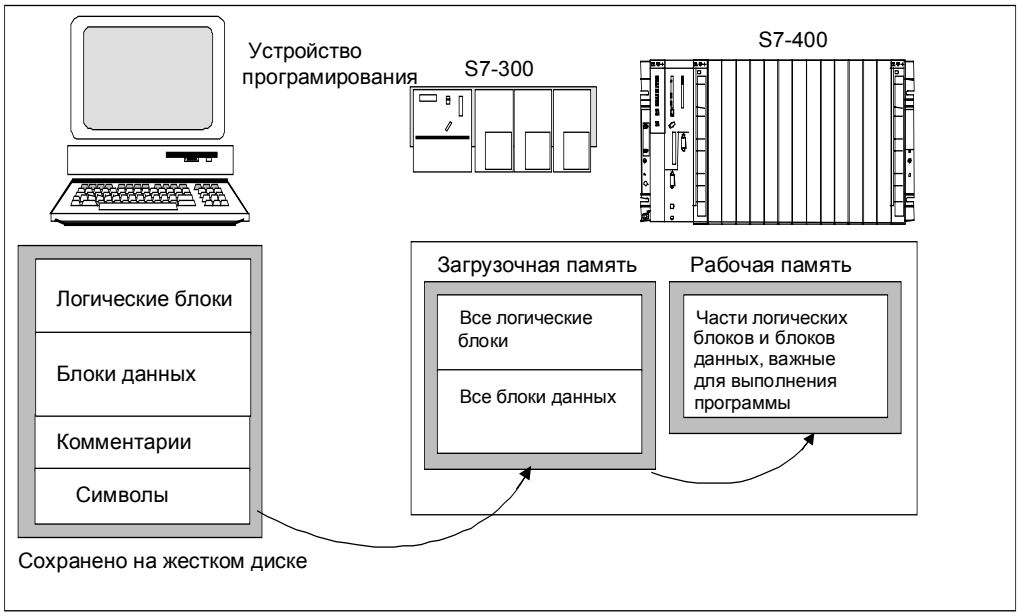
Назначение символических адресов (таблица символов) и комментарии к блокам остаются в устройстве программирования.

Подразделение программы пользователя

Чтобы обеспечить быстрое выполнение программы пользователя и избежать ненужной нагрузки на рабочую память, которую невозможно расширить, в рабочую память загружаются только части блоков, существенные для выполнения программы.

Части блоков, которые не требуются для выполнения программы (например, заголовки блоков), остаются в загрузочной памяти.

Следующий рисунок показывает программу, загруженную в память CPU.



Примечание

Блоки данных, создаваемые в программе пользователя с помощью системных функций (например, SFC22 CREAT_DB) хранятся полностью в рабочей памяти CPU.

Некоторые CPU имеют раздельно управляемые области рабочей памяти для программного кода и данных. Размер и назначение этих областей указаны во вкладке " Memory [Память]" в Информации о модулях (Module Information) для этих CPU.

Идентификация блоков данных как "несущественных для выполнения"

Блоки данных, которые были запрограммированы в исходном файле как часть программы на AWL, могут быть идентифицированы как «несущественные для выполнения» (ключевое слово UNLINKED). Это означает, что эти DB, будучи загруженными в CPU, хранятся только в загрузочной памяти. Содержимое таких блоков можно при необходимости скопировать в рабочую память, используя SFC20 BLKMOV.

Эта методика экономит пространство рабочей памяти. Тогда расширяемая загрузочная память используется как буфер (например, для формул смесей: в рабочую память загружается только формула для следующей порции).

Структура загрузочной памяти

Загрузочную память можно расширить, используя платы памяти. За информацией о максимальном размере загрузочной памяти обратитесь к руководству "S7-300 Programmable Controller, Hardware and Installation Manual [Программируемый контроллер S7-300. Аппаратные средства и монтаж]" и к справочному руководству "S7-400, M7-400 Programmable Controllers Module Specifications Reference Manual [Программируемые контроллеры S7-400, M7-400. Данные модулей]".

Загрузочная память в CPU S7-300 может иметь как встроенный блок СППЗУ (EPROM), так и встроенный блок ОЗУ (RAM). Области в блоках данных можно объявлять как сохраняемые, задавая параметры STEP 7 (см. раздел «Сохраняемые области памяти в CPU S7-300»).

Чтобы расширить загрузочную память в CPU S7-400, вам необходимо использовать плату памяти (RAM или EPROM). Встроенная загрузочная память – это оперативная память (RAM), и она используется, главным образом, для перезагрузки и исправления блоков. В новых CPU S7-400 можно подключать также дополнительную рабочую память.

Поведение загрузочной памяти в областях RAM и EPROM

В зависимости от того, выбираете ли вы для расширения загрузочной памяти плату памяти типа RAM (ОЗУ) или типа EPROM (СППЗУ), загрузочная память может реагировать по-разному во время загрузки, перезагрузки или сброса памяти.

Следующая таблица показывает различные методы загрузки:

Тип памяти	Метод загрузки	Тип загрузки
RAM	Загрузка и удаление отдельных блоков	Соединение PG-CPU
	Загрузка и удаление всей программы S7	Соединение PG-CPU
	Перезагрузка отдельных блоков	Соединение PG-CPU
Встроенная (только в S7-300) или съемная EPROM	Загрузка программ S7 целиком	Соединение PG-CPU
Съемная EPROM	Загрузка программ S7 целиком	Считывание EPROM в PG и вставка платы памяти в CPU Загрузка EPROM в CPU

Хранимые в RAM программы теряются, когда вы сбрасываете память CPU (MRES) либо снимаете CPU или плату памяти RAM.

Программы, хранимые на платах памяти EPROM, не стираются при сбросе памяти CPU и сохраняются даже при отсутствии резервного батарейного питания (транспортировка, резервные копии).

A.2.3 Системная память

A.2.3.1 Использование областей системной памяти

Системная память CPU S7 разделена на адресные области (см. таблицу ниже). Вы адресуете данные непосредственно в соответствующей адресной области с помощью команд своей программы.

Адресная область	Доступ через единицы следующего размера	Обозначение в S7 (IEC)	Описание
Таблица входов образа процесса	Вход (бит)	I	В начале цикла сканирования CPU считывает входы модулей ввода и записывает значения в этой области.
	Входной байт	IB	
	Входное слово	IW	
	Входное двойное слово	ID	
Таблица выходов образа процесса	Выход (бит)	Q	Во время цикла сканирования программа вычисляет значения выходов и помещает их в эту область. В конце цикла сканирования CPU передает расчетные значения выходов модулям вывода.
	Выходной байт	QB	
	Выходное слово	QW	
	Выходное двойное слово	QD	
Память с побитовым доступом	Бит памяти	M	Эта область обеспечивает хранение промежуточных результатов вычислений в программе.
	Байт памяти	MB	
	Слово памяти	MW	
	Двойное слово памяти	MD	
Таймеры	Таймер (T)	T	Эта область обеспечивает хранение таймеров.
Счетчики	Счетчик (C)	C	Эта область обеспечивает хранение счетчиков.
Блок данных	Блок данных, открытый посредством "OPN DB":	DB	Блоки данных содержат информацию для программы. Они могут определяться для совместного использования всеми логическими блоками (разделяемые DB) или назначаться заданному FB или SFB (экземплярные DB).
	Бит данных	DBX	
	Байт данных	DBB	
	Слово данных	DBW	
	Двойное слово данных	DBD	
	Блок данных, открытый посредством "OPN DI":	DI	
	Бит данных	DIX	
	Байт данных	DIB	

Адресная область	Доступ через единицы следующего размера	Обозначение в S7 (IEC)	Описание
	Слово данных	DIW	
	Двойное слово данных	DID	
Локальные данные	Бит локальных данных	L	Эта область содержит временно хранимые данные блока в то время, когда блок выполняется. L-стек предоставляет также память для передачи параметров блоков и для записи промежуточных результатов из сегментов контактного плана.
	Байт локальных данных	LB	
	Слово локальных данных	LW	
	Двойное слово локальных данных	LD	
Периферийная область (I/O): входы	Периферийный входной байт	PIB	Периферийные входные и выходные области допускают прямой доступ к центральным и децентрализованным модулям ввода и вывода (DP).
	Периферийное входное слово	PIW	
	Периферийное входное двойное слово	PID	
Периферийная область (I/O): выходы	Периферийный выходной байт	PQB	
	Периферийное выходное слово	PQW	
	Периферийное выходное двойное слово	PQD	

За информацией о том, какие области адресов возможны в вашем CPU, обратитесь к следующим руководствам по CPU и спискам команд CPU:

- Руководство "S7-300 Programmable Controller, Hardware and Installation [Программируемый контроллер S7-300: Аппаратные средства и монтаж]"
- Справочное руководство "S7-400, M7-400 Programmable Controllers, Module Specifications [Программируемые контроллеры S7-400, M7-400. Спецификации модулей]"
- "S7-300 Programmable Controller, Instruction List [Программируемый контроллер S7-300: Список команд]"
- "S7-400 Programmable Controller, Reference Guide [Программируемый контроллер S7-400: Справочное руководство]"

A.2.3.2 Таблицы входов и выходов образа процесса

Если в программе пользователя происходит обращение к входным (I) и выходным (Q) адресным областям, то программа не сканирует состояния сигналов в модулях цифровых сигналов, а обращается к области памяти в системной памяти CPU и к периферийным входам/выходам. Эта область памяти известна как образ процесса.

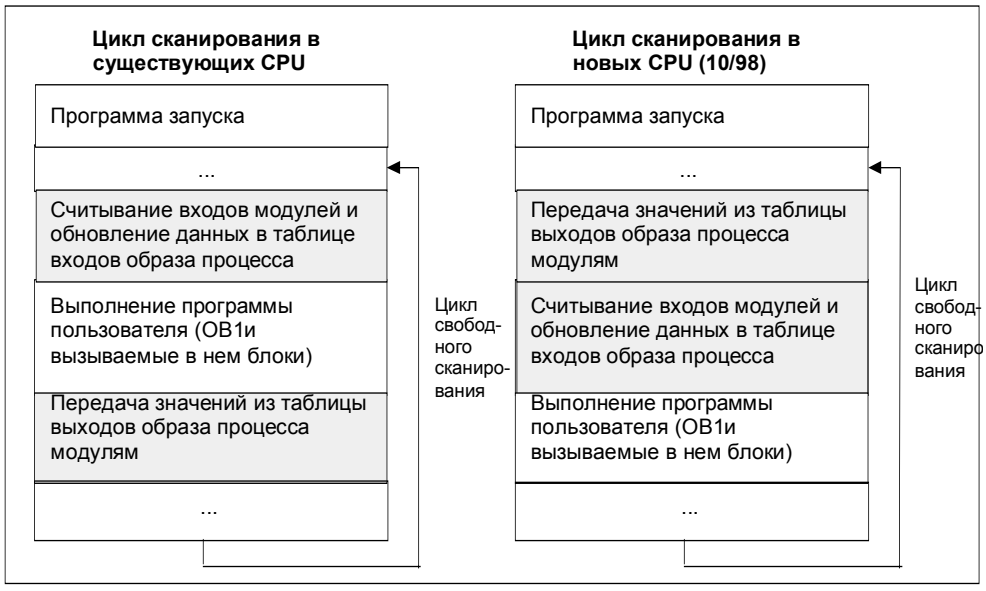
Образ процесса подразделяется на две части: таблица входов образа процесса и таблица выходов образа процесса.

Требование при обращении к образу процесса

CPU может обращаться к образу процесса только тех модулей, которые вы конфигурировали с помощью STEP 7 или которые доступны с помощью "Daedalus"-адресации.

Обновление образа процесса

Образ процесса циклически обновляется операционной системой. Следующий рисунок показывает шаги обработки в пределах цикла сканирования, сравнивая существующие CPU и новые CPU, доступные с октября 1998 г.



Преимущества использования образа процесса

По сравнению с прямым доступом к модулям ввода–вывода основное преимущество доступа к образу процесса состоит в том, что CPU имеет непротиворечивый образ сигналов процесса в течение одного цикла программы. Если состояние сигнала во входном модуле изменяется в то время, когда выполняется программа, то состояние сигнала в образе процесса сохраняется до тех пор, пока образ процесса не обновится снова в следующем цикле. Обращение к образу процесса также требует гораздо меньшего времени, чем прямой доступ к модулям сигналов, так как образ процесса находится во внутренней памяти CPU.

Обновление разделов образа процесса

Некоторые CPU позволяют структурировать и обновлять до восьми разделов таблицы образа процесса (см. руководство "S7-300 Programmable Controller, Hardware and Installation Manual [Программируемый контроллер S7-300: Аппаратные средства и монтаж]" и справочное руководство "S7-400, M7-400 Programmable Controllers Module Specifications Reference Manual [Программируемые контроллеры S7-400, M7-400: Спецификации модулей]"). Это означает, что программа пользователя может при необходимости обновлять разделы таблицы образа процесса независимо от циклического обновления таблицы образа процесса.

Вы определяете разделы образа процесса посредством STEP 7. Для обновления раздела таблицы образа процесса используются SFC.

Использование SFC

Программа пользователя может обновлять всю таблицу образа процесса или разделы таблицы образа процесса, используя следующие SFC:

- SFC26 UPDAT_PI обновляет таблицу входов образа процесса.
- SFC27 UPDAT_PO обновляет таблицу выходов образа процесса.

Примечание

В CPU S7-300 входы и выходы, не используемые в таблицах образа процесса, можно использовать как дополнительные области памяти с побитовым доступом. Программы, использующие эту возможность, не могут выполняться в CPU S7-400.

A.2.3.3 Стек локальных данных

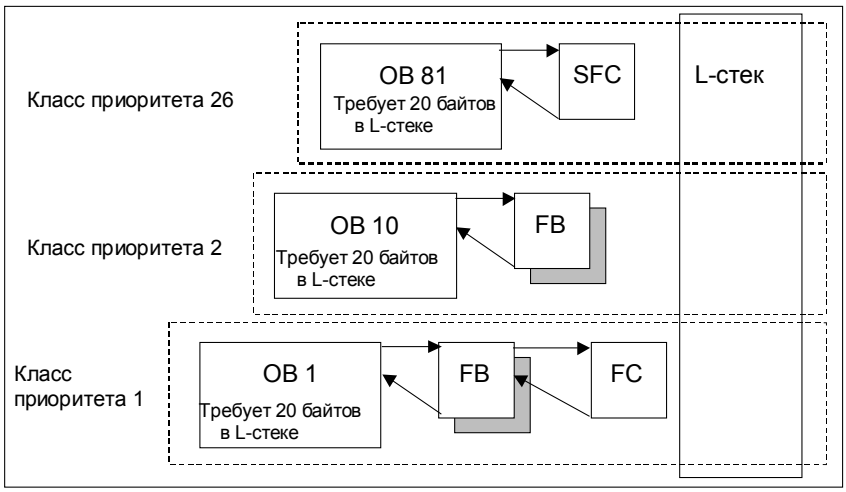
L-стек хранит:

- временные переменные локальных данных блоков
- стартовую информацию организационных блоков
- информацию о передаваемых параметрах
- промежуточные результаты логики в программах, представленных в виде контактного плана

Программируя организационные блоки, вы можете объявлять временные переменные (TEMP), которые доступны только тогда, когда блок выполняется, а после этого поверх них записываются новые данные. Прежде чем вы обратитесь к стеку локальных данных в первый раз, локальные данные должны быть инициализированы. Кроме того, каждый организационный блок требует также 20 байтов локальных данных для своей стартовой информации.

CPU имеет ограниченный объем памяти для временных переменных (локальных данных) блоков, выполняемых в текущий момент времени. Размер этой области памяти, стека локальных данных, зависит от CPU. Локальный стек данных разделен поровну между классами приоритета (значение по умолчанию). Это означает, что каждый класс приоритета имеет свою собственную область локальных данных, гарантируя тем самым, что классы более высокого приоритета и их OB тоже имеют пространство, доступное для их локальных данных.

Следующий рисунок показывает соответствие локальных данных классам приоритета на примере, где в L-стеке OB1 прерывается OB10, который затем прерывается OB81.



Предостережение
Все временные переменные (TEMP) OB и связанных с ним блоков сохраняются в L-стеке. Если вы используете слишком много уровней вложенности при выполнении ваших блоков, то L-стек может переполниться

Когда превышается допустимый размер L-стека для программы, CPU S7 переключаются в состояние STOP.

Проверьте в вашей программе L-стек (временные переменные).

Нужно учитывать в локальных данных потребности OB синхронных ошибок.

Назначение локальных данных классам приоритета

Не каждый класс приоритета требует одного и того же объема памяти в стеке локальных данных. Назначая параметры в STEP 7, вы можете устанавливать для отдельных классов приоритета в CPU S7-400 и CPU 318 области локальных данных разного размера. Любые классы приоритета, которые вам не требуются, можно отменить. Тогда в CPU S7-400 и CPU 318 увеличивается область памяти для других классов приоритета. Во время выполнения программы деактивированные OB игнорируются и экономят время цикла.

В других CPU S7-300 каждому классу приоритета назначается фиксированный объем локальных данных (256 байтов), который не может изменяться.

A.2.3.4 Стек прерываний

Если выполнение программы прерывается OB более высокого приоритета, то операционная система сохраняет текущее содержимое аккумуляторов и адресных регистров, номера и размеры открытых блоков данных в стеке прерываний.

Когда новый OB будет выполнен, операционная система загрузит информацию из I-стека и возобновит выполнение прерванного блока с точки прерывания.

Когда CPU находится в состоянии STOP, вы можете отобразить I-стек на экране устройства программирования, используя STEP 7. Это позволит вам выяснить, почему CPU переключился в состояние STOP.

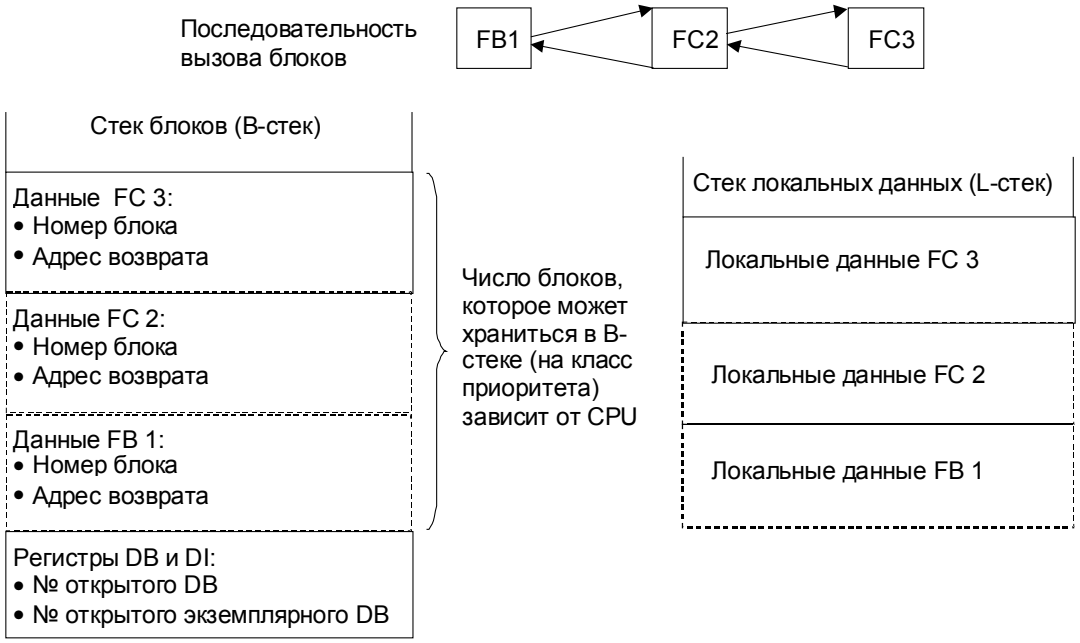
A.2.3.5 Стек блоков

Если обработка блока прерывается вызовом другого блока или классом более высокого приоритета (обслуживание прерывания/ошибки), то B-стек сохраняет следующие данные:

- Номер, тип (OB, FB, FC, SFB, SFC) и адрес возврата прерванного блока.
- Номера блоков данных (из регистров DB и DI), открытых к моменту, когда блок был прерван.

Тогда с использованием этих данных выполнение программы пользователя может быть возобновлено после прерывания.

Если CPU находится в состоянии STOP, то вы можете с помощью STEP 7 отобразить B-стек на экране устройства программирования. B-стек перечисляет все блоки, которые не были полностью выполнены к моменту, когда CPU переключился в состояние STOP. Блоки перечисляются в порядке, в котором была начата обработка (см. рисунок ниже).



Регистры блоков данных

Есть два регистра блоков данных. Они содержат номера открытых блоков данных:

- Регистр DB содержит номер открытого совместно используемого блока данных.
- Регистр DI содержит номер открытого экземплярного блока данных.

A.2.3.6 Анализ диагностического буфера

Частью списка состояний системы является диагностический буфер, содержащий подробную информацию о системных диагностических событиях и диагностических событиях, определенных пользователем, в той последовательности, в которой они произошли. Информация, вводимая в диагностический буфер при появлении системного диагностического события, идентична стартовой информации, передаваемой соответствующему организационному блоку.

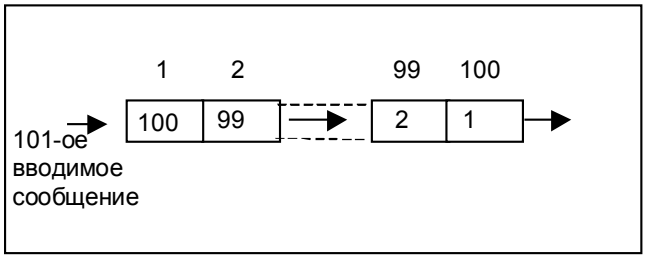
Вы не можете удалить данные, вводимые в диагностический буфер, и его содержимое сохраняется даже после сброса памяти.

Диагностический буфер предоставляет вам следующие возможности:

- Когда CPU переключается в состояние STOP, вы можете проанализировать последние события, приведшие к этому состоянию, и установить причину.
- Можно гораздо быстрее устанавливать причины ошибок, что увеличивает работоспособность системы.
- Вы можете оценивать и оптимизировать динамические характеристики системы.

Организация диагностического буфера

Диагностический буфер рассчитан на работу в качестве циклического буфера для некоторого максимального числа введенных сообщений, которое зависит индивидуально от модуля. Это означает, что после того, как количество введенных сообщений достигнет этого максимума, следующее сообщение для диагностического буфера вызовет стирание самого старого введенного сообщения. Затем все введенные сообщения сдвигаются назад на одну позицию. Это означает, что самое новое введенное сообщение всегда является первым в диагностическом буфере. Для CPU 314 в S7-300 количество возможных введенных сообщений равно 100:



Число введенных сообщений, отображаемых в диагностическом буфере, зависит от модуля и его режима работы в текущий момент времени. В некоторых CPU возможно задавать длину диагностического буфера.

Содержимое диагностического буфера

Верхнее окно содержит список всех произошедших диагностических событий вместе со следующей информацией:

- Порядковый номер записи (самая последняя запись имеет номер 1)
- Время и дата диагностического события: время и дата модуля отображаются, если в модуле есть встроенные часы. Для достоверности информации о времени в буфере важно, чтобы вы установили время и дату в модуле и регулярно проверяли их.
- Краткое описание события диагностики.

В **нижнем** окне отображается вся дополнительная информация для события, выбранного в списке в верхнем окне. Она включает в себя:

- номер события
- описание события
- переключение режима, вызванное диагностическим событием
- указание местоположения ошибки в блоке (тип блока, номер блока, относительный адрес), вызвавшей ввод сообщения в буфер
- состояние события, входящего или исходящего
- дополнительная информация, характерная для события

С помощью кнопки "Help on Event [Справка о событии]" вы можете отобразить дополнительную информацию о событии, выбранном в верхнем окне.

Сохранение содержимого в текстовом файле

Вы можете сохранить содержимое диагностического буфера в виде текста в коде ASCII, нажав кнопку "Save As [Сохранить как…]" во вкладке "Diagnostic Buffer [Диагностический буфер]" диалогового окна "Module Information [Информация о модуле]".

Отображение диагностического буфера

Вы можете отобразить содержимое диагностического буфера в устройстве программирования через вкладку "Diagnostic Buffer [Диагностический буфер]" в диалоговом окне "Module Information [Информация о модуле]" или через программу, используя системную функцию SFC51 RDSYSST.

Последняя запись перед состоянием STOP

Вы можете указать, чтобы последнее сообщение, введенное в диагностический буфер перед переключением из режима RUN в состояние STOP, автоматически передавалось зарегистрированному контрольному устройству (например, PG,OP, TD) для того, чтобы быстрее установить и устранить причину переключения в состояние STOP.

A.2.3.7 Сохраняемые области памяти в CPU S7-300

Если прекращается подача электропитания или сбрасывается память CPU (MRES), то память CPU S7-300 (динамическая загрузочная память (ОЗУ), рабочая память и системная память) сбрасывается, и все содержавшиеся прежде в этих областях данные теряются. В CPU S7-300 вы можете защитить вашу программу и ее данные следующими способами:

- Вы можете защитить все данные в загрузочной памяти, рабочей памяти и в разделах системной памяти с помощью резервного батарейного питания.
- Вы можете хранить вашу программу в СППЗУ (на плате памяти либо встроенное в CPU; обратитесь к руководству "S7-300 Programmable Controller, Hardware and Installation [Программируемый контроллер S7-300: Аппаратные средства и монтаж]")
- В зависимости от CPU, вы можете хранить определенное количество данных в области энергонезависимой памяти (NVRAM).

Использование NVRAM

Ваш CPU S7-300 предоставляет область в NVRAM (энергонезависимое ОЗУ) (см. рисунок ниже). Если вы сохранили вашу программу в загрузочной памяти СППЗУ, то вы можете сохранять определенные данные (когда прекращается подача электропитания или CPU переключается из состояния STOP в режим RUN), конфигурируя ваш CPU соответствующим образом.

Чтобы сделать это, настройте CPU таким образом, чтобы в энергонезависимом ОЗУ сохранялись следующие данные:

- данные, содержащиеся в DB (это полезно только тогда, когда вы также сохранили вашу программу в загрузочной памяти EPROM)
- значения таймеров и счетчиков
- данные, хранимые в памяти с побитовым доступом.

В каждом CPU вы можете сохранять определенное количество таймеров, счетчиков и битов памяти (меркеров). Также доступно определенное количество байтов, в которых можно сохранять данные, содержащиеся в DB.

Адрес MPI вашего CPU хранится в NVRAM. Это гарантирует, что ваш CPU способен к связи после прекращения подачи питания или сброса памяти.

Использование резервного батарейного питания для защиты данных

При использовании резервного батарейного питания загрузочная память и рабочая память сохраняются в период отсутствия электропитания. Если вы конфигурируете ваш CPU так, чтобы таймеры, счетчики и память с побитовым доступом сохранялись в NVRAM, то эта информация сохраняется также независимо от того, используете ли вы резервное батарейное питание или нет.

Конфигурирование данных NVRAM

Когда вы конфигурируете ваш CPU с помощью STEP 7, вы можете решать, какие области памяти будут сохраняемыми.

Объем памяти, который можно конфигурировать в NVRAM, зависит от используемого вами CPU. Вы не можете резервировать большее количество данных, чем то, которое указано для вашего CPU.

A.2.3.8 Сохраняемые области памяти в CPU S7-400

Эксплуатация без резервного батарейного питания

Если вы эксплуатируете вашу систему без резервного батарейного питания и прекращается подача электропитания или вы сбрасываете память CPU (MRES), то память CPU S7-400 (динамическая загрузочная память (RAM), рабочая память и системная память) сбрасывается, и все содержащиеся в этих областях памяти данные теряются.

Без резервного батарейного питания возможен только теплый рестарт, и не существует сохраняемых областей памяти. После прекращения подачи электропитания сохраняются только параметры MPI (например, адрес MPI CPU). Это означает, что после прекращения подачи электропитания или сброса памяти CPU остается способным к связи.

Эксплуатация с резервным батарейным питанием

Если вы используете резервное батарейное питание для резервирования вашей памяти, то:

- Полное содержимое всех областей ОЗУ сохраняется, когда CPU перезапускается после отказа электропитания.
- Во время теплого рестарта области адресов памяти с побитовым доступом, таймеров и счетчиков стираются. Содержимое блоков данных сохраняется.
- Содержимое рабочей памяти ОЗУ также сохраняется, кроме памяти с побитовым доступом, таймеров и счетчиков, которые были спроектированы как не сохраняемые.

Конфигурирование сохраняемых областей данных

Вы можете объявлять определенное количество битов памяти, таймеров и счетчиков в качестве сохраняемых (это количество зависит от вашего CPU). Эти данные сохраняются также во время теплого рестарта, когда вы используете резервное батарейное питание.

Когда вы назначаете параметры с помощью STEP 7, вы определяете, какие биты памяти, таймеры и счетчики должны сохраняться во время теплого рестарта. Вы можете резервировать только такое количество данных, какое допускает ваш CPU.

За более подробной информацией об определении сохраняемых областей памяти обратитесь к справочному руководству "S7-400, M7-400 Programmable

Controllers, Module Specifications [Программируемые контроллеры S7-400, M7-400: Спецификации модулей]”.

A.2.3.9 Конфигурируемые объекты в рабочей памяти

В некоторых CPU размер объектов, таких как локальный буфер или диагностический буфер, может устанавливаться в HW Config. Например, если вы уменьшаете значения, заданные по умолчанию, то становится доступным более крупный раздел рабочей памяти где-нибудь в другом месте. Параметры настройки этих CPU могут отображаться во вкладке "Memory [Память]" окна “Module Information” [Информация о модулях]" (кнопка "Details [Подробности]").

После того как конфигурация памяти была изменена и загружена в программируемый контроллер, вы должны выполнить холодный рестарт, чтобы изменения стали действующими.

A.3 Типы данных и типы параметров

A.3.1 Введение в типы данных и типы параметров

Все данные в программе пользователя должны быть идентифицированы типом данных. Доступны следующие типы данных:

- элементарные типы данных, предоставляемые STEP 7
- составные типы данных, которые вы создаете сами, комбинируя элементарные типы данных
- типы параметров, с помощью которых вы определяете параметры, передаваемые в FB или FC.

Общие сведения

Команды списка операторов, контактного плана и функционального плана работают с объектами данных определенного размера. Например, команды двоичной логики работают с битами. Команды загрузки и передачи (AWL) и команды пересылки (KOP и FUP) работают с байтами, словами и двойными словами.

Бит – это двоичная цифра "0" или "1". Байт состоит из восьми битов, слово – из 16 битов, а двойное слово – из 32 битов.

Математические команды тоже работают с байтами, словами или двойными словами. По адресам этих байтов, слов или двойных слов вы можете закодировать числа разного формата, такие как целые числа и числа с плавающей точкой.

Когда вы используете символическую адресацию, вы определяете символы и задаете тип данных для этих символов (см. таблицу ниже). Разные типы данных имеют различные варианты формата и системы записи чисел.

Эта глава описывает только некоторых из способов записи чисел и констант. Следующая таблица перечисляет форматы чисел и констант, которые не будут объясняться подробно.

Формат	Размер в битах	Запись числа
Шестнадцатеричный	8, 16 и 32	B#16#, W#16# и DW#16#
Двоичный	8, 16 и 32	2#
Дата IEC	16	D#
Время IEC	32	T#
Время суток	32	TOD#
Символ	8	'A'

A.3.2 Элементарные типы данных

A.3.2.1 Элементарные типы данных

Каждый элементарный тип данных имеет определенную длину. Следующая таблица перечисляет элементарные типы данных.

Тип и описание	Размер в битах	Варианты формата	Диапазон и запись чисел (минимальное – максимальное значение)	Пример
BOOL (бит)	1	Булевский текст	TRUE/FALSE	TRUE
BYTE (байт)	8	Шестнадцатеричное число	от B16#0 до B16#FF	L B#16#10 L byte#16#10
WORD (слово)	16	Двоичное число Шестнадцатеричное число BCD Десятичное число без знака	от 2#0 до 2#1111_1111_1111_1111 от W#16#0 до W#16#FFFF от C#0 до C#999 от B#(0.0) до B#(255.255)	L 2#0001_0000_0000_0000 L W#16#1000 L word#1000 L C#998 L B#(10,20) L byte#(10,20)
DWORD (двойное слово)	32	Двоичное число Шестнадцатеричное число Десятичное число без знака	от 2#0 до 2#1111_1111_1111_1111 1111_1111_1111_1111 от DW#16#0000_0000 до DW#16#FFFF_FFFF от B#(0,0,0,0) до B#(255,255,255,255)	2#1000_0001_0001_1000_ 1011_1011_0111_1111 L DW#16#00A2_1234 L dword#16#00A2_1234 L B#(1, 14, 100, 120) L byte#(1,14,100,120)
INT (целое)	16	Десятичное число со знаком	от -32768 до 32767	L 1
DINT (целое, 32 бита)	32	Десятичное число со знаком	от L#-2147483648 до L#2147483647	L L#1
REAL (число с плавающей точкой)	32	Число с плавающей точкой в формате IEEE	Верхний предел: ±3.402823e+38 Нижний предел: ±1.175 495e-38	L 1.234567e+13
S5TIME (время SIMATIC)	16	Время S7 с шагом 10 мс (по умолч.)	от S5T#0H_0M_0S_10MS до S5T#2H_46M_30S_0MS и S5T#0H_0M_0S_0MS	L S5T#0H_1M_0S_0MS L S5TIME#0H_1H_1M_0S_0MS
TIME (время IEC)	32	Время IEC с шагом 1 мс, целое со знаком	от -#24D_20H_31M_23S_648MS до T#24D_20H_31M_23S_647MS	L T#0D_1H_1M_0S_0MS L TIME#0D_1H_1M_0S_0MS

Тип и описание	Раз-мер в битах	Варианты формата	Диапазон и запись чисел (минимальное – максимальное значение)	Пример
DATE (дата IEC)	16	Дата IEC с шагом 1 день	от D#1990-1-1 до D#2168-12-31	L D#1996-3-15 L DATE#1996-3-15
TIME_OF_DAY (время)	32	Время с шагом 1 мс	от TOD#0:0:0.0 до TOD#23:59:59.999	L TOD#1:10:3.3 L TIME_OF_DAY#1:10:3.3
CHAR (символ)	8	Символы кода ASCII	'A','B' и т.д.	L 'E'

A.3.2.2 Формат типа данных INT (16-битные целые числа)

Целое число имеет знак, указывающий, является ли оно положительным или отрицательным целым числом. Целое число (16 битов) занимает в памяти пространство размером в одно слово. Следующая таблица показывает диапазон целых чисел (16 битов).

Формат	Диапазон
Целое число (16 битов)	От –32 768 до +32 767

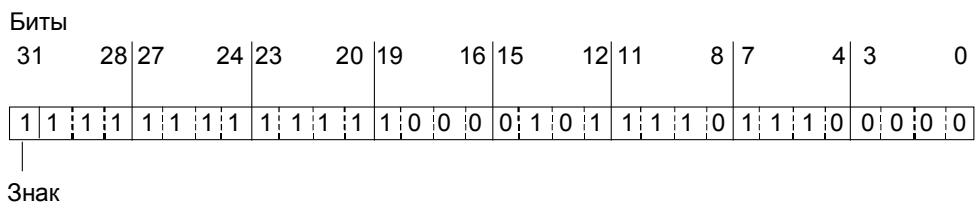
Следующий рисунок показывает целое число +44 как двоичное число:

A.3.2.3 Формат типа данных DINT (32- битные целые числа)

Целое число имеет знак, указывающий, является ли оно положительным или отрицательным целым числом. Двойное целое число занимает в памяти пространство размером в два слова. Следующая таблица показывает диапазон двойных целых чисел.

Формат	Диапазон
Целое число (32 бита)	От –2 147 483 648 до +2 147 483 647

Следующий рисунок показывает целое число -500 000 как двоичное число. В двоичной системе исчисления отрицательное целое число представляется дополнением положительного целого числа до 2. Вы получаете дополнение целого числа до 2, изменяя значения всех битов на противоположные и прибавляя затем +1 к результату.



A.3.2.4 Формат типа данных REAL (числа с плавающей точкой)

Числа в формате с плавающей точкой представляются в общем виде как "число = m * b, возведенное в степень E". Основание "b" и показатель "E" являются целыми числами; мантисса "m" является рациональным числом.

Этот тип представления чисел имеет то преимущество, что он способен представлять как очень большие, так и очень малые значения в ограниченном пространстве памяти. При ограниченном количестве битов для мантиссы и порядка можно охватить широкий диапазон чисел.

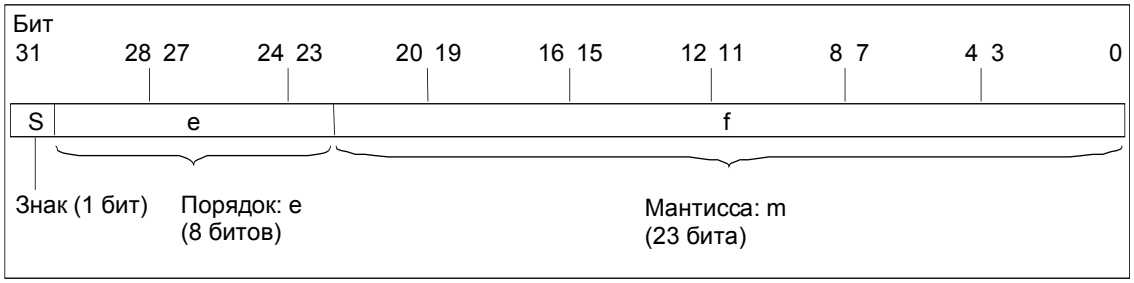
Недостатком является ограниченная точность вычислений. Например, при формировании суммы двух чисел показатели нужно согласовывать, сдвигая мантиссу (то есть плавающую десятичную точку), так как можно складывать только числа с одинаковыми показателями степени.

Формат числа с плавающей точкой в STEP 7

Числа с плавающей точкой в STEP 7 соответствуют основному формату для одинарной ширины, описанному в стандарте ANSI/IEEE 754–1985: *IEEE Standard for Binary Floating-Point Arithmetic* [Стандарт IEEE для двоичной арифметики с плавающей точкой]". Они состоят из следующих компонентов:

- Знак S
- Порядок e = E + смещение, увеличенный на константу (смещение = +127)
- Дробная часть мантиссы m.
  Целочисленная часть мантиссы не хранится вместе с остальной частью, потому что она всегда равна 1 в пределах допустимого диапазона чисел.

Эти три компонента вместе занимают одно двойное слово (32 бита):



Следующая таблица показывает значения отдельных битов в формате с плавающей точкой.

Компонент числа с плавающей точкой	Номер бита	Значение
Знак S	31	
Порядок **e**	30	2 в степени 7
...	...	...
Порядок **e**	24	2 в степени 1
Порядок **e**	23	2 в степени 0
Мантисса **m**	22	2 в степени –1
...	...	...
Мантисса **m**	1	2 в степени –22
Мантисса **m**	0	2 в степени –23

С помощью этих трех компонентов **S**, **e** и **m** значение числа, представленного в этой форме, определяется формулой:

Число = 1.**m** ∗ 2 в степени (e-смещение),

где

- e: $1 \leq e \leq 254$
- Смещение: смещение = 127. Это означает, что дополнительный знак для порядка не требуется.
- S: S = 0 для положительного числа, и S = 1 для отрицательного числа.

Диапазон значений чисел с плавающей точкой

Использование указанного выше формата с плавающей точкой приводит к следующему:

- Минимальное число с плавающей точкой = 1.0 ∗ 2 в степени (1-127) = 1.0 ∗ 2 в степени (-126) = 1.175 495E–38
- Максимальное число с плавающей точкой = 2-2 в степени (-23) ∗ 2 в степени (254-127) = 2-2 в степени (-23) ∗ 2 в степени (+127) = 3.402 823E+38

Число «нуль» представляется посредством e = m = 0; «бесконечность» представляется посредством e = 255 и m = 0.

Формат	Диапазон [1)]
Числа с плавающей точкой в соответствии со стандартом ANSI/IEEE	от –3.402 823E+38 до –1.175 495E–38 и 0 и от +1.175 495E–38 до +3.402 823E+38

Следующая таблица показывает состояния сигналов битов в слове состояния для результатов команд над числами с плавающей точкой, которые не лежат в пределах допустимого диапазона:

Недопустимый диапазон для результата	CC1	CC0	OV	OS
-1.175494E-38 < результат < -1.401298E-45 (отрицательное число) потеря значимости	0	0	1	1
+1.401298E-45 < результат < +1.175494E-38 (положительное число) потеря значимости	0	0	1	1
результат < -3.402823E+38 (отрицательное число) переполнение	0	1	1	1
результат > 3.402823E+38 (положительное число) переполнение	1	0	1	1
Недопустимое число с плавающей точкой или недопустимая команда (входное значение вне диапазона допустимых значений).	1	1	1	1

Примечание к использованию математических операций:

Результат "Недопустимое число с плавающей точкой" получается, например, когда вы пытаетесь извлечь квадратный корень из -2. Поэтому вам всегда следует сначала оценивать биты состояния в математических операциях прежде, чем продолжать вычисления, основанные на результате.

Примечание к изменению переменных:

В частности, если значения для операций с плавающей точкой хранятся в двойных словах памяти, то вы можете изменять эти значения с помощью любых битовых комбинаций. Однако не каждая битовая комбинация является допустимым числом.

Точность вычисления чисел с плавающей точкой

Предостережение
Вычисления, влекущие за собой длинный ряд значений, включая очень большие и очень малые числа, могут давать неточные результаты.

Числа с плавающей точкой в STEP 7 имеют точность до 6 десятичных разрядов. Поэтому при вводе констант с плавающей точкой вы можете задавать только до 6 десятичных разрядов.

Примечание

Точность вычисления в 6 десятичных разрядов означает, в частности, что число1 + число2 = число1, если число1 больше, чем число 2 * 10 в степени y, где y> 6:

100 000 000 + 1 = 100 000 000.

Примеры чисел в формате с плавающей точкой

Следующий рисунок показывает формат с плавающей точкой для следующих десятичных значений:

- 10.0
- π (3.141593)
- квадратный корень из 2 ($\sqrt{2}$ = 1.414214)

В первом примере число 10.0 получается из формата с плавающей точкой (шестнадцатеричное представление: 4120 0000) следующим образом:

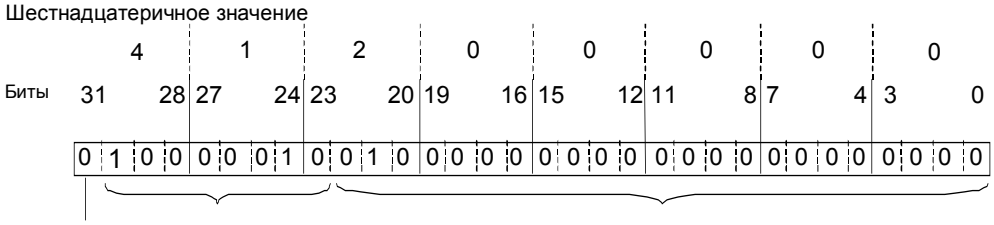
e = 2 в степени 1 + 2 в степени 7 = 2 + 128 = 130

m = 2 в степени (-2) = 0.25

Это приводит к следующему: 1.m $*$ 2 в степени (e – смещение) = 1.25 $*$ 2 в степени (130 – 127) = 1.25 $*$ 2 в степени 3 = 10.0.

Десятичное значение 10.0

Шестнадцатеричное значение



$e = 2^7 + 2^1 = 130$

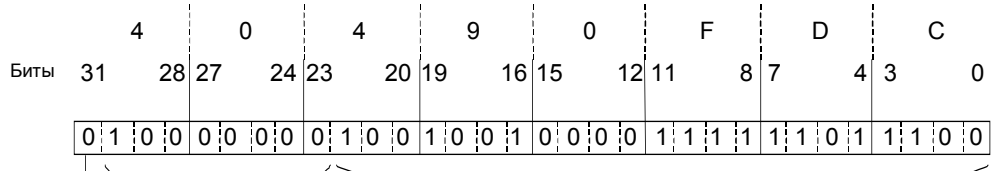
$f = 2^{-2} = 0.25$

$1.f * 2^{e\text{-}bias} = 1.25 * 2^3 = 10.0$

$[1.25 * 2^{(130-127)} = 1.25 * 2^3 = 10.0]$

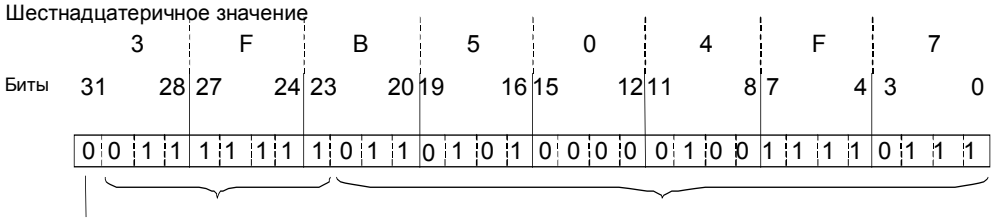
Десятичное значение 3.141593

Шестнадцатеричное значение



Десятичное значение: 1.414214

Шестнадцатеричное значение



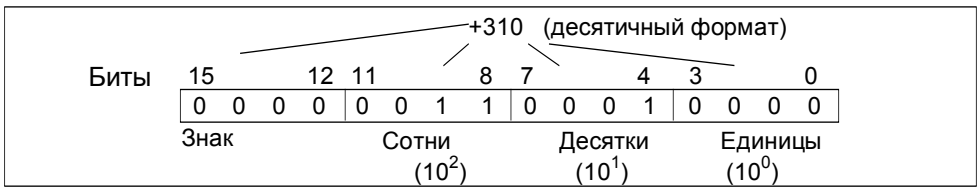
A.3.2.5 Формат типов данных WORD и DWORD для двоично-десятичных чисел

Двоично-десятичный формат (BCD) представляет десятичное число посредством групп двоичных цифр (битов). Одна группа из 4 битов представляет одну цифру десятичного числа со знаком или знак десятичного числа. Группы по 4 бита объединяются, чтобы сформировать слово (16 битов) или двойное слово (32 бита). Четыре старших значащих бита показывают знак числа (1111 обозначает «минус», а 0000 обозначает «плюс»). Команды с адресами в двоично-десятичном коде оценивают только самый старший бит (бит 15 в слове, бит 31 в двойном слове). Следующая таблица показывает формат и диапазон BCD-чисел двух типов.

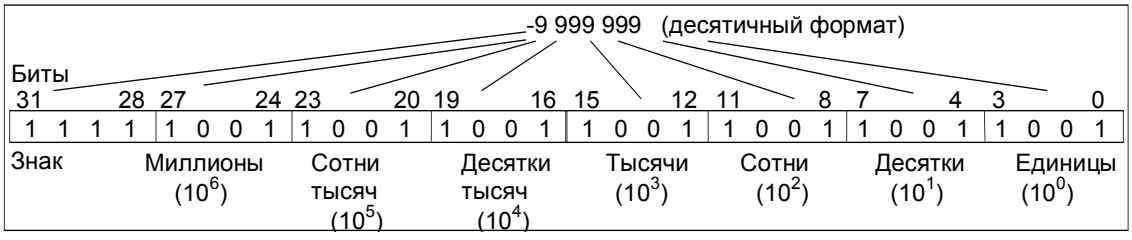
Формат	Диапазон
Слово (16 битов, 3-разрядное BCD-число со знаком)	от –999 до +999
Двойное слово (32 бита, 7-разрядное BCD-число со знаком)	от –9 999 999 до +9 999 999

Следующие рисунки представляют пример двоично-десятичного числа в следующих форматах:

- Формат слова



- Формат двойного слова



A.3.2.6 Формат типа данных S5TIME (продолжительность времени)

Когда вы вводите продолжительность времени, используя тип данных S5TIME, введенные вами значения запоминаются в двоично-десятичном формате. Следующий рисунок показывает содержимое ячейки таймера со значением времени 127 и базой времени 1 с.

При работе с типом S5TIME вы вводите значение времени в диапазоне от 0 до 999 и указываете базу времени (см. следующую таблицу). База времени обозначает интервал времени, через который таймер уменьшает значение времени на одну единицу до тех пор, пока оно не достигнет 0.

База времени для S5TIME:

База времени	Двоичный код базы времени
10 мс	00
100 мс	01
1 с	10
10 с	11

Вы можете предварительно загружать значение времени, используя любой из следующих синтаксических форматов:

- L[1)] W#16#wxyz,

  где w = база времени (т.е. интервал времени или разрешение)

  xyz = значение времени в двоично-десятичном формате

- L[1)] S5T#aH_bbM_ccS_dddMS,

  где a = часы, bb = минуты, cc = секунды и dd = миллисекунды

  База времени выбирается автоматически, и значение округляется до ближайшего меньшего числа с такой базой времени.

Максимальное значение времени, которое вы можете ввести, равно 9 990 секунд или 2H_46M_30S.

[1)] = L нужно задавать только при программировании в форме AWL

A.3.3 Составные типы данных

A.3.3.1 Составные типы данных

Составные типы данных определяют группы данных, занимающих более 32 битов, или группы данных, состоящие из других типов данных. STEP 7 допускает следующие составные типы данных:

- DATE_AND_TIME
- STRING
- ARRAY
- STRUCT
- UDT (типы данных, определяемые пользователем)
- FB и SFB

Следующая таблица описывает составные типы данных. Вы определяете структуры и массивы либо в разделе описания переменных логического блока, либо в блоке данных.

Тип данных	Описание
DATE_AND_TIME DT	Определяет область с 64 битами (8 байтов). Этот тип данных сохраняет данные в двоично-десятичном формате.
STRING	Определяет группу из максимум 254 символов (тип данных CHAR). Стандартная область, зарезервированная для символьной строки, имеет длину 256 байтов. Это пространство, требующееся для сохранения 254 символов и заголовка длиной 2 байта. Вы можете уменьшить память, требующуюся для строки, определяя число символов, которое будет сохраняться в символьной строке (например: string[9] 'Siemens').
ARRAY	Определяет многомерную группу данных одного типа (элементарного или составного). Например, ”ARRAY [1..2,1..3] OF INT" определяет массив размерности 2 x 3, состоящий из целых чисел. Вы обращаетесь к данным, хранимым в массиве, используя индекс (”[2,2]"). В одном массиве вы можете определить максимум 6 измерений. Индекс может быть любым целым числом (от -32768 до 32767).
STRUCT	Определяет группирование данных с любой комбинацией типов данных. Например, вы можете определить массив структур или структуру из структур и массивов.
UDT	Упрощает структурирование больших объемов данных и типов вводимых данных при создании блоков данных или описании переменных в разделе описания переменных. В STEP 7 вы можете комбинировать составные и элементарные типы данных, чтобы создать ваш собственный, "определяемый пользователем", тип данных. UDT имеют свое собственное имя и поэтому могут использоваться более одного раза.
FB, SFB	Вы определяете структуру из назначенных экземплярных блоков данных и разрешаете для нескольких вызовов FB передачу экземплярных данных в один экземплярный DB.

Структурированные типы данных хранятся в памяти с соблюдением границ слов (выравнивание по WORD).

A.3.3.2 Формат типа данных DATE_AND_TIME

Когда вы вводите дату и время, используя тип данных DATE_AND_TIME (DT), вводимые вами данные сохраняются в двоично-десятичном формате в 8 байтах. Тип данных DATE_AND_TIME имеет следующий диапазон значений:

от DT#1990-1-1-0:0:0.0 до DT#2089-12-31-23:59:59.999

Следующие примеры показывают синтаксис даты и времени для четверга 25 декабря 1993 года, утром в 8 часов 01 минуту и 1.23 секунды. Возможны два следующих формата:

- DATE_AND_TIME#1993–12–25–8:01:1.23
- DT#1993-12-25-8:01:1.23

Для работы с типом данных DATE_AND_TIME имеются в распоряжении следующие специальные функции стандарта IEC (International Electrotechnical Commission [Международная электротехническая комиссия]):

- Преобразование даты и времени суток в формат DATE_AND_TIME
  FC3: D_TOD_DT
- Извлечение даты из формата DATE_AND_TIME
  FC6: DT_DATE
- Извлечение дня недели из формата DATE_AND_TIME
  FC7: DT_DAY
- Извлечение времени суток из формата DATE_AND_TIME
  FC8: DT_TOD

Следующая таблица показывает содержимое байтов, которые содержат информацию о дате и времени для примера «четверг 25 декабря 1993 года, утром в 8 часов 01 минуту и 1.23 секунды».

Байт	Содержимое	Пример
0	Год	B#16#93
1	Месяц	B#16#12
2	День	B#16#25
3	Часы	B#16#08
4	Минуты	B#16#01
5	Секунды	B#16#01
6	Два старших значащих разряда мсек	B#16#23
7 (4MSB)	Два младших значащих разряда мсек	B#16#0
7 (4LSB)	День недели 1 = воскресенье 2 = понедельник ... 7 = суббота	B#16#5

Допустимый диапазон значений для типа данных DATE_AND_TIME:

- min.: DT#1990-1-1-0:0:0.0
- max.: DT#2089-12-31-23:59:59.999

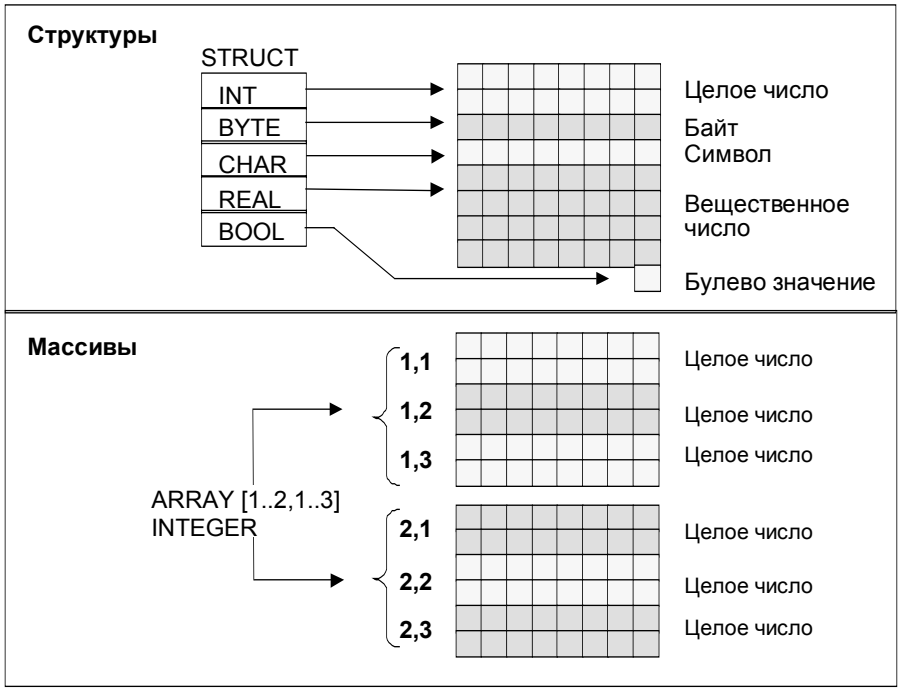
	Возможный диапазон значений	Двоично-десятичный код (BCD)
Год	1990 – 1999 2000 – 2089	90h – 99h 90h – 99h
Месяц	1 – 12	01h – 12h
День	1 – 31	01h – 31h
Часы	00 – 23	00h – 23h
Минуты	00 – 59	00h – 59h
Секунды	00 – 59	00h – 59h
Миллисекунды	0 – 999	000h – 999h
День недели	воскресенье – суббота	1h – 7h

A.3.3.3 Использование составных типов данных

Вы можете формировать новые типы данных, объединяя элементарные и составные типы данных, чтобы создать следующие составные типы данных:

- Массив (тип данных ARRAY): массив объединяет группу данных одного типа, образуя одно целое.
- Структура (тип данных STRUCT): структура объединяет данные разного типа, образуя одно целое.
- Символьная строка (тип данных STRING): символьная строка определяет одномерный массив длиной максимум 254 символа (тип данных CHAR). Символьная строка может передаваться только как одно целое. Длина символьной строки должна соответствовать формальному и фактическому параметру блока.
- Дата и время (тип данных DATE_AND_TIME): тип данных «дата и время» хранит год, месяц, день, часы, минуты, секунды, миллисекунды и день недели.

Следующий рисунок показывает, как массивы и структуры могут структурировать типы данных в одной области памяти и хранить информацию. Вы определяете массив или структуру либо в DB, либо в разделе описания переменных FB, OB или FC.



A.3.3.4 Использование массивов для доступа к данным

Массивы

Массив объединяет группу данных одного типа (элементарного или составного), образуя одно целое. Вы можете создавать массив, состоящий из массивов. Определяя массив, вы должны сделать следующее:

- Присвоить массиву имя.
- Описать массив с помощью ключевого слова ARRAY.
- Определить размер массива, используя индекс. Вы определяете номер первого и последнего элемента по отдельным измерениям массива (максимум 6 измерений). Индекс вводят в квадратных скобках, разделяя измерения посредством запятой, а номера первого и последнего элемента измерения – двумя точками. Например, следующий индекс определяет, трехмерный массив:

  [1..5,–2..3,30..32]

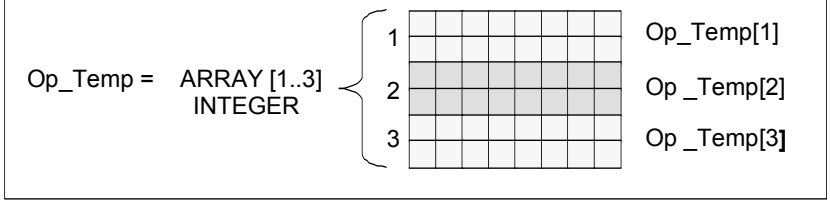
- Вы указываете тип данных, которые должны содержаться в массиве.

Пример 1

Следующий рисунок показывает массив с тремя целыми числами. Вы обращаетесь к данным, хранимым в массиве, используя индекс. Индекс – это номер в квадратных скобках. Например, вторым целым числом является Op_temp[2]".

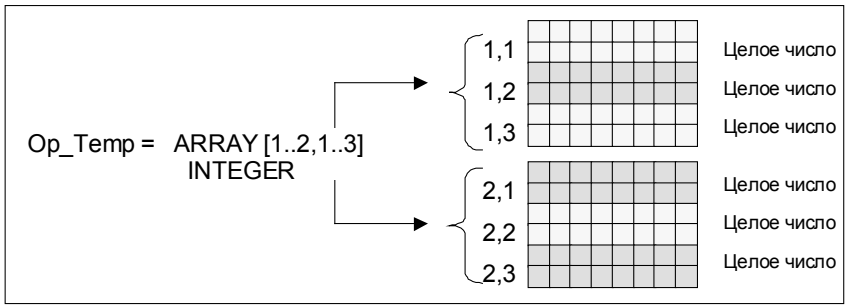
Индекс может быть любым целым числом (от -32768 до 32767), включая отрицательные значения. Массив на следующем рисунке можно было бы определить также как ARRAY [-1 .. 1]. Тогда первым целым числом было бы Op_temp[-1], вторым целым числом – Op_temp[0] и третьим целым числом – Op_temp[1].

Address	Name	Type	Init. value	Comment
0.0		STRUCT		
+0.0	Op_Temp	ARRAY[1..3]		
*2.0		INT		
=3.0		END_STRUCT		



Пример 2

Массив может также описывать многомерную группу типов данных. Следующий рисунок показывает двумерный массив целых чисел.



Вы обращаетесь к данным в многомерном массиве, используя индекс. В этом примере первым целым числом является Op_temp[1,1]", третьим – Op_temp[1,3]", четвертым – Op_temp[2,1]" и шестым – Op_temp [2,3]".

Вы можете определять в массиве до 6 измерений (6 индексов). Например, вы могли бы определить переменную Op_temp как шестимерный массив следующим образом:

ARRAY [1..3,1..2,1..3,1..4,1..3,1..4]

Индексом первого элемента в этом массиве является [1,1,1,1,1,1]. Индексом последнего элемента является [3,2,3,4,3,4].

Создание массивов

Вы определяете массивы, объявляя данные в DB или в разделе описания переменных. Когда вы объявляете массив, вы указываете ключевое слово (ARRAY), а затем размер в квадратных скобках следующим образом:

[значение нижней границы.. значение верхней границы]

В многомерном массиве вы указываете также дополнительные верхние и нижние границы и разделяете отдельные измерения посредством запятой. Следующий рисунок показывает описание для создания массива размерности 2 x 3.

Address	Name	Type	Init. value	Comment
0.0		STRUCT		
+0.0	Heat_2x3	ARRAY[1..2,1..3]		
*2.0		INT		
=6.0		END_STRUCT		

Ввод начальных значений для массива

Создавая массивы, вы можете каждому элементу массива присваивать начальное значение. STEP 7 предоставляет два метода ввода начальных значений:

- Ввод индивидуальных значений: для каждого элемента массива вы указываете значение, допустимое для типа данных этого массива. Значения указываются в порядке следования элементов: [1,1]. Помните, что отдельные элементы должны отделяться друг от друга запятой.
- Задание коэффициента повторения: при наличии последовательных элементов, имеющих одинаковое начальное значение, вы можете указать число таких элементов (коэффициент повторения) и начальное значение для этих элементов. Формат ввода коэффициента повторения имеет вид: x (y), где x – коэффициент повторения, а y – повторяемое значение.

Если вы используете массив, описанный на рисунке, показанном выше, то вы можете задать начальное значение для всех шести элементов следующим образом: 17, 23, -45, 556, 3342, 0. Вы могли бы также установить начальное значение всех шести элементов равным 10, указав 6(10). Вы могли бы задать определенные значения для первых двух элементов, а затем установить остальные 4 элемента в 0, указав следующее: 17, 23, 4(0).

Доступ к данным в массиве

Вы обращаетесь к данным в массиве через индекс определенного элемента в массиве. Индекс используется в сочетании с символическим именем.

Пример: Если массив, описанный на рисунке выше, начинается в первом байте DB20 (motor), вы обращаетесь ко второму элементу этого массива по следующему адресу:

Motor.Heat_2x3[1,2].

Использование массивов в качестве параметров

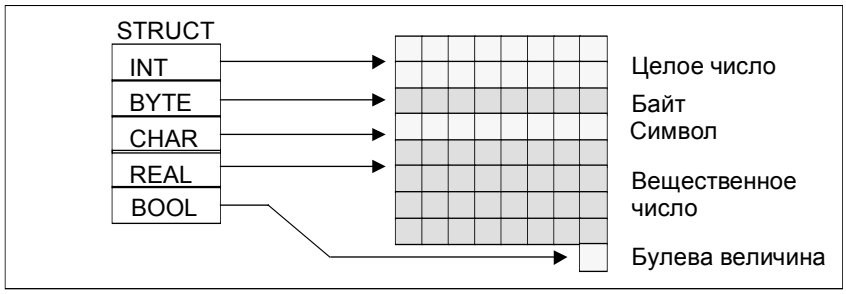
Вы можете передавать массивы как параметры. Если параметр описан в разделе описания переменных как ARRAY, то вы должны передать весь массив (а не отдельные элементы). Однако параметру может присваиваться элемент массива, когда вы вызываете блок, если элемент массива соответствует типу данных параметра.

Если вы используете массивы как параметры, то не требуется, чтобы эти массивы имели такое же имя (для них даже не нужно имени). Однако оба массива (и формальный параметр, и фактический параметр) должны иметь одинаковую структуру. Например, массив размерности 2 x 3, состоящий из целых чисел, может передаваться как параметр только тогда, когда формальный параметр блока определен как массив размерности 2 x 3, состоящий из целых чисел, и фактический параметр, предоставляемый операцией вызова, тоже является массивом размерности 2 x 3, состоящим из целых чисел.

A.3.3.5 Использование структур для доступа к данным

Структуры

Структура объединяет различные типы данных (элементарные и составные типы данных, включая массивы и структуры), образуя одно целое. Вы можете группировать данные так, чтобы приспособить их к управлению вашим процессом. Поэтому вы можете также передавать параметры как единицу данных, а не как отдельные элементы. Следующий рисунок показывает структуру, состоящую из целого числа, байта, символа, числа с плавающей точкой и булевой величины.



Структура может иметь до 8 вложенных уровней (например, структура, состоящая из структур, содержащих массивы).

Создание структуры

Вы определяете структуры, описывая данные внутри DB или в разделе описания переменных логического блока.

Следующий рисунок иллюстрирует описание структуры (*Stack_1*), которая состоит из следующих элементов: целое число (для хранения количества), байт (для хранения исходных данных), символ (для хранения управляющего кода), число с плавающей точкой (для хранения температуры), и булев бит памяти (для завершения сигнала).

Address	Name	Type	Init. value	Comment
0.0	Stack_1	STRUCT		
+0.0	Amount	INT	100	
+2.0	Original_data	BYTE		
+4.0	Control_code	CHAR		
+6.0	Temperature	REAL	120	
+8.1	End	BOOL	FALSE	
=10.0		END_STRUCT		

Присваивание структуре начальных значений

Если вы хотите присвоить начальное значение каждому элементу структуры, то указывайте значение, допустимое для типа данных и имени элемента.

Например, вы можете присвоить следующие начальные значения (структуре, объявленной на рисунке выше):

Amount [количество]	=	100
Original_data [исходные_данные]	=	B#(0)
Control_code [управляющий_код]	=	'C'
Temperature [температура]	=	120
End [конец]	=	False

Хранение и доступ к данным в структурах

У вас есть доступ к отдельным элементам структуры. Вы можете использовать символические адреса (например, *Stack_1.Temperature*). Однако вы можете указывать абсолютный адрес, по которому расположен элемент (пример: если Stack_1 расположен в DB20, начиная с байта 0, то абсолютный адрес для *amount* – это *DB20.DBW0* и адрес для *temperature* – это *DB20.DBD6*).

Использование структур в качестве параметров

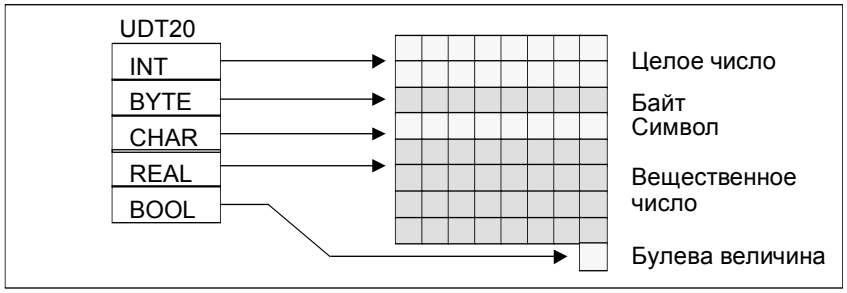
Вы можете передавать структуры в качестве параметров. Если параметр описывается как STRUCT в разделе описания переменных, то вы должны передавать структуру с теми же самыми компонентами. Однако параметру может присваиваться также элемент структуры, когда вы вызываете блок, если элемент структуры соответствует типу данных параметра.

Если вы используете структуры в качестве параметров, то обе структуры (для формальных параметров и для фактических параметров) должны иметь одинаковые компоненты, другими словами, одинаковые типы данных должны располагаться в одинаковой последовательности.

A.3.3.6 Использование определяемых пользователем типов данных для доступа к данным

Типы данных, определяемые пользователем

Типы данных, определяемые пользователем (UDT), могут объединять элементарные и составные типы данных. Вы можете присваивать UDT имена и использовать их более одного раза. Следующий рисунок иллюстрирует структуру определяемого пользователем типа данных, состоящего из целого числа, байта, символа, числа с плавающей точкой и булевой величины.



Вместо ввода всех типов данных по одному или в виде структуры вам нужно лишь определить "UDT20" как тип данных, и STEP 7 автоматически выделит соответствующее пространство памяти.

Создание определяемого пользователем типа данных

Вы определяете UDT с помощью STEP 7. Следующий рисунок показывает UDT, состоящий из следующих элементов: целое число (для хранения количества), байт (для хранения исходных данных), символ (для хранения управляющего кода), число с плавающей точкой (для хранения температуры), и булев бит памяти (для завершения сигнала). Вы можете присвоить UDT символическое имя в таблице символов (например, *process data*).

Address	Name	Type	Init. value	Comment
0.0	Stack_1	STRUCT		
+0.0	Amount	INT	100	
+2.0	Original_data	BYTE		
+4.0	Control_code	CHAR		
+6.0	Temperature	REAL	120	
+8.1	End	BOOL	FALSE	
=10.0		END_STRUCT		

Как только вы создали UDT, вы можете использовать этот UDT подобно типу данных, например, когда вы описываете тип данных *UDT200* для переменной в DB (или в разделе описания переменных FB).

Следующий рисунок показывает DB с переменными *process_data_1* с типом данных *UDT200*. Вы определяете только *UDT200* и *process_data_1*. Массивы, показанные курсивом, создаются, когда вы компилируете DB.

Address	Name	Type	Init. value	Comment
0.0		STRUCT		
+6.0	Process_data **_1**	UDT200		
=6.0		END_STRUCT		

Присваивание начальных значений определяемому пользователем типу данных

Если вы хотите присвоить начальное значение каждому элементу типа данных, определяемого пользователем, то указывайте значение, допустимое для типа данных и имени элемента. Например, вы можете присвоить следующие начальные значения (определяемому пользователем типу данных, описанному на рисунке выше):

```
Amount [количество]                =    100
Original_data [исходные_данные]    =    B#(0)
Control_code [управляющий_код]     =    'C'
Temperature [температура]                =    120
End [конец]                        =    False
```

Если вы описываете переменные как UDT, то начальными значениями таких переменных являются значения, заданные вами при создании UDT.

Хранение и доступ к данным в определяемом пользователем типе данных

У вас есть доступ к отдельным элементам UDT. Вы можете использовать символические адреса (например, *Stack_1.Temperature*). Однако вы можете указывать и абсолютный адрес, по которому расположен элемент (пример: если *Stack_1* расположен в DB20, начиная с байта 0, то абсолютный адрес для *amount* – это *DB20.DBW0* и адрес для *temperature* – это *DB20.DBD6*).

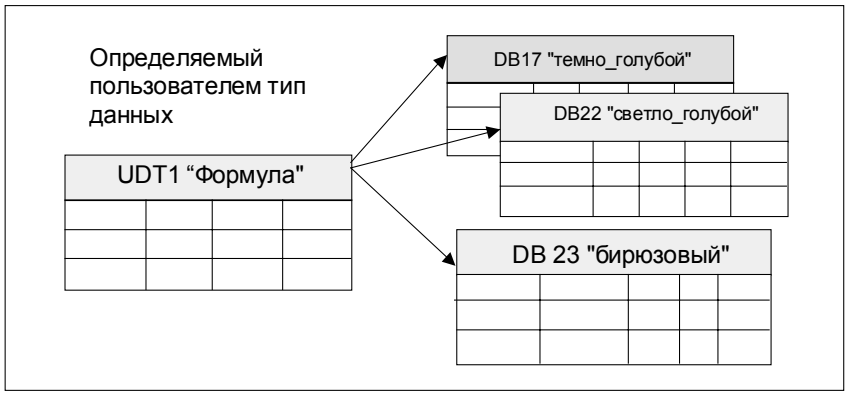
Использование определяемых пользователем типов данных в качестве параметров

Вы можете передавать переменные типа данных UDT в качестве параметров. Если параметр описывается как UDT в разделе описания переменных, то вы должны передавать UDT с той же самой же структурой. Однако параметру может присваиваться также элемент UDT, когда вы вызываете блок, если элемент UDT соответствует типу данных параметра.

Преимущества DB с назначенным ему UDT

Используя созданный вами однажды UDT, вы можете генерировать большое количество блоков данных с той же самой структурой. Вы можете потом использовать эти блоки данных для ввода различных фактических значений в конкретных задачах.

Например, если вы создаете UDT для формулы (например, для смешивания красок), вы можете поставить этот UDT в соответствие нескольким DB, каждый из которых содержит различные пропорции.



Структура блока данных определяется соответствующим ему UDT.

A.3.4 Параметрические типы

A.3.4.1 Параметрические типы

Кроме элементарных и составных типов данных, вы можете также определять параметрические типы для формальных параметров, передаваемых между блоками. STEP 7 распознает следующие параметрические типы:

- TIMER или COUNTER: определяет конкретный таймер или конкретный счетчик, который будет использоваться во время выполнения блока. Если вы снабжаете формальный параметр типа TIMER или COUNTER значением, то соответствующий фактический параметр должен быть таймером или счетчиком, другими словами, вы вводите "T" или "C" с последующим положительным целым числом.
- BLOCK: определяет конкретный блок, используемый как вход или выход. Описание этого параметра определяет используемый тип блока (FB, FC, DB и т.д.). Когда вы снабжаете формальный параметр типа BLOCK значением, задавайте в качестве фактического параметра адрес блока. Пример: ”FC101" (при использовании абсолютной адресации) или ”Valve" (при символической адресации).
- POINTER: указывает адрес переменной. Указатель содержит адрес вместо значения. Когда вы снабжаете формальный параметр типа POINTER значением, задавайте в качестве фактического параметра адрес. В STEP 7 вы можете задавать указатель в формате указателя или просто как адрес (например, M 50.0). Пример формата указателя для адресации данных, начинающихся с M 50.0: P#M50.0

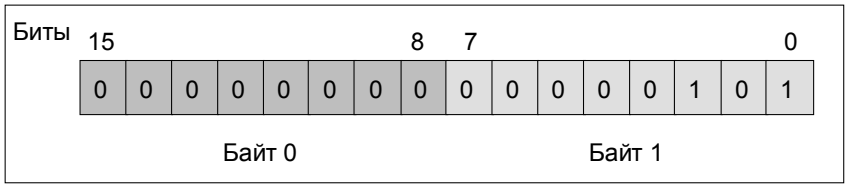
- ANY: используется, когда тип данных фактического параметра неизвестен или когда можно использовать любой тип данных. Для получения дополнительной информации о типе параметра ANY, обратитесь к разделам "Формат параметрического типа ANY" и "Использование параметрического типа ANY".

Параметрический тип может использоваться также в определяемом пользователем типе данных (UDT). Для получения дополнительной информации об UDT, обратитесь к разделу "Использование определяемых пользователем типов данных для доступа к данным".

Параметр	Емкость	Описание
TIMER	2 байта	Обозначает таймер, используемый программой в вызываемом логическом блоке. Формат: T1
COUNTER	2 байта	Обозначает счетчик, используемый программой в вызываемом логическом блоке. Формат: C10
BLOCK_FB BLOCK_FC BLOCK_DB BLOCK_SDB	2 байта	Обозначает блок, используемый программой в вызываемом логическом блоке. Формат: FC101 DB42
POINTER	6 байтов	Обозначает адрес. Формат: P#M50.0
ANY	10 байтов	Используется, когда тип данных текущего параметра неизвестен. Формат: P#M50.0 BYTE 10 P#M100.0 WORD 5

A.3.4.2 Формат параметрических типов BLOCK, COUNTER, TIMER

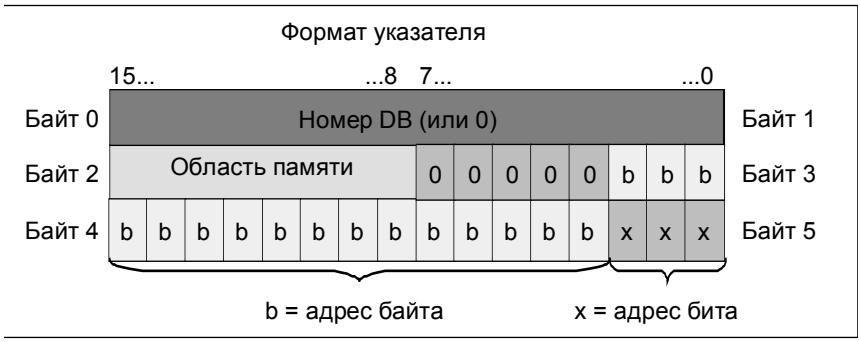
STEP 7 хранит параметрические типы BLOCK, COUNTER и TIMER как двоичные числа в слове (32 бита). Следующий рисунок показывает формат этих параметрических типов.



Допустимое число блоков, таймеров и счетчиков зависит от типа вашего CPU S7. Вы найдете подробную информацию о допустимом числе таймеров и счетчиков и максимальном числе имеющихся в распоряжении блоков в спецификациях вашего CPU в руководстве "S7-300 Programmable Controller, Hardware and Installation [Программируемый контроллер S7-300: Аппаратные средства и монтаж]" или в руководстве "S7-400, M7-400 Programmable Controllers, Hardware and Installation [Программируемые контроллеры S7-400, M7-400: Аппаратные средства и монтаж]".

A.3.4.3 Формат параметрического типа POINTER

Следующий рисунок показывает тип данных, хранимых в каждом байте.



Параметрический тип POINTER хранит следующую информацию:

- Номер DB (или 0, если данные хранятся не в DB)
- Область памяти CPU (следующая таблица показывает шестнадцатеричные коды областей памяти для параметрического типа POINTER)

Шестнадцатеричный код	Область памяти	Описание
b#16#81	I	Область входов
b#16#82	Q	Область выходов
b#16#83	M	Область памяти с побитовым доступом
b#16#84	DB	Блок данных
b#16#85	DI	Экземплярный блок данных
b#16#86	L	Локальные данные (L-стек)
b#16#87		Предыдущие локальные данные

- Адрес данных (в формате Байт.Бит)

  STEP 7 обеспечивает следующий формат указателя: p#область_памяти адрес_байт.бит. (Если формальный параметр был описан как параметр типа POINTER, то вам нужно указать лишь область памяти и адрес. STEP 7 автоматически переформатирует ваш ввод в формат указателя.) Следующие примеры показывают, как вводится параметр типа POINTER для данных, которые начинаются с M50.0:

- P#M50.0
- M50.0 (если формальный параметр был описан как POINTER).

A.3.4.4 Использование параметрического типа POINTER

Указатель используется для указания на адрес. Преимущество этого типа адресации состоит в том, что вы можете динамически изменять адрес оператора во время обработки программы.

Указатель для косвенной адресации через память

Операторы программы, работающие с косвенной адресацией через память, состоят из кода команды, идентификатора адреса и смещения (смещение нужно задавать в квадратных скобках).

Пример указателя в формате двойного слова:

L	P#8.7	Загрузка значения указателя в аккумулятор 1.
T	MD2	Передача указателя в MD2.
A	I [MD2]"	Опрос состояния сигнала во входном бите I 8.7 и
=	Q [MD2]"	присваивание этого состояния сигнала выходному биту Q 8.7.

Указатель для адресации внутри области и с пересечением области

Операторы программы, работающие с адресацией этих типов, состоят из кода команды и следующих частей: идентификатор адреса, идентификатор адресного регистра, смещение.

Адресный регистр (AR1/2) и смещение должны указываться вместе в квадратных скобках.

Пример адресации внутри области

Указатель не содержит обозначения области памяти:

L	P#8.7	Загрузка значения указателя в аккумулятор 1.
LAR1		Загрузка указателя из аккумулятора 1 в AR1.
A	I [AR1, P#0.0]"	Опрос состояния сигнала во входном бите I 8.7 и
=	Q [AR1, P#1.1]"	присваивание этого состояния сигнала выходному биту Q 10.0.

Смещение 0.0 не оказывает влияния. Выход 10.0 вычислен как 8.7 (AR1) плюс смещение 1.1 . Результат равен 10.0, а не 9.8 (см. формат указателя).

Пример адресации с пересечением области

При адресации с пересечением области в указателе обозначается область памяти (в примере I и Q).

L	P# I8.7	Загрузка значения указателя и обозначения области памяти в аккумулятор 1.
LAR1		Загрузка области памяти I и адреса 8.7 в AR1.
L	P# Q8.7	Загрузка значения указателя и обозначения области памяти в аккумулятор 1.
LAR2		Загрузка области памяти Q и адреса 8.7 в AR2.
A	[AR1, P#0.0]"	Опрос состояния сигнала во входном бите I 8.7 и
=	[AR2, P#1.1]"	присваивание этого состояния сигнала выходному биту Q 10.0.

Смещение 0.0 не оказывает влияния. Выход 10.0 вычислен как 8.7 (AR2) плюс смещение 1.1 . Результат равен 10.0, а не 9.8 (см. формат указателя).

A.3.4.5 Блок для изменения указателя

Используя в качестве примера блок FC3 "Маршрутизация указателей", можно изменять адрес бита или байта указателя. Изменяемый указатель передается переменной "pointer [указатель]" при вызове FC (могут использоваться указатели внутри области памяти или пересекающие область памяти в формате двойного слова).

С помощью параметра "Bit_Byte" вы можете изменять адрес бита или байта указателя (0: адрес бита, 1: адрес байта). Переменная "Inc_Value" (в формате целого числа) задает число, которое должно быть прибавлено к содержимому адреса или вычтено из него. Вы можете задавать также отрицательные числа, чтобы уменьшать адрес.

В случае изменения адреса бита имеет место перенос в адрес байта (также и при уменьшении), например:

- P#M 5.3, Bit_Byte = 0, Inc_Value = 6 => P#M 6.1 или
- P#M 5.3, Bit_Byte = 0, Inc_Value = -6 => P#M 4.5.

Эта функция не влияет на информацию об области памяти указателя

FC перехватывает переполнение/потерю значимости указателя. В этом случае указатель не изменяется, и выходная переменная "RET_VAL" (возможна обработка ошибки) устанавливается в "1" (до следующей правильной обработки FC3). Это имеет место, когда:

- 1. Выбран битовый адрес и Inc_Value >7 или <-7.
- 2. Выбран адрес бита или байта, и изменение привело бы к "отрицательному" адресу байта.
- 3. Выбран адрес бита или байта, и изменение привело бы к недопустимо большому адресу байта.

Пример блока для изменения указателя в форме AWL

```
FUNCTION FC 3: BOOL
TITLE = Маршрутизация указателей
//FC3 может использоваться для изменения указателей.
AUTHOR : AUT1CS1
FAMILY : INDADDR
NAME : ADDRPOINT
VERSION : 0.0

VAR_INPUT
   Bit_Byte : BOOL ; //0: адрес бита, 1: адрес байта
Inc_Value : INT ; //приращение (если отрицательное значение =>
                  //уменьшение/если положительное значение => увеличение)
END_VAR

VAR_IN_OUT
   Pointer : DWORD ; // указатель, подлежащий изменению
END_VAR
VAR_TEMP
   Inc_Value1 : INT ; //Приращение промежуточного значения
   Pointer1 : DWORD ; //Указатель промежуточного значения
   Int_Value : DWORD ; //Вспомогательная переменная
END_VAR
BEGIN
NETWORK
TITLE =
//Блок автоматически перехватывает изменения, изменяющие информацию
//об области памяти в указателе или ведущие к "отрицательным" указателям
   SET    ; //Установка RLO в 1 и
   R      #RET_VAL; //сбросить переполнение
   L      #Pointer; //Снабдить значением указатель
   T      #Pointer1; //временного промежуточного значения
   L      #Inc_Value; // Снабдить значением приращение
   T      #Inc_Value1; //временного промежуточного значения
   A      #Bit_Byte; //Если =1, то команда для адреса байта
```

```
      JC    Byte; //Перейти к вычислению адреса байта
      L     7; //Если значение приращения > 7,
      L     #Inc_Value1;
      <I    ;
      S     #RET_VAL; //то установить RET_VAL и
      JC    End; //перейти на конец
      L     -7; //Если значение приращения < -7,
      <I    ;
      S     #RET_VAL; //то установить RET_VAL и
      JC    End; // перейти на конец
      A     L     1.3; //Если бит 4 значения = 1 (Inc_Value отрицательно),
      JC    neg; //то перейти к вычитанию битового адреса
      L     #Pointer1; //Загрузить информацию об адресе указателя
      L     #Inc_Value1; //и прибавить приращение
      +D    ;
      JU    test; //перейти к проверке на отрицательный результат
neg:  L     #Pointer1; //загрузить информацию об адресе указателя
      L     #Inc_Value1; // загрузить приращение
      NEGI  ; //Изменить знак отрицательного значения на противоположный,
      -D    ; //вычесть значение
      JU    test; //и перейти к проверке
Byte: L     0; //Начало изменения адреса байта
      L     #Inc_Value1; //Если приращение >=0, то
      <I    ;
      JC    pos; //перейти к прибавлению, в противном случае
      L     #Pointer1; //загрузить информацию об адресе указателя,
      L     #Inc_Value1; // загрузить приращение,
      NEGI  ; //изменить знак отрицательного значения на противоположный,
      SLD   3; //сдвинуть приращение на 3 разряда влево,
      –D    ; //вычесть значение
      JU    test; //и перейти к проверке.
pos:  SLD   3; //Сдвинуть приращение на 3 разряда влево
      L     #Pointer1; //Загрузить информацию об адресе указателя
      +D    ; //Прибавить приращение
test: T     #Int_Value; //Передать результаты вычислений в Int_Value
```

```
      A     L     7.3; //Если адрес байта недопустим (слишком
      S     #RET_VAL; //велик или отрицателен), то установить RET_VAL
      JC    End; //и перейти на конец,
      L     #Int_Value; //в противном случае передать результат
      T     #Pointer; //в указатель
End:  NOP   0;
END_FUNCTION
```

A.3.4.6 Формат параметрического типа ANY

STEP 7 хранит параметрический тип ANY в 10 байтах (80 битов). При построении параметрического типа ANY вы должны гарантировать, что все 80 битов заняты, потому что вызываемый блок оценивает содержимое параметра в целом. Например, если вы задаете номер DB в байте 4, то вы должны также явно задать область памяти в байте 6.

STEP 7 управляет данными элементарных и составных типов иначе, чем данными параметрического типа.

Формат ANY для типов данных

Для элементарных и составных типов данных STEP 7 хранит следующие сведения:

- Типы данных
- Коэффициент повторения
- Номер DB
- Область памяти, в которой хранится информация
- Начальный адрес данных

Коэффициент повторения идентифицирует количество данных указанного типа, передаваемых параметром типа ANY. Это означает, что вы можете задавать область данных, а также использовать массивы и структуры в сочетании с параметрическим типом ANY. STEP 7 идентифицирует массивы и структуры как количество байтов (с помощью коэффициента повторения). Например, если нужно передать 10 слов, то в качестве коэффициента повторения нужно ввести значение 20 (байтов).

Адрес хранится в формате Байт.Бит, в котором адрес байта хранится в битах с 0 по 2 в байте 7, в битах с 0 по 7 в байте 8 и в битах с 3 по 7 в байте 9. Адрес бита хранится в битах с 0 по 2 в байте 9.

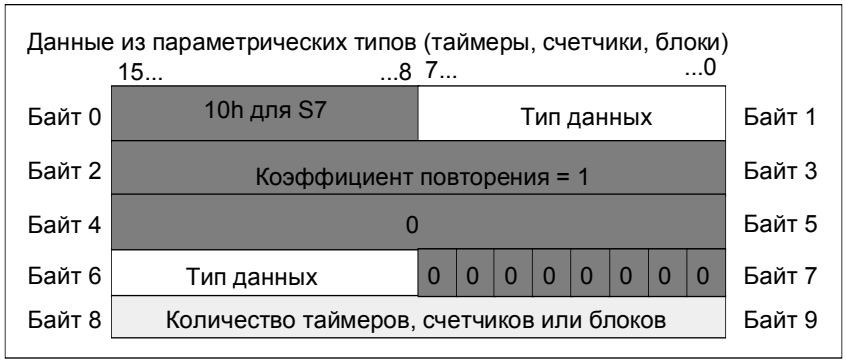
Нулевым указателем типа NIL всем байтам, начиная с байта 1, назначается 0.

Следующая таблица показывает кодирование типов данных для параметрического типа ANY.

Шестнадцатеричный код	Тип данных	Описание
b#16#00	NIL	Нулевой указатель
b#16#01	BOOL	Биты
b#16#02	BYTE	Байты (8 битов)
b#16#03	CHAR	Символы (8 битов)
b#16#04	WORD	Слова (16 битов)
b#16#05	INT	Целые числа (16 битов)
B#16#06	DWORD	Слова (32 бита)
b#16#07	DINT	Двойные целые числа (32 бита)
b#16#08	REAL	Числа с плавающей точкой (32 бита)
b#16#09	DATE	Дата
b#16#0A	TIME_OF_DAY (TOD)	Время суток
b#16#0B	TIME	Время
b#16#0C	S5TIME	Тип данных S5TIME
b#16#0E	DATE_AND_TIME (DT)	Дата и время (64 бита)
b#16#13	STRING	Строка

Формат ANY для параметрических типов

Для параметрических типов STEP 7 хранит тип данных и адрес параметров. Коэффициент повторения всегда равен 1. Байты 4, 5 и 7 всегда равны 0. Байты 8 и 9 показывают номер таймера, счетчика или блока.



Следующие таблицы показывают кодирование типов данных и областей памяти для параметрического типа ANY, используемого для параметрических типов.

Кодирование типов данных		
Шестнадцатеричный код	**Тип данных**	**Описание**
b#16#17	BLOCK_FB	Номер FB
b#16#18	BLOCK_FC	Номер FC
b#16#19	BLOCK_DB	Номер DB
b#16#1A	BLOCK_SDB	Номер SDB
b#16#1C	COUNTER	Номер счетчика
b#16#1D	TIMER	Номер таймера

Кодирование областей памяти		
Шестнадцатеричный код	**Область памяти**	**Описание**
b#16#81	I	Область входов
b#16#82	Q	Область выходов
b#16#83	M	Область памяти с побитовым доступом
b#16#84	DB	Блок данных
b#16#85	DI	Экземплярный блок данных
b#16#86	L	Локальные данные (L-стек)
b#16#87	V	Предыдущие локальные данные

A.3.4.7 Использование параметрического типа ANY

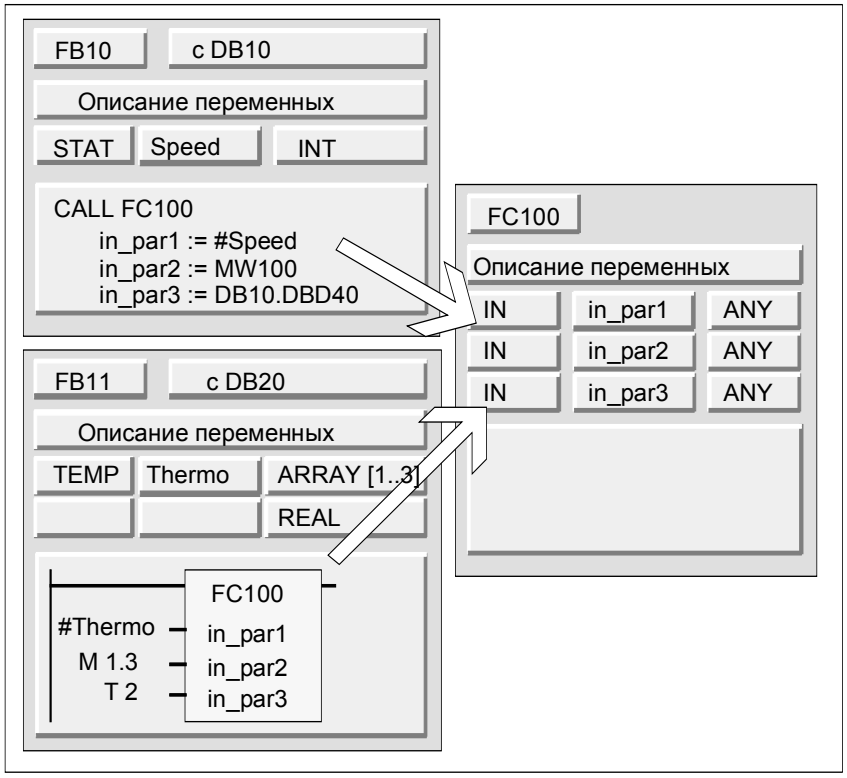
Вы можете определять для блока формальные параметры, пригодные для фактических параметров с любым типом данных. Это особенно полезно, когда тип данных фактического параметра, передаваемого при вызове блока, неизвестен или может изменяться (и когда допускается любой тип данных). В разделе описания переменных блока вы описываете этот параметр как имеющий тип данных ANY. Тогда вы можете назначать фактический параметр с любым типом данных в STEP 7.

STEP 7 выделяет 80 битов памяти для переменной с типом данных ANY. Когда вы назначаете этому формальному параметру фактический параметр, STEP 7 кодирует в 80 битах начальный адрес, тип данных и длину фактического параметра. Вызываемый блок анализирует эти 80 битов данных, сохраняемых для параметра ANY, и получает информацию, нужную для дальнейшей обработки.

Назначение параметру ANY фактического параметра

Если вы объявляете для параметра тип данных ANY, то вы можете назначать этому формальному параметру фактический параметр с любым типом данных. В STEP 7 вы можете назначать в качестве фактических параметров следующие типы данных:

- Элементарные типы данных: вы указываете абсолютный адрес или символическое имя фактического параметра.
- Составные типы данных: вы указываете символическое имя данных, относящихся к составному типу данных (например, массивы и структуры).
- Таймеры, счетчики и блоки: вы указываете номер (например, T1, C20 или FB6).
- Следующий рисунок показывает, как данные передаются в FC через параметры с типом данных ANY.



В этом примере FC100 имеет три параметра (*in_par1, in_par2, u in_par3*), описанные как тип данных ANY.

- Когда FB10 вызывает FC100, FB10 передает целое число (статическая переменная speed), слово (MW100) и двойное слово в DB10 (DB10.DBD40) .
- Когда FB11 вызывает FC10, FB11 передает массив вещественных чисел (временная переменная "Thermo"), булево значение (M 1.3) и таймер (T2).

Задание области данных для параметра ANY

Вы можете присваивать параметру ANY не только отдельные адреса (например, MW100), но можете задавать также область данных. Если вы хотите задать в качестве фактического параметра область данных, то для указания количества передаваемых данных используйте следующий формат константы:

p# *Идентификатор области памяти Байт.Бит Тип данных Коэффициент повторения*

В качестве элемента *Тип данных* вы можете указывать любые элементарные типы данных и тип данных DATE_AND_TIME в формате для константы. Если тип данных не BOOL, то нужно задавать адрес бита равным 0 (x.0). Следующая таблица показывает примеры формата для задания областей памяти, передаваемых параметру ANY:

Фактический параметр	Описание
p# M 50.0 BYTE 10	Определяет 10 байтов в области памяти с побайтовым доступом: с MB50 по MB59.
p# DB10.DBX5.0 S5TIME 3	Определяет 3 элемента данных с типом данных S5TIME, которые расположены в DB10: с байта 5 в DB по байт 10 в DB
p# Q 10.0 BOOL 4	Определяет 4 бита в области выходов: с Q 10.0 по Q 10.3.

Пример использования параметрического типа ANY

Следующий пример показывает, как вы можете скопировать область памяти размером 10 байтов, используя параметрический тип ANY и системную функцию SFC20 BLKMOV.

```
AWL                          Объяснение
FUNCTION FC10: VOID
VAR_TEMP
   Source   : ANY;            Источник
   Target   : ANY;            Цель
END_VAR
BEGIN
LAR1     P#Source;           Загрузить начальный адрес указателя ANY в AR1

L  B#16#10;                  Загрузить идентификатор синтаксиса и
T  LB[AR1,P#0.0];            передать его указателю ANY.

L  B#16#02;                  Загрузить тип данных Byte и
T  LB[AR1,P#1.0];            передать его указателю ANY.

L  10;                       Загрузить 10 байтов и
T  LW[AR1,P#2.0];            передать их указателю ANY.

L  22;                       Источником является DB22, DBB11
T  LW[AR1,P#4.0];
L  P#DBX11.0;
T  LD[AR1,P#6.0];

LAR1     P#Target;           Загрузить начальный адрес указателя ANY в AR1.

L  B#16#10;                  Загрузить идентификатор синтаксиса и
T  LB[AR1,P#0.0];            передать его указателю ANY.

L  B#16#02;                  Загрузить тип данных Byte и
T  LB[AR1,P#1.0];            передать его указателю ANY.

L  10;                       Загрузить 10 байтов и
T  LW[AR1,P#2.0];            передать их указателю ANY.

L  33;                       Адресатом является DB33, DBB202
T  LW[AR1,P#4.0];
L  P#DBX202.0;
T  LD[AR1,P#6.0];

CALL SFC 20 (                Вызвать системную функцию BLKMOV
   SRC BLK := Source,
   RET_VAL := MW 12,         Проанализировать бит BR и MW12
   DSTBLK := Target
    );
END FUNCTION
```

A.3.4.8 Назначение типов данных локальным данным логических блоков

В STEP 7 есть ограничения на типы данных (элементарные и составные типы данных и параметрические типы), которые можно назначать локальным данным блока в разделе описания переменных.

Допустимые типы данных для локальных данных OB

Следующая таблица показывает ограничения (–) на описание локальных данных для OB. Так как вы не можете вызывать OB, то у OB не может быть параметров (входных, выходных или проходных). Так как OB не имеет экземплярного DB, то вы не можете описывать какие-либо статические переменные для OB. Типами данных временных переменных OB могут быть элементарные или составные типы данных и тип данных ANY.

Допустимые назначения отмечены символом ● .

Тип описания	Элементарные типы данных	Составные типы данных	Параметрический тип	Параметрический тип	Параметрический тип	Параметрический тип	Параметрический тип
			TIMER	COUNTER	BLOCK	POINTER	ANY
Вход	—	—	—	—	—	—	—
Выход	—	—	—	—	—	—	—
Вход/Выход	—	—	—	—	—	—	—
Статический	—	—	—	—	—	—	—
Временный	●[1]	●[1]	—	—	—	—	□[1]

[1] Располагается в L–стеке OB.

Допустимые типы данных для локальных данных FB

Следующая таблица показывает ограничения (–) на описание локальных данных для FB. Благодаря экземплярному DB, при описании локальных данных для FB имеется меньшее количество ограничений. При описании входных параметров ограничений нет вообще; для выходного параметра вы не можете объявлять никакие параметрические типы, а для проходных параметров разрешены только параметрические типы POINTER и ANY. Вы можете описывать временные переменные как имеющие тип данных ANY. Все другие параметрические типы запрещены.

Допустимые назначения отмечены символом ● .

Тип описания	Элементарные типы данных	Составные типы данных	Параметрический тип	Параметрический тип	Параметрический тип	Параметрический тип	Параметрический тип
			TIMER	**COUNTER**	**BLOCK**	**POINTER**	**ANY**
Вход	●	●	●	●	●	●	●
Выход	●	●	—	—	—	—	—
Вход/Выход	●	●[1]	—	—	—	●	●
Статический	●	●	—	—	—	—	—
Временный	●[2]	●[2]	—	—	—	—	●[2]

[1] Хранится как ссылка (48-битный указатель) в экземплярном блоке данных.
[2] Располагается в L–стеке FB.

Допустимые типы данных для локальных данных FC

Следующая таблица показывает ограничения (–) на описание локальных данных для FC. Так как FC не имеет экземплярного DB, то он не имеет также статических переменных. Для входных, выходных и проходных параметров разрешены только параметрические типы POINTER и ANY. Вы можете также описывать временные переменные с параметрическим типом ANY.

Допустимые назначения отмечены символом ● .

Тип описания	Элементарные типы данных	Составные типы данных	Параметрический тип	Параметрический тип	Параметрический тип	Параметрический тип	Параметрический тип
			TIMER	**COUNTER**	**BLOCK**	**POINTER**	**ANY**
Вход	●	●	●	●	●	●	●
Выход	●	●	—	—	—	●	●
Вход/Выход	●	●	—	—	—	●	●
Статический	—	—	—	—	—	—	—
Временный	●[1]	●[1]	—	—	—	—	●[1]

[1] Располагается в L–стеке FC.

A.3.4.9 Разрешенные типы данных при передаче параметров

Правила передачи параметров между блоками

Когда вы назначаете формальным параметрам фактические параметры, вы можете указывать или абсолютный адрес, или символическое имя, или константу. STEP 7 ограничивает допустимые назначения для различных параметров. Например, выходным и проходным (in/out) параметрам не может назначаться постоянное значение (так как целью выходного или проходного параметра является изменение его значения). Эти ограничения особенно относятся к параметрам с составными типами данных, которым не может назначаться ни абсолютный адрес, ни константа.

Следующие таблицы показывают эти ограничения (–), включая типы данных фактических параметров, назначаемых формальным параметрам.

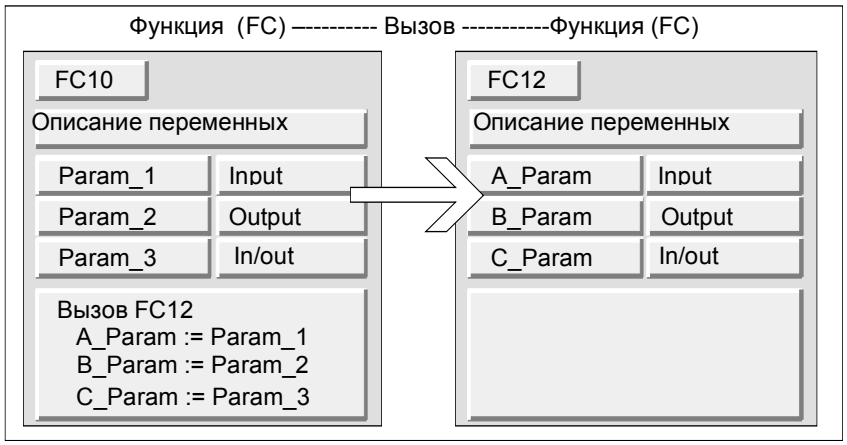
Допустимые назначения отмечены символом ● .

Элементарные типы данных				
Тип описания	**Абсолютный адрес**	**Символическое имя (в таблице символов)**	**Временный локальный символ**	**Константа**
Вход	●	●	●	●
Выход	●	●	●	—
Вход/Выход	●	●	●	—

Составные типы данных				
Тип описания	**Абсолютный адрес**	**Символическое имя элемента DB (в таблице символов)**	**Временный локальный символ**	**Константа**
Вход	—	●	●	—
Выход	—	●	●	—
Вход/Выход	—	●	●	—

Допустимые типы данных при вызове функции функцией

Вы можете назначать формальным параметрам вызываемого FC формальные параметры вызывающего FC. Следующий рисунок показывает формальные параметры FC10, назначаемые в качестве фактических параметров формальным параметрам FC12.



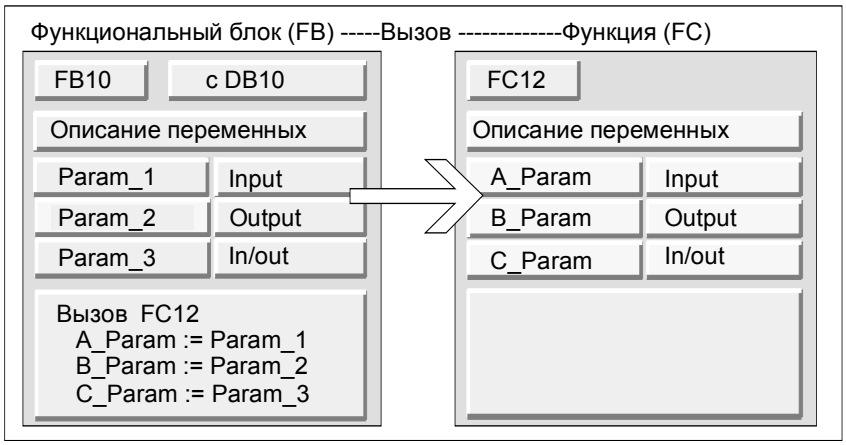
STEP 7 ограничивает назначение формальных параметров одного FC в качестве фактических параметров формальным параметрам другого FC. Вы не можете, например, назначать параметры с составными типами данных или с параметрическим типом в качестве фактического параметра.

Следующая таблица показывает разрешенные типы данных (●), когда один FC вызывает другой FC.

Тип описания	Элементарные типы данных	Составные типы данных	Параметрический тип	Параметрический тип	Параметрический тип	Параметрический тип	Параметрический тип
			TIMER	COUNTER	BLOCK	POINTER	ANY
Вход → Вход	●	—	—	—	—	—	—
Вход → Выход	—	—	—	—	—	—	—
Вход → Вход/Выход	—	—	—	—	—	—	—
Выход → Вход	—	—	—	—	—	—	—
Выход → Выход	●	—	—	—	—	—	—
Выход → Вход/Выход	—	—	—	—	—	—	—
Вход/Выход → Вход	●	—	—	—	—	—	—
Вход/Выход → Выход	●	—	—	—	—	—	—
Вход/Выход → Вход/Выход	●	—	—	—	—	—	—

Допустимые типы данных при вызове функции функциональным блоком

Вы можете назначать формальным параметрам вызываемой FC формальные параметры вызывающего FB. Следующий рисунок показывает формальные параметры FB10, назначаемые в качестве фактических параметров формальным параметрам FC12.

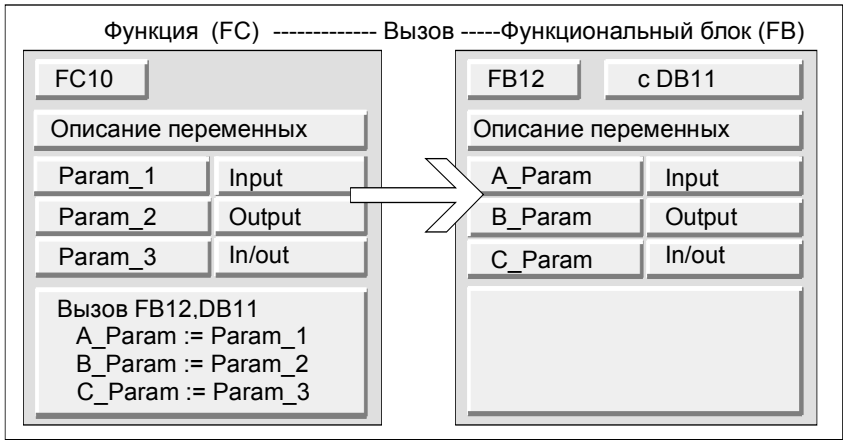


STEP 7 ограничивает назначение формальных параметров FB формальным параметрам FC. Вы не можете, например, назначать параметры, имеющие параметрический тип, в качестве фактических параметров. Следующая таблица показывает разрешенные типы данных (●), когда FB вызывает FC.

Тип описания	Элементарные типы данных	Составные типы данных	Параметрический тип	Параметрический тип	Параметрический тип	Параметрический тип	Параметрический тип
			TIMER	COUNTER	BLOCK	POINTER	ANY
Вход → Вход	●	●	—	—	—	—	—
Вход → Выход	—	—	—	—	—	—	—
Вход → Вход/Выход	—	—	—	—	—	—	—
Выход → Вход	—	—	—	—	—	—	—
Выход → Выход	●	●	—	—	—	—	—
Выход → Вход/Выход	—	—	—	—	—	—	—
Вход/Выход → Вход	●	—	—	—	—	—	—
Вход/Выход → Выход	●	—	—	—	—	—	—
Вход/Выход → Вход/Выход	●	—	—	—	—	—	—

Допустимые типы данных при вызове функционального блока функцией

Вы можете назначать формальным параметрам вызываемого FB формальные параметры вызывающей FC. Следующий рисунок показывает формальные параметры FC10, назначаемые в качестве фактических параметров формальным параметрам FB12



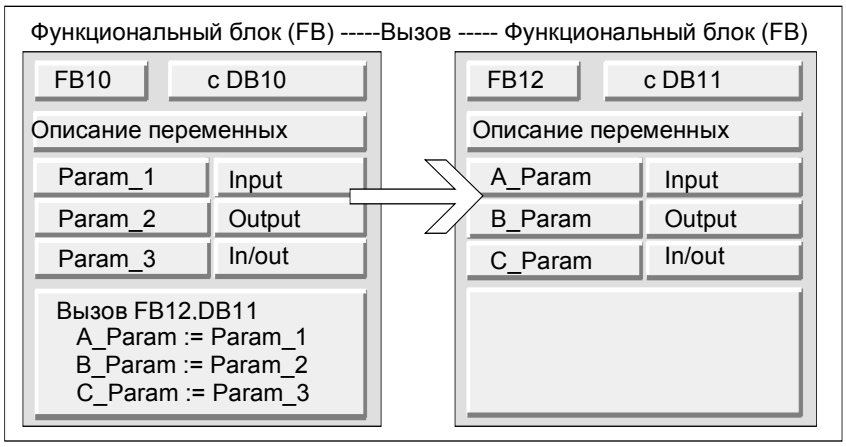
STEP 7 ограничивает назначение формальных параметров FC формальным параметрам FB. Вы не можете, например, назначать параметры с составным типом данных в качестве фактических параметров. Однако вы можете назначать входные параметры, имеющие параметрический тип TIMER, COUNTER или BLOCK, входным параметрам вызываемого FB.

Следующая таблица показывает разрешенные типы данных (●), когда FC вызывает FB.

Тип описания	Элементарные типы данных	Составные типы данных	Параметрический тип	Параметрический тип	Параметрический тип	Параметрический тип	Параметрический тип
			TIMER	**COUNTER**	**BLOCK**	**POINTER**	**ANY**
Вход → Вход	●	—	●	●	●	—	—
Вход → Выход	—	—	—	—	—	—	—
Вход → Вход/Выход	—	—	—	—	—	—	—
Выход → Вход	—	—	—	—	—	—	—
Выход → Выход	●	—	—	—	—	—	—
Выход → Вход/Выход	—	—	—	—	—	—	—
Вход/Выход → Вход	●	—	—	—	—	—	—
Вход/Выход → Выход	●	—	—	—	—	—	—
Вход/Выход → Вход/Выход	●	—	—	—	—	—	—

Допустимые типы данных при вызове функционального блока функциональным блоком

Вы можете назначать формальным параметрам вызываемого FB формальные параметры вызывающего FB. Следующий рисунок показывает формальные параметры FB10, назначаемые в качестве фактических параметров формальным параметрам FB12.



STEP 7 ограничивает назначение формальных параметров одного FB формальным параметрам другого FB. Вы не можете, например, назначать входные и выходные параметры с составными типами данных в качестве фактических параметров входным и выходным параметрам вызываемого FB. Однако вы можете назначать входные параметры, имеющие параметрический тип TIMER, COUNTER или BLOCK, входным параметрам вызываемого FB.

Следующая таблица показывает разрешенные типы данных (●), когда один FB вызывает другой FB.

Тип описания	Элементарные типы данных	Составные типы данных	Параметрический тип	Параметрический тип	Параметрический тип	Параметрический тип	Параметрический тип
			TIMER	**COUNTER**	**BLOCK**	**POINTER**	**ANY**
Вход → Вход	●	●	●	●	●	—	—
Вход → Выход	—	—	—	—	—	—	—
Вход → Вход/Выход	—	—	—	—	—	—	—
Выход → Вход	—	—	—	—	—	—	—
Выход → Выход	●	●	—	—	—	—	—
Выход → Вход/Выход	—	—	—	—	—	—	—
Вход/Выход → Вход	●	—	—	—	—	—	—
Вход/Выход → Выход	●	—	—	—	—	—	—
Вход/Выход → Вход/Выход	●	—	—	—	—	—	—

A.3.4.10 Передача параметрам IN_OUT функционального блока

Если параметрам IN_OUT функционального блока (FB) передаются составные типы данных, то передается адрес переменной (вызов по ссылке).

Если параметрам IN_OUT функционального блока (FB) передаются элементарные типы данных, то значения копируются в экземплярный блок данных перед запуском функционального блока и копируются из экземплярного блока данных после завершения функционального блока.

Это означает, что переменные IN_OUT с элементарным типом данных можно инициализировать значением.

Однако, при вызове невозможно вместо переменной IN_OUT задавать в качестве фактического параметра константу, потому что невозможно производить запись в константу.

Переменные с типом данных STRUCT или ARRAY невозможно инициализировать, потому что в этом случае в экземплярном блоке данных находится только один адрес.

A.4 Работа с более старыми проектами

A.4.1 Преобразование проектов версии 1

Вы можете повторно использовать проекты, которые вы создавали с помощью STEP 7 версии 1. Для этого вам нужно преобразовать проекты версии 1 в проекты версии 2.

Сохраняются следующие компоненты проекта версии 1:

- Структура проекта с программами
- Блоки
- Исходные файлы на AWL
- Таблица символов

Конфигурация аппаратных средств не преобразуется. Вы можете копировать компоненты программы, содержащиеся в проекте, в другие проекты. Вы можете также добавить в новый проект станцию, сконфигурировать и назначить ей параметры.

Как только вы выполнили преобразование в проект версии 2, вы можете в диалоговом окне принимать решение о том, хотите ли вы теперь преобразовать этот проект версии 2 в проект текущей версии вашего STEP 7.

Примечание

Отдельные блоки по своим свойствам остаются такими же, как блоки версии 1. Код, сгенерированный в версии 1, не изменяется, и поэтому эти блоки не могут использоваться совместно с мультиэкземплярами.

Если вы хотите описать в преобразованных блоках мультиэкземпляры, то сначала сгенерируйте из преобразованных блоков исходные файлы на AWL, используя приложение "LAD/STL/FBD: Programming Blocks[KOP/AWL/FUP: Программирование блоков]", а затем скомпилируйте их обратно в блоки.

Программирование мультиэкземпляров – это новое свойство STEP 7 версии 2, используемое для создания функциональных блоков (FB). Если вы хотите продолжать использовать функциональные блоки, созданные с помощью версии 1, прежним образом в проекте версии 2, то вам не нужно преобразовывать их.

Последовательность действий

Для преобразования проектов версии 1 действуйте следующим образом:

1. Выберите команду меню **File > Open Version 1 Project [Файл > Открыть проект версии 1]**.
2. В появляющемся диалоговом окне выберите проект версии 1, который вы хотите использовать в проекте версии 2. Вы распознаете проект версии 1 по его расширению *.s7a.
3. Затем в следующем диалоговом окне введите имя нового проекта, в который вы хотите преобразовать проект версии 1.

A.4.2 Преобразование проектов версии 2

В STEP 7 вы можете также открывать проекты версии 2, используя команду меню **File > Open [Файл > Открыть]**.

Проекты/библиотеки версии 2 можно преобразовать (перенести) в проект текущей версии вашего STEP 7, используя команду меню **File > Save As [Файл > Сохранить как…]** и опцию "Rearrange before saving [Переупорядочить перед сохранением]". Тогда проект сохраняется как проект текущей версии STEP 7.

Вы можете редактировать проекты и библиотеки из более старых версий STEP 7, поддерживая их формат, и сохранять их, выбирая в качестве типа файла более старую версию STEP 7 в диалоговом окне "Save Project As [Сохранить проект как…]". Например, чтобы редактировать объекты с помощью STEP 7 версии 2.1, выберите здесь "Project 2.x" или "Library 2.x".

Обозначение типа файла

	STEP 7 V3	от STEP 7 V4
Тип файла текущей версии	Project3.x Library3.x	Project Library
Тип файла более старой версии	Project2.x Library2.x	Project2.x Library2.x

Это означает, что вы имеете доступ только к области функций более старой версии STEP 7. Однако вы все еще можете продолжать управлять проектами и библиотеками с помощью более старой версии STEP 7.

Примечание

Переход от версии 3 к версии 4 и выше влечет за собой только изменение имени, а формат остается идентичным. Поэтому в STEP 7 V4 нет файла типа "Project3.x".

Последовательность действий

Чтобы преобразовать проект версии 2 в проект в формате текущей версии STEP 7, действуйте следующим образом:

1. Выполните для проекта команду ”Save As [Сохранить как…]" в меню File [Файл] с опцией "Rearrange before saving [Переупорядочить перед сохранением]".
2. Выберите тип файла "Project" в диалоговом окне "Save Project As [Сохранить проект как…]" и нажмите кнопку "Save [Сохранить]".

Чтобы преобразовать проект версии 2, сохраняя его формат, в проект текущей версии STEP 7 действуйте следующим образом:

1. Выполните вышеупомянутый шаг 1 в случае необходимости.
2. Выберите тип файла более старой версии STEP 7 в диалоговом окне "Save Project As [Сохранить проект как…]" и нажмите кнопку "Save [Сохранить]".

A.4.3 Замечания к проектам STEP 7 V.2.1 со связью через глобальные данные

- Если вы хотите преобразовать проект с глобальными данными из STEP 7 V2.1 в STEP 7 V5, то вам нужно сначала с помощью STEP 7 V5.0 открыть таблицу глобальных данных (GD) в проекте STEP 7 V2.1. Данные связи, сконфигурированные прежде, автоматически преобразуются в новую структуру через GD-связь.
- Когда вы архивируете проекты STEP 7 V2.1, более старые программы (ARJ, PKZIP...) могут выдавать сообщение об ошибке, если проект содержит файлы с именами длиной более восьми символов. Такое сообщение появляется также тогда, когда Network MPI в проекте STEP 7 V2.1 была отредактирована с идентификатором, имеющим длину более восьми символов. В проектах STEP 7 V2.1 с глобальными данными прежде, чем начинать в первый раз конфигурировать связь через глобальные данные, отредактируйте имя сети MPI, которое должно быть длиной максимум восемь символов.
- Если вы хотите переименовать проект STEP 7 V2.1, то вы должны переназначить заголовки столбцов (CPU) в GD-таблице, выбирая заново соответствующие CPU. Если вы восстанавливаете старое имя проекта, то назначения отображаются еще раз.

A.5 Типовые программы

A.5.1 Типовые проекты и типовые программы

Установочный CD содержит ряд типовых проектов. Для проектов, не описанных в этой главе, описание включено в соответствующий OB1.

Примеры и типовые проекты	Включено в CD	Описано в этой главе	Описание в OB1
Проект "S7-Mix" (промышленный процесс смешивания)	•	•	
Проект "GS-*" (первые шаги и упражнения)	•	Отдельное руководство	•
Проект "S7-Zebra" (управление светофором на пешеходном переходе «зебра»)	•		•
Проект "COM_SFB" (обмен данными между двумя CPU S7-400)	•		•
Проекты "COM_SFC1" и "COM_SFC2" (обмен данными с использованием коммуникационных SFC для не конфигурированных соединений)	•		•
Пример обработки прерываний по времени		•	
Пример обработки прерываний с задержкой		•	
Пример маскирования и демаскирования синхронных ошибок		•	
Пример блокировки и разблокировки прерываний и асинхронных ошибок		•	
Пример задержанной обработки прерываний и асинхронных ошибок		•	

Акцент в примерах сделан не на обучении особенностям стиля программирования или специальным знаниям, необходимым для управления специфическим процессом. Примеры рассчитаны просто на то, чтобы иллюстрировать шаги, которым нужно следовать при проектировании программы.

Удаление и установка поставляемых типовых проектов

В SIMATIC Manager можно удалять, а затем повторно устанавливать поставляемые типовые проекты. Для установки типовых проектов вам нужно запустить программу установки STEP 7 V5.0. Типовые проекты можно устанавливать выборочно в более позднее время.

Примечание

Поставляемые типовые проекты копируются во время установки STEP 7, если не указано иное. Если вы отредактировали поставляемые типовые проекты, то во время повторной установки STEP 7 поверх этих измененных проектов записываются оригиналы.

По этой причине перед выполнением каких-либо изменений вы должны скопировать поставляемые типовые проекты и потом только редактировать копии.

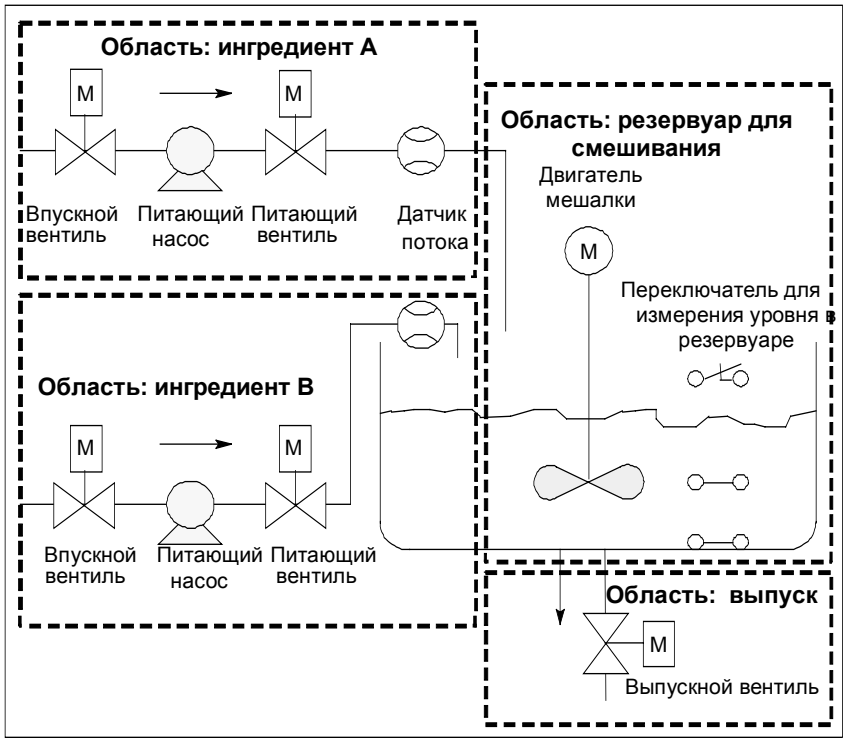
A.5.2 Типовая программа для промышленного процесса смешивания

A.5.2.1 Типовая программа для промышленного процесса смешивания

Типовая программа использует информацию об управлении промышленным процессом смешивания, которую вы уже получили в части 1 данного руководства.

Задача

Два ингредиента (ингредиент A и ингредиент B) смешиваются в резервуаре для смешивания посредством мешалки. Готовый продукт сливается из резервуара через выпускной вентиль. Следующий рисунок показывает схему этого типового процесса.



Описание частей процесса

Часть 1 руководства включает описание того, как типовой процесс подразделяется на функциональные области и отдельные задачи. Отдельные области описаны ниже.

Области ингредиентов A и B:

- Трубы для каждого ингредиента оборудованы впускным вентилем, питающим вентилем и питающим насосом.
- Впускные трубы имеют также датчики потока.
- Включение питающих насосов должно блокироваться, когда датчик уровня резервуара показывает, что резервуар полон.
- Запуск питающих насосов должен блокироваться, когда выпускной вентиль открыт.
- Впускной и питающий вентили должны открываться, самое раннее, через 1 секунду после запуска питающего насоса.
- Вентили должны закрываться немедленно после останова питающих насосов (сигнал датчика потока), чтобы предотвратить просачивание ингредиентов из насоса.
- Запуск питающих насосов объединен с функцией контроля времени, иными словами, в течение 7 секунд после запуска насосов датчик потока должен сообщить о наличии потока.
- Питающие насосы должны выключаться настолько быстро, насколько возможно, если датчик потока больше не сигнализирует о наличии потока в то время, как питающие насосы работают.
- Количество запусков питающих насосов должно подсчитываться (интервал технического обслуживания).

Область резервуара для смешивания:

- Запуск электродвигателя мешалки должен блокироваться, когда датчик уровня резервуара показывает «уровень ниже минимума» или открыт выпускной вентиль.
- Электродвигатель мешалки после достижения номинальной скорости посылает ответный сигнал. Если этот сигнал не принимается в течение 10 секунд после запуска электродвигателя, то электродвигатель должен быть выключен.
- Количество запусков электродвигателя мешалки должно подсчитываться (интервал технического обслуживания).
- В резервуаре для смешивания должны устанавливаться три датчика:
  - Резервуар полон: нормально замкнутый контакт. Этот контакт размыкается, когда достигается максимальный уровень резервуара.
  - Уровень резервуара выше минимума: нормально разомкнутый контакт. Этот контакт замыкается, когда достигается минимальный уровень резервуара.
  - Резервуар непустой: нормально разомкнутый контакт. Этот контакт замкнут, если резервуар непустой.

Область выпуска:

- Выпуск из резервуара контролируется электромагнитным вентилем.
- Электромагнитный вентиль управляется оператором, но должен закрываться снова, самое позднее, когда генерируется сигнал «резервуар пуст».
- Открытие выпускного вентиля блокируется, когда
  - работает электродвигатель мешалки
  - резервуар пуст.

Станция оператора

Чтобы дать возможность оператору запускать, останавливать и контролировать процесс, требуется также станция оператора. Станция оператора оборудована следующим:

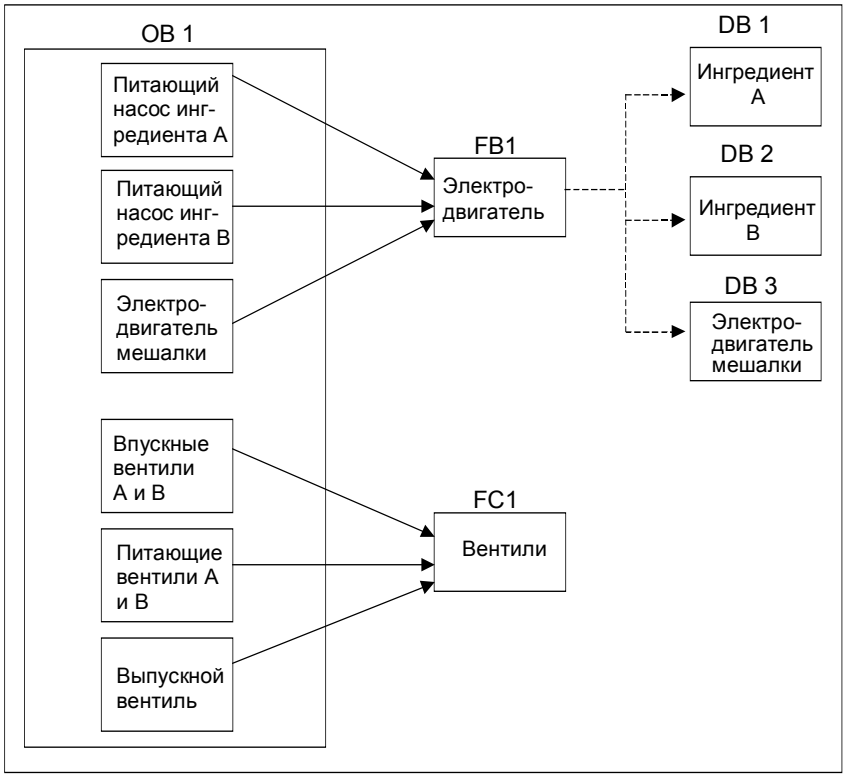
- Переключатели управления наиболее важными стадиями процесса. С помощью переключателя "сброс отображения технического обслуживания" вы можете выключать лампы отображения технического обслуживания для электродвигателей, подлежащих техническому обслуживанию, и устанавливать соответствующие счетчики интервала технического обслуживания в 0.
- Лампы устройства отображения для индикации состояния процесса.
- Выключатель аварийного останова.

A.5.2.2 Определение логических блоков

Вы структурируете программу, распределяя программу пользователя по различным блокам и устанавливая иерархию вызовов блоков.

Иерархия вызовов блоков

Следующий рисунок показывает иерархию блоков, вызываемых в структурированной программе.



- OB1: Образует интерфейс с операционной системой CPU и содержит основную программу. В OB1 вызываются блоки FB1 и FC1 и передаются специальные параметры, требуемые для управления процессом.
- FB1: Питающий насос для ингредиента A, питающий насос для ингредиента B и электродвигатель мешалки могут управляться одним функциональным блоком, потому что требования (включение, выключение, подсчет и т.д.) идентичны.
- Экземплярные DB 1-3: Фактические параметры и статические данные для управления питающими насосами для ингредиента A, ингредиента B и для электродвигателя мешалки различаются и поэтому сохраняются в трех экземплярных DB, связанных с FB1.
- FC1: Впускные и питающие вентили для ингредиентов A и B и выпускной вентиль тоже используют общий логический блок. Поскольку должна программироваться только функция "открыть и закрыть", то достаточно одного единственного FC.

A.5.2.3 Назначение символических имен

Определение символических имен

Символы используются в типовой программе, и они должны быть определены в таблице символов с помощью STEP 7. Следующие таблицы показывают символические имена и абсолютные адреса элементов, используемых в программе.

Символические адреса для питающего насоса, электродвигателя мешалки и впускных вентилей

Символическое имя	Адрес	Тип данных	Описание
Feed_pump_A_start	I0.0	BOOL	Запускает питающий насос для ингредиента A
Feed_pump_A_stop	I0.1	BOOL	Останавливает питающий насос для ингредиента A
Flow_A	I0.2	BOOL	Ингредиент A поступает
Inlet_valve_A	Q4.0	BOOL	Включает впускной вентиль для ингредиента A
Feed_valve_A	Q4.1	BOOL	Включает питающий вентиль для ингредиента A
Feed_pump_A_on	Q4.2	BOOL	Лампа «питающий насос ингредиента A работает»
Feed_pump_A_off	Q4.3	BOOL	Лампа «питающий насос ингредиента A не работает»
Feed_pump_A	Q4.4	BOOL	Включает питающий насос для ингредиента A
Feed_pump_A_fault	Q4.5	BOOL	Лампа «питающий насос A неисправен»
Feed_pump_A_maint	Q4.6	BOOL	Лампа «ремонт питающего насоса A»
Feed_pump_B_start	I0.3	BOOL	Запускает питающий насос для ингредиента B
Feed_pump_B_stop	I0.4	BOOL	Останавливает питающий насос для ингредиента B
Flow_B	I0.5	BOOL	Ингредиента B поступает
Inlet_valve_B	Q5.0	BOOL	Включает впускной вентиль для ингредиента B
Feed_valve_B	Q5.1	BOOL	Включает питательный вентиль для ингредиента B
Feed_pump_B_on	Q5.2	BOOL	Лампа «питающий насос ингредиента B работает»
Feed_pump_B_off	Q5.3	BOOL	Лампа «питающий насос ингредиента B не работает»
Feed_pump_B	Q5.4	BOOL	Включает питающий насос для ингредиента B
Feed_pump_B_fault	Q5.5	BOOL	Лампа «питающий насос B неисправен»
Feed_pump_B_maint	Q5.6	BOOL	Лампа «ремонт питающего насоса B»
Agitator_running	I1.0	BOOL	Ответный сигнал электродвигателя мешалки
Agitator_start	I1.1	BOOL	Кнопка запуска мешалки
Agitator_stop	I1.2	BOOL	Кнопка останова мешалки
Agitator	Q8.0	BOOL	Запускает мешалку
Agitator_on	Q8.1	BOOL	Лампа «мешалка работает»
Agitator_off	Q8.2	BOOL	Лампа «мешалка не работает»
Agitator_fault	Q8.3	BOOL	Лампа «электродвигатель мешалки неисправен»

Символические адреса для питающего насоса, электродвигателя мешалки и впускных вентилей			
Символическое имя	**Адрес**	**Тип данных**	**Описание**
Agitator_maint	Q8.4	BOOL	Лампа «ремонт электродвигателя мешалки»

Символические адреса для датчиков и отображения уровня резервуара			
Символическое имя	**Адрес**	**Тип данных**	**Описание**
Tank_below_max	I1.3	BOOL	Датчик «резервуар для смешивания неполон»
Tank_above_min	I1.4	BOOL	Датчик «уровень резервуара для смешивания выше минимума»
Tank_not_empty	I1.5	BOOL	Датчик «резервуар для смешивания не пуст»
Tank_max_disp	Q9.0	BOOL	Лампа «резервуар для смешивания полон»
Tank_min_disp	Q9.1	BOOL	Лампа «уровень резервуара для смешивания ниже минимума»
Tank_empty_disp	Q9.2	BOOL	Лампа «резервуар для смешивания пуст»

Символические адреса для выпускного вентиля			
Символическое имя	**Адрес**	**Тип данных**	**Описание**
Drain_open	I0.6	BOOL	Кнопка открытия выпускного вентиля
Drain_closed	I0.7	BOOL	Кнопка закрытия выпускного вентиля
Drain	Q9.5	BOOL	Запускает выпускной вентиль
Drain_open_disp	Q9.6	BOOL	Лампа «выпускной вентиль открыт»
Drain_closed_disp	Q9.7	BOOL	Лампа «выпускной вентиль закрыт»

Символические адреса для других элементов программы			
Символическое имя	**Адрес**	**Тип данных**	**Описание**
EMER_STOP_off	I1.6	BOOL	Выключатель АВАРИЙНЫЙ ОСТАНОВ
Reset_maint	I1.7	BOOL	Переключатель сброса ламп технического обслуживания всех электродвигателей
Motor_block	FB1	FB1	FB для управления насосами и электродвигателем
Valve_block	FC1	FC1	FC для управления вентилями
DB_feed_pump_A	DB1	FB1	Экземплярный DB для управления питающим насосом A
DB_feed_pump_B	DB2	FB1	Экземплярный DB для управления питающим насосом B
DB_agitator	DB3	FB1	Экземплярный DB для управления электродвигателем мешалки

A.5.2.4 Создание FB электродвигателя

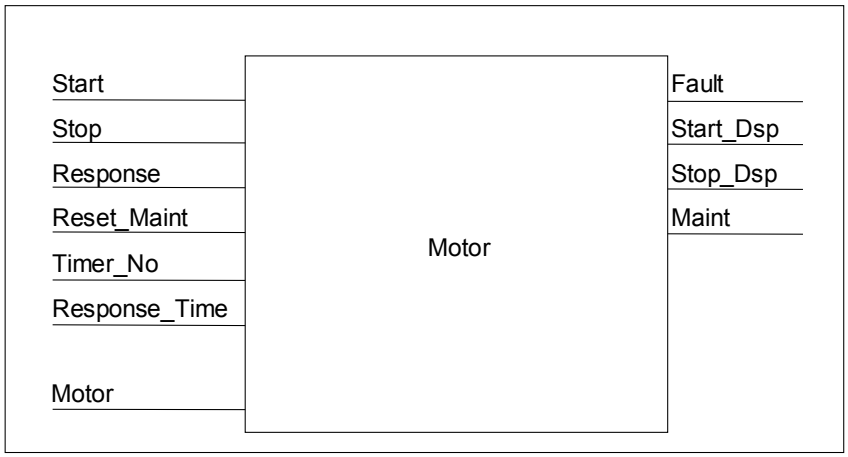
Что требуется от FB?

FB электродвигателя содержит следующие логические функции:

- Имеются вход запуска и вход останова.
- Ряд блокировок разрешает работу устройств (насосов и электродвигателя мешалки). Состояние блокировок хранится во временных локальных данных (L-стек) OB1 (”Motor_enable", ”Valve_enable") и логически объединяется со входами запуска и останова, когда FB для электродвигателя обрабатывается.
- Сигнал обратной связи от устройств должен появляться в течение заданного времени. В противном случае предполагается, что произошла ошибка или отказ. Тогда эта функция останавливает электродвигатель.
- Должны задаваться момент времени и длительность ответного сигнала или период ошибки/отказа.
- Если нажимается кнопка запуска и электродвигатель разблокирован, то устройство самостоятельно включается и работает до тех пор, пока не нажата кнопка останова.
- Когда включается устройство, запускается таймер. Если ответный сигнал устройства не будет принят прежде, чем истечет время таймера, то устройство останавливается.

Спецификация входов и выходов

Следующий рисунок показывает входы и выходы общего FB для электродвигателя.



Определение параметров FB

Если вы используете мультиэкземплярный FB электродвигателя (для управления как насосами, так и электродвигателем), то вы должны определить общие имена параметров для входов и выходов.

FB электродвигателя в типовом процессе требует следующего:

- Он должен получать от станции оператора сигналы на останов и запуск электродвигателя и насосов.
- Он требует сигналов ответа от электродвигателя и насосов, означающих, что электродвигатель работает.
- Он должен вычислять время между передачей сигнала на запуск электродвигателя и приемом ответного сигнала. Если за это время ответный сигнал не будет получен, то электродвигатель должен выключаться.
- Он должен включать и выключать лампы на станции оператора.
- Он выдает сигнал, запускающий электродвигатель.

Эти требования можно определить в качестве входов и выходов FB. Следующая таблица показывает параметры FB электродвигателя в нашем типовом процессе.

Имя параметра	Input [Входной]	Output [Выходной]	In/out [Проходной]
Start [Пуск]	✓		
Stop [Останов]	✓		
Response [Ответ]	✓		
Reset_maint [Сброс обслуживания]	✓		
Timer_No [№ таймера]	✓		
Response_Time [Время ответа]	✓		
Fault [Неисправность]		✓	
Start_Dsp [Отображение пуска]		✓	
Stop_Dsp [Отображение останова]		✓	
Maint [Обслуживание]		✓	
Motor [Двигатель]			✓

Описание переменных FB для электродвигателя

Вы должны описать входные, выходные и проходные (in/out) FB для электродвигателя.

Адрес	Описание	Имя	Тип	Начальное значение
0.0	IN	Start	BOOL	FALSE
0.1	IN	Stop	BOOL	FALSE
0.2	IN	Response	BOOL	FALSE
0.3	IN	Reset_Maint	BOOL	FALSE
2.0	IN	Timer_No	TIMER	
4.0	IN	Response_Time	S5TIME	S5T#0MS
6.0	OUT	Fault	BOOL	FALSE
6.1	OUT	Start_Dsp	BOOL	FALSE
6.2	OUT	Stop_Dsp	BOOL	FALSE
6.3	OUT	Maint	BOOL	FALSE
8.0	IN_OUT	Motor	BOOL	FALSE
10.0	STAT	Time_bin	WORD	W#16#0
12.0	STAT	Time_BCD	WORD	W#16#0
14.0	STAT	Starts	INT	0
16.0	STAT	Start_Edge	BOOL	FALSE

В FB входные, выходные, проходные (in/out) и статические переменные сохраняются в экземплярном DB, указанном в команде вызова. Временные переменные сохраняются в L-стеке.

Программирование FB для электродвигателя

В STEP 7 каждый блок, вызываемый другим блоком, должен создаваться раньше блока, содержащего его вызов. Поэтому в типовой программе вы должны создать FB для электродвигателя раньше OB1.

Раздел кода FB1 представляется на языке программирования AWL следующим образом:

```
Network [Сегмент] 1       Запуск/останов и самоудержание
   A(
   O     #Start
   O     #Motor
   )
   AN    #Stop
   =     #Motor

Network 2 Контроль запуска
   A     #Motor
   L     #Response_Time
   SD    #Timer_No
   AN    #Motor
   R     #Timer_No
   L     #Timer_No
   T     #Timer_bin
   LC    #Timer_No
   T     #Timer_BCD
   A     #Timer_No
   AN    #Response
   S     #Fault
   R     #Motor

Network 3 Лампа запуска и сброс отказа
   A     #Response
   =     #Start_Dsp
   R     #Fault

Network 4 Лампа останова
   AN    #Response
   =     #Stop_Dsp

Network 5 Подсчет запусков
   A     #Motor
   FP    #Start_Edge
   JCN   lab1
   L     #Starts
   +     1
   T     #Starts
   lab1: NOP   0
```

Network 6 Лампа технического обслуживания

```
L     #Starts
L     50
>=I
=     #Maint
```

Network 7 Сброс счетчика запусков

```
A     #Reset_Maint
A     #Maint
JCN   END
L     0
T     #Starts
END: NOP   0
```

Создание экземплярных блоков данных

Создайте три блока данных и откройте их один за другим. В диалоговом окне "New Data Block [Новый блок данных]" выберите опцию "Data block referencing a function block [Блок данных, ссылающийся на функциональный блок]". В списке "Reference [Ссылка]" выберите "FB1". Тогда блоки данных определяются как экземплярные блоки данных с фиксированным назначением блоку FB1.

A.5.2.5 Создание FC для вентилей

Что требуется от FC?

Функция для впускных и питательных вентилей и выпускного вентиля содержит следующие логические функции:

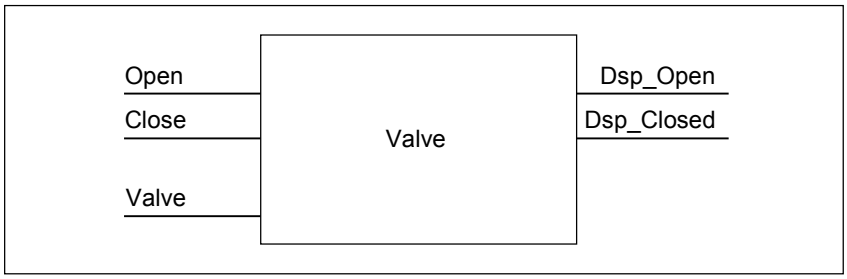
- Имеются вход для открытия и вход для закрытия вентилей.
- Блокировки разрешают вентилям открываться. Состояние блокировок хранится во временных локальных данных (L-стек) OB1 (”Valve_enable") и логически объединяется с входами для открытия и закрытия, когда FC для вентилей обрабатывается.

Следующая таблица показывает параметры, передаваемые в FC.

Параметры вентилей	Вход	Выход	Вход /Выход
Open [Открыть]	✓		
Close [Закрыть]	✓		
Dsp_Open [Отображение открытия]		✓	
Dsp_Closed [Отображения закрытия]		✓	
Valve [Вентиль]			✓

Спецификация входов и выходов

Следующий рисунок показывает входы и выходы общей FC для вентилей. Устройства, вызывающие FB электродвигателя, передают входные параметры. FC вентилей возвращает выходные параметры.



Описание переменных FC для вентилей

Так же, как и у FB для электродвигателя, вы должны для FC вентилей описать входные, выходные и проходные (in/out) параметры (см. следующую таблицу описания переменных).

Адрес	Описание	Имя	Тип	Начальное значение
0.0	IN	Open	BOOL	FALSE
0.1	IN	Close	BOOL	FALSE
2.0	OUT	Dsp_Open	BOOL	FALSE
2.1	OUT	Dsp_Closed	BOOL	FALSE
4.0	IN_OUT	Valve	BOOL	FALSE

В FC временные переменные хранятся в L-стеке. Входные, выходные и проходные (in/out) переменные хранятся в виде указателей на логический блок, который вызвал FC. Для этих переменных используется дополнительное пространство памяти в L-стеке (после временных переменных) .

Программирование FC для вентилей

Функция FC1 для вентилей должна создаваться раньше OB1, так как вызываемые блоки должны создаваться раньше вызывающих блоков.

Раздел кода FC1 представляется на языке программирования AWL так, как показано ниже:

Network 1 Открытие/закрытие и самоудержание

```
A(
O     #Open
O     #Valve
)
AN    #Close
=     #Valve
```

Network 2 Индикация «вентиль открыт»

```
A     #Valve
=     #Dsp_Open
```

Network 3 Индикация «вентиль закрыт»

```
AN    #Valve
=     #Dsp_Closed
```

A.5.2.6 Создание OB1

OB1 определяет структуру типовой программы. OB1 содержит также параметры, передаваемые различным функциям, например:

- Сегменты AWL для питающих насосов и электродвигателя мешалки снабжают FB электродвигателя входными параметрами для запуска ("Start"), останова ("Stop"), отклика ("Response") и сброса отображения технического обслуживания ("Reset_Maint"). FB электродвигателя обрабатывается в каждом цикле ПЛК.
- Если FB электродвигателя обрабатывается, входы Timer_No и Response_Time сообщают функции об используемом таймере и о том, какое время должно измеряться.
- FC вентилей и FB электродвигателя обрабатываются в каждом цикле программы программируемого контроллера, потому что они вызываются в OB1.

Программа использует FB электродвигателя с разными экземплярными DB, чтобы обрабатывать задачи управления питающими насосами и электродвигателем мешалки.

Описание переменных для OB1

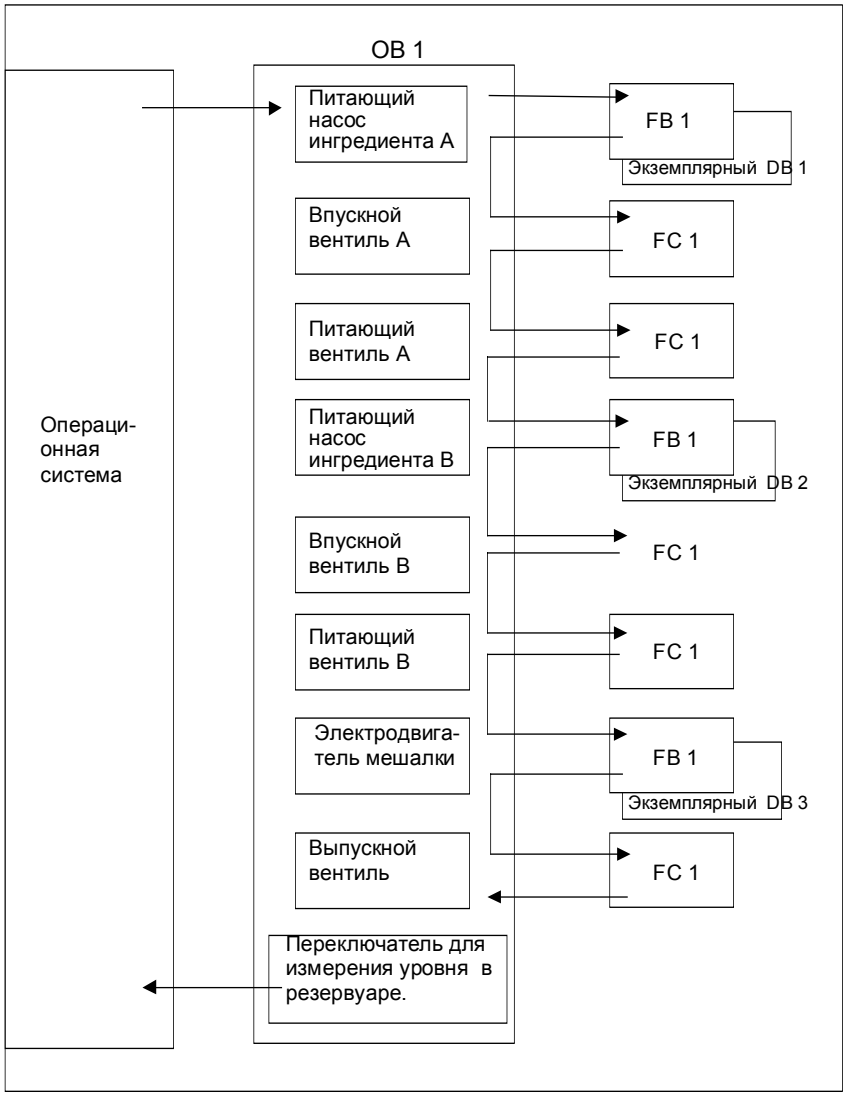
Таблица описания переменных для OB1 показана ниже. Первые 20 байтов содержат стартовую информацию OB1 и не должны изменяться.

Адрес	Описание	Имя	Тип
0.0	TEMP	OB1_EV_CLASS	BYTE
1.0	TEMP	OB1_SCAN1	BYTE
2.0	TEMP	OB1_PRIORITY	BYTE
3.0	TEMP	OB1_OB_NUMBR	BYTE
4.0	TEMP	OB1_RESERVED_1	BYTE
5.0	TEMP	OB1_RESERVED_2	BYTE
6.0	TEMP	OB1_PREV_CYCLE	INT
8.0	TEMP	OB1_MIN_CYCLE	INT
10.0	TEMP	OB1_MAX_CYCLE	INT
12.0	TEMP	OB1_DATE_TIME	DATE_AND_TIME
20.0	TEMP	Enable_motor	BOOL
20.1	TEMP	Enable_valve	BOOL
20.2	TEMP	Start_fulfilled	BOOL
20.3	TEMP	Stop_fulfilled	BOOL
20.4	TEMP	Inlet_valve_A_open	BOOL
20.5	TEMP	Inlet_valve_A_closed	BOOL
20.6	TEMP	Feed_valve_A_open	BOOL
20.7	TEMP	Feed_valve_A_closed	BOOL
21.0	TEMP	Inlet_valve_B_open	BOOL
21.1	TEMP	Inlet_valve_B_closed	BOOL
21.2	TEMP	Feed_valve_B_open	BOOL
21.3	TEMP	Feed_valve_B_closed	BOOL
21.4	TEMP	Open_drain	BOOL
21.5	TEMP	Close_drain	BOOL
21.6	TEMP	Valve_closed_fulfilled	BOOL

Создание программы для OB1

В STEP 7 каждый блок, вызываемый другим блоком, должен создаваться прежде блока, содержащего его вызов. Поэтому в типовой программе вы должны создать и FB электродвигателя, и FC вентилей прежде, чем программу для OB1.

Блоки FB1 и FC1 вызываются в OB1 более одного раза; FB1 вызывается с разными экземплярными DB:



Раздел кода OB1 представляется на языке программирования AWL так, как показано ниже:

Network 1 Блокировки для питающего насоса A

```
A     "EMER_STOP_off"
A     "Tank_below_max"
AN    "Drain"
=     #Enable_Motor
```

Network 2 Вызов FB электродвигателя для ингредиента A

```
A     "Feed_pump_A_start"
A     #Enable_Motor
=     #Start_Fulfilled
A(
O     "Feed_pump_A_stop"
ON    #Enable_Motor
)
=     #Stop_Fulfilled
CALL "Motor_block", "DB_feed_pump_A"
   Start          :=#Start_Fulfilled
   Stop           :=#Stop_Fullfilled
   Response       :="Flow_A"
   Reset_Maint    :="Reset_maint"
   Timer_No       :=T12
   Reponse_Time:=S5T#7S
   Fault          :="Feed_pump_A_fault"
   Start_Dsp      :="Feed_pump_A_on"
   Stop_Dsp       :="Feed_pump_A_off"
   Maint          :="Feed_pump_A_maint"
   Motor          :="Feed_pump_A"
```

Network 3 Задержка разблокировки вентиля ингредиента A

```
A     "Feed_pump_A"
L     S5T#1S
SD    T     13
AN    "Feed_pump_A"
R     T     13
A     T     13
=     #Enable_Valve
```

Network 4 Управление впускным вентилем для ингредиента A

```
AN"Flow_A"
AN"Feed_pump_A"
=     #Close_Valve_Fulfilled
CALL "Valve_block"
   Open        :=#Enable_Valve
   Close       :=#Close_Valve_Fulfilled
   Dsp_Open    :=#Inlet_Valve_A_Open
   Dsp_Closed:=#Inlet_Valve_A_Closed
   Valve       :="Inlet_Valve_A"
```

Network 5 Управление питающим вентилем для ингредиента A

```
AN"Flow_A"
AN"Feed_pump_A"
```

```
=       #Close_Valve_Fulfilled
CALL "Valve_block"
    Open        :=#Enable_Valve
    Close       :=#Close_Valve_Fulfilled
    Dsp_Open :=#Feed_Valve_A_Open
    Dsp_Closed:=#Feed_Valve_A_Closed
    Valve       :="Feed_Valve_A"
```

Network 6 Блокировки для питающего насоса B

```
A       "EMER_STOP_off"
A       "Tank_below_max"
AN      "Drain"
=       "Enable_Motor
```

Network 7 Вызов FB электродвигателя для ингредиента B

```
A       "Feed_pump_B_start"
A       #Enable_Motor
=       #Start_Fulfilled
A(
O       "Feed_pump_B_stop"
ON      #Enable_Motor
)
=       #Stop_Fulfilled
CALL "Motor_block", "DB_feed_pump_B"
    Start           :=#Start_Fulfilled
    Stop            :=#Stop_Fullfilled
    Response        :="Flow_B"
    Reset_Maint :="Reset_maint"
    Timer_No        :=T14
    Reponse_Time:=S5T#7S
    Fault           :="Feed_pump_B_fault"
    Start_Dsp       :="Feed_pump_B_on"
    Stop_Dsp        :="Feed_pump_B_off"
    Maint           :="Feed_pump_B_maint"
    Motor           :="Feed_pump_B"
```

Network 8 Задержка разблокировки вентиля ингредиента B

```
A       "Feed_pump_B"
L       S5T#1S
SD      T     15
AN      "Feed_pump_B"
R       T     15
A       T     15
=       #Enable_Valve
```

Network 9 Управление впускным вентилем для ингредиента B

```
AN"Flow_B"
AN"Feed_pump_B"
=       #Close_Valve_Fulfilled
CALL "Valve_block"
    Open        :=#Enable_Valve
    Close       :=#Close_Valve_Fulfilled
    Dsp_Open :=#Inlet_Valve_B_Open
    Dsp_Closed:=#Inlet_Valve_B_Closed
    Valve       :="Inlet_Valve_B"
```

Network 10 Управление питающим вентилем для ингредиента B

```
AN"Flow_B"
AN"Feed_pump_B"
=     #Close_Valve_Fulfilled
CALL "Valve_block"
   Open       :=#Enable_Valve
   Close      :=#Close_Valve_Fulfilled
   Dsp_Open :=#Feed_Valve_B_Open
   Dsp_Closed:=#Feed_Valve_B_Closed
   Valve      :="Feed_Valve_B"
```

Network 11 Блокировки для мешалки

```
A     "EMER_STOP_off"
A     "Tank_above_min"
AN    "Drain"
=     #Enable_Motor
```

Network 12 Вызов FB электродвигателя для мешалки

```
A     "Agitator_start"
A     #Enable_Motor
=     #Start_Fulfilled
A(
O     "Agitator_stop"
ON    #Enable_Motor
)
=     #Stop_Fulfilled
CALL "Motor_block", "DB_Agitator"
   Start          :=#Start_Fulfilled
   Stop           :=#Stop_Fullfilled
   Response       :="Agitator_running"
   Reset_Maint :="Reset_maint"
   Timer_No       :=T16
   Reponse_Time:=S5T#10S
   Fault          :="Agitator_fault"
   Start_Dsp      :="Agitator_on"
   Stop_Dsp       :="Agitator_off"
   Maint          :="Agitator_maint"
   Motor          :="Agitator"
```

Network 13 Блокировки для выпускного вентиля

```
A     "EMER_STOP_off"
A     "Tank_not_empty"
AN    "Agitator"
=     "Enable_Valve
```

Network 14 Управление сливным вентилем

```
A     "Drain_open"
A     #Enable_Valve
=     #Open_Drain
A(
O     "Drain_closed"
ON    #Enable_Valve
)
=     #Close_Drain
CALL "Valve_block"
   Open       :=#Open_Drain
   Close      :=#Close_Drain
   Dsp_Open   :="Drain_open_disp"
```

```
      Dsp_Closed  :="Drain_closed_disp"
      Valve       :="Drain"
```

Network 15 Отображение уровня резервуара

```
AN    "Tank_below_max"
=     "Tank_max_disp"
AN    "Tank_above_min"
=     "Tank_min_disp"
AN    "Tank_not_empty"
=     "Tank_empty_disp"
```

A.5.3 Пример обработки прерываний по времени

A.5.3.1 Структура программы пользователя "Прерывания по времени"

Задача

Выход Q 4.0 должен быть установлен в период с 5.00 утра в понедельник до 8.00 пополудни в пятницу. В период с 8.00 пополудни в пятницу до 5.00 утра в понедельник выход Q 4.0 должен быть сброшен.

Преобразование в программу пользователя

Следующая таблица показывает подзадачи используемых блоков.

Блок	Подзадача
OB1	Вызывает функцию FC12
FC12	В зависимости от состояния выхода Q 4.0, состояния прерывания по времени и входов I 0.0 и I 0.1 • Определить время запуска • Установить прерывание по времени • Активизировать прерывание по времени • CAN_TINT
OB10	В зависимости от текущего дня недели • Определить время запуска • Установить или сбросить выход Q 4.0 • Установить следующее прерывание по времени • Активизировать следующее прерывание по времени
OB80	Установить выход Q 4.1 Сохранить информацию о событии запуска OB80 в области памяти с побитовым доступом

Используемые адреса

Следующая таблица показывает используемые общедоступные адреса. Временные локальные переменные описываются в разделе описаний соответствующего блока.

Адрес	Значение
I0.0	Вход для разблокировки действий "установка прерывания по времени" и "запуск прерывание по времени"
I0.1	Вход для отмены прерывания по времени
Q4.0	Выход, устанавливаемый/сбрасываемый прерыванием по времени OB (OB10)
Q4.1	Выход, устанавливаемый ошибкой времени (OB80)
MW16	STATUS [состояние] прерывания по времени (SFC31 "QRY_TINT")
с MB100 по MB107	Память для информации о событии запуска OB10 (только время суток)
с MB110 по MB129	Память для информации о событии запуска OB80 (ошибка времени)
MW200	RET_VAL в SFC28 "SET_TINT"
MB202	Буфер двоичного результата (бит состояния BR) для SFC
MW204	RET_VAL в SFC30 "ACT_TINT"
MW208	RET_VAL в SFC31 "QRY_TINT"

Системные функции и используемые функции

В примере программирования используются следующие системные функции:

- SFC28 "SET_TINT" : Установка прерывания по времени
- SFC29 "CAN_TINT" : Отмена прерывания по времени
- SFC30 "ACT_TINT" : Запуск прерывания по времени
- SFC31 "QRY_TINT" : Запрос прерывания по времени
- FC3 "D_TOD_DT" : Объединение DATE и TIME_OF_DAY в DT

A.5.3.2 FC12

Раздел описаний

В разделе описаний FC12 описываются следующие временные локальные переменные:

Имя переменной	Тип данных	Описание	Комментарий
IN_TIME	TIME_OF_DAY	TEMP	Время запуска
IN_DATE	DATE	TEMP	Дата запуска
OUT_TIME_DATE	DATE_AND_TIME	TEMP	Дата/время запуска преобразованные
OK_MEMORY	BOOL	TEMP	Разблокировка установки прерывания по времени

Раздел кода AWL

Введите в раздел кода FC12 следующую программу пользователя на AWL:

```
AWL (FC12)                                  Объяснение
Network 1
        CALL SFC 31                         SFC QRY_TINT
         OB_NO  := 10                       Запрос STATUS [состояния] прерываний по
         RET_VAL:= MW 208                   времени
         STATUS := MW 16

Network 2:
        AN      Q       4.0
        JC      mond                        Задать время запуска в зависимости от
        L       D#1995–1–27                 Q 4.0 (в переменной
        T       #IN_DATE                    #IN_DATE и #IN_TIME)
        L       TOD#20:0:0.0                Дата запуска – пятница
        T       #IN_TIME
        JU      cnvrt
mond:   L       D#1995–1–23
        T       #IN_DATE
        L       TOD#5:0:0.0                 Дата запуска – понедельник
        T       #IN_TIME
cnvrt:  NOP     0
```

AWL (FC12)			Объяснение
Network 3:			
	CALL FC 3		**Преобразовать форматы DATE и TIME_OF_DAY**
	IN1	**:= #IN_DATE**	**в формат DATE_AND_TIME (для установки**
	IN2	**:= #IN_TIME**	**прерывания по времени)**
	RET_VAL	**:= #OUT_TIME_DATE**	
Network 4:			
	A	**I 0.0**	**Все требования для установки прерывания по**
	AN	**M 17.2**	**времени удовлетворены? (вход**
	A	**M 17.4**	**разблокировки установлен, и прерывание по**
	=	**#OK_MEMORY**	**времени не активно и OB прерывания по**
Network 5:			**времени загружен)**
	A	**#OK_MEMORY**	**Если да, то установить прерывание по**
	JNB	**m001**	**времени ...**
	CALL SFC 28		
	OB_NO	**:= 10**	
	SDT	**:= #OUT_TIME_DATE**	
	PERIOD	**:= W#16#1201**	
	RET_VAL	**:= MW 200**	
m001	**A**	**BR**	
	=	**M 202.3**	
Network 6:			
	A	**#OK_MEMORY**	**...и запустить прерывание по времени.**
	JNB	**m002**	
	CALL SFC 30		
	OB_NO	**:= 10**	
	RET_VAL	**:= MW 204**	
m002	**A**	**BR**	
	=	**M 202.4**	
Network 7:			
	A	**I 0.1**	**Если установлен вход отмены прерываний по**
	JNB	**m003**	**времени, то отменить прерывание по времени.**
	CALL SFC 29		
	OB_NO	**:= 10**	
	RET_VAL	**:= MW 210**	
m003	**A**	**BR**	
	=	**M 202.5**	

A.5.3.3 OB10

Раздел описаний

В отличие от раздела описаний OB10, заданного по умолчанию, описываются следующие временные локальные переменные:

- Структура для всей информации о событии запуска (STARTINFO)
- Структура для времени (T_STMP) внутри структуры STARTINFO
- Другие временные локальные переменные WDAY, IN_DATE, IN_TIME и OUT_TIME_DATE

Имя переменной	Тип данных	Описание	Комментарий
STARTINFO	STRUCT	TEMP	Вся информация о событии запуска OB10, описанная как структура
E_ID	WORD	TEMP	Идентификатор события:
PR_CLASS	BYTE	TEMP	Класс приоритета
OB_NO	BYTE	TEMP	Номер OB
RESERVED_1	BYTE	TEMP	Зарезервировано
RESERVED_2	BYTE	TEMP	Зарезервировано
PERIOD	WORD	TEMP	Периодичность прерывания по времени
RESERVED_3	DWORD	TEMP	Зарезервировано
T_STMP	STRUCT	TEMP	Структура для элементов времени суток
YEAR	BYTE	TEMP	
MONTH	BYTE	TEMP	
DAY	BYTE	TEMP	
HOUR	BYTE	TEMP	
MINUTES	BYTE	TEMP	
SECONDS	BYTE	TEMP	
MSEC_WDAY	WORD	TEMP	
	END_STRUCT	TEMP	
	END_STRUCT	TEMP	
WDAY	INT	TEMP	День недели
IN_DATE	DATE	TEMP	Входная переменная FC3 (преобразования формата времени)
IN_TIME	TIME_OF_DAY	TEMP	Входная переменная FC3 (преобразования формата времени)
OUT_TIME_DATE	DATE_AND_TIME	TEMP	Выходная переменная FC3 и входная переменная SFC28

Раздел кода AWL

Введите в раздел кода OB10 следующую программу пользователя на AWL:

```
AWL (OB10)                                  Объяснение
Network 1
          L     #STARTINFO.T_STMP.MSEC_WDAY  Выбрать день недели
          L     W#16#F
          AW
          T     #WDAY                        и сохранить.
Network 2:
          L     #WDAY                        Если день недели не понедельник, то задать
          L     2                            понедельник 5.00 утра как следующий момент
          <>I                                запуска и сбросить выход Q 4.0.
          JC    mond
Network 3:                                   В противном случае, если день недели
          L     D#1995–1–27                  понедельник, то задать пятницу 8.00 пополудни
          T     #IN_DATE                     (20.00) как следующий момент запуска и
          L     TOD#20:0:0.0                 установить Q 4.0.
          T     #IN_TIME
          SET
          =     Q 4.0
          JU    cnvrt

mond:     L     D#1995–1–23
          T     #IN_DATE
          L     TOD#5:0:0.0
          T     #IN_TIME
          CLR                                Время запуска задано.
          =     Q 4.0                        Преобразовать заданное время запуска в
                                             формат DATE_AND_TIME (для SFC28).
cnvrt:    NOP   0
Network 4:
          CALL  FC 3                         Установить прерывание по времени.
           IN1     := #IN_DATE
           IN2     := #IN_TIME
           RET_VAL := #OUT_TIME_DATE
Network 5:
          CALL SFC 28
           OB_NO  := 10
           SDT    := #OUT_TIME_DATE
           PERIOD  := W#16#1201
           RET_VAL := MW 200
          A     BR
          =     M 202.1
Network 6:
          CALL SFC 30                        Запустить прерывание по времени.
           OB_NO  := 10
           RET_VAL := MW 204
          A     BR
          =     M 202.2
Network 7:
          CALL SFC 20                        Пересылка блоков: сохранить время суток из
           SRCBLK  := #STARTINFO.T_STMP      информации о событии запуска OB10 в
           RET_VAL := MW 206                 области памяти с MB100 по MB107.
           DSTBLK  := P#M 100.0 BYTE  8
```

A.5.3.4 OB1 и OB80

Поскольку в этом примере информация о событии запуска OB1 (OB для циклической программы) не оценивается, то отображается только информация о событии запуска OB80.

Раздел кода OB1

Введите в раздел кода OB1 следующую программу пользователя на AWL:

AWL (OB1)		**Объяснение**
CALL	**FC 12**	**Вызвать функцию FC12**

Раздел описаний OB80

В отличие от заданного по умолчанию раздела описаний, в OB80 описываются следующие временные локальные переменные:

- Структура для всей информации о событии запуска (STARTINFO)
- Структура для времени (T_STMP) внутри структуры STARTINFO

Имя переменной	Тип данных	Описание	Комментарий
STARTINFO	STRUCT	TEMP	Вся информация о событии запуска OB80, объявленная как структура
E_ID	WORD	TEMP	Идентификатор события:
PR_CLASS	BYTE	TEMP	Класс приоритета
OB_NO	BYTE	TEMP	Номер OB
RESERVED_1	BYTE	TEMP	Зарезервировано
RESERVED_2	BYTE	TEMP	Зарезервировано
A1_INFO	WORD	TEMP	Дополнительная информация о событии, вызвавшем ошибку
A2_INFO	DWORD	TEMP	Дополнительная информация об идентификаторе события, классе приоритета и номере OB ошибки
T_STMP	STRUCT	TEMP	Структура для элементов времени суток
YEAR	BYTE	TEMP	
MONTH	BYTE	TEMP	
DAY	BYTE	TEMP	
HOUR	BYTE	TEMP	
MINUTES	BYTE	TEMP	
SECONDS	BYTE	TEMP	
MSEC_WDAY	WORD	TEMP	
	END_STRUCT	TEMP	
	END_STRUCT	TEMP	

Раздел кода OB80

Введите в раздел кода OB80, вызываемого операционной системой при появлении ошибки времени, следующую программу пользователя на AWL:

AWL (OB80)	Объяснение
Network 1	
AN Q 4.1 **S Q 4.1**	**Установить выход Q 4.1, если произошла ошибка времени.**
CALL SFC 20 **SRCBLK := #STARTINFO** **RET_VAL := MW 210** **DSTBLK := P#M 110.0 Byte 20**	**Передача блоков: сохранить всю информацию о событии запуска в области памяти с MB110 по MB129.**

A.5.4 Пример обработки прерываний с задержкой

A.5.4.1 Структура программы пользователя "Прерывания с задержкой"

Задача

Когда вход I 0.0 устанавливается, выход Q 4.0 должен устанавливаться 10 секундами позже. Каждый раз, когда вход I 0.0 устанавливается, задержка должна перезапускаться.

Время (секунды и миллисекунды) запуска прерывания с задержкой должно появляться в виде специфического для пользователя идентификатора в информации о событии запуска OB прерываний с задержкой (OB20).

Если в течение этих 10 секунд устанавливается I 0.1, то организационный блок OB20 не должен вызываться; значение выхода Q 4.0 не должно устанавливаться.

Когда устанавливается вход I 0.2, выход Q 4.0 должен сбрасываться.

Преобразование в программу пользователя

Следующая таблица показывает подзадачи используемых блоков.

Блок	Подзадача
OB1	Чтение текущего времени и подготовка к запуску прерывания с задержкой Запуск прерывания с задержкой в зависимости от фронта сигнала на входе I 0.0 Отмена прерывания с задержкой в зависимости от состояния прерывания с задержкой и фронта сигнала на входе I 0.1 Сброс выхода Q 4.0в зависимости от состояния входа I 0.2
OB20	Установка выхода Q 4.0 Чтение и подготовка текущего времени Сохранение информации о событии запуска в памяти с побитовым доступом

Используемые адреса

Следующая таблица показывает используемые общедоступные адреса. Временные локальные переменные описываются в разделе описаний соответствующего блока.

Адрес	Значение
I0.0	Вход для разблокировки действия «запуск прерывания с задержкой»
I0.1	Вход для отмены прерывания с задержкой
I0.2	Вход для сброса выхода Q 4.0
Q4.0	Выход, устанавливаемый OB (OB20) прерываний с задержкой
MB1	Используется как флаг фронта и буфер двоичного результата (бит состояния BR) для SFC
MW4	STATUS [состояние] прерывания с задержкой (SFC34 "QRY_TINT")
MD10	Секунды и миллисекунды в двоично-десятичном коде из информации о событии запуска OB1
MW 100	RET_VAL в SFC32 "SRT_DINT"
MW102	RET_VAL в SFC34 "QRY_DINT"
MW104	RET_VAL в SFC33 "CAN_DINT"
MW106	RET_VAL в SFC20 "BLKMOV"
с MB120 по MB139	Память для информации о событии запуска OB20
MD140	Секунды и миллисекунды в двоично-десятичном коде из информации о событии запуска OB20
MW144	Секунды и миллисекунды в двоично-десятичном коде из информации о событии запуска OB1; извлекаются из информации о событии запуска OB20 (специфический для пользователя ID SIGN)

Используемые системные функции

В программе пользователя «Прерывания с задержкой» используются следующие SFC:

- SFC32 "SRT_DINT" : Запуск прерывания с задержкой
- SFC33 "CAN_DINT" : Отмена прерывания с задержкой
- SFC34 "QRY_DINT" : Запрос прерывания с задержкой

A.5.4.2 OB20

Раздел описаний

В отличие от раздела описаний OB20, заданного по умолчанию, описываются следующие временные локальные переменные:

- Структура для всей информации о событии запуска (STARTINFO)
- Структура для времени (T_STMP) внутри структуры STARTINFO

Имя переменной	Тип данных	Описание	Комментарий
STARTINFO	STRUCT	TEMP	Информация о запуске для OB20
E_ID	WORD	TEMP	Идентификатор события:
PC_NO	BYTE	TEMP	Класс приоритета
OB_NO	BYTE	TEMP	Номер OB
D_ID 1	BYTE	TEMP	Данные ID 1
D_ID 2	BYTE	TEMP	Данные ID 2
SIGN	WORD	TEMP	Специфический для пользователя ID
DTIME	TIME	TEMP	Время, с которым запускается прерывание с задержкой
T_STMP	STRUCT	TEMP	Структура для элементов времени суток (временной ярлык)
YEAR	BYTE	TEMP	
MONTH	BYTE	TEMP	
DAY	BYTE	TEMP	
HOUR	BYTE	TEMP	
MINUTES	BYTE	TEMP	
SECONDS	BYTE	TEMP	
MSEC_WDAY	WORD	TEMP	
	END_STRUCT	TEMP	
	END_STRUCT	TEMP	

Раздел кода

Введите в раздел кода OB20 следующую программу пользователя на AWL:

AWL (OB20)		Объяснение
Network 1		
SET		**Безусловно установить выход Q 4.0**
=	**Q 4.0**	
Network 2:		
L	**QW 4**	**Немедленно активизировать выходное слово**
T	**PQW 4**	
Network 3:		
L	**#STARTINFO.T_STMP.SECONDS**	**Считать секунды из информации о событии запуска**
T	**MW 140**	
L	**#STARTINFO.T_STMP.MSEC_WDAY**	**Считать миллисекунды и день недели из информации о событии запуска**
T	**MW 142**	
L	**MD 140**	
SRD	**4**	**Удалить день недели и записать обратно миллисекунды (теперь в BCD-коде в MW 142)**
T	**MD 140**	
Network 4:		**Считать время запуска прерывания с задержкой (= вызова SFC32) из информации о событии запуска**
L	**#STARTINFO.SIGN**	
T	**MW 144**	
Network 5:		
CALL SFC 20		**Копировать информацию о событии запуска в область памяти (с MB120 по MB139)**
SRCBLK	**:= STARTINFO**	
RET_VAL	**:= MW 106**	
DSTBLK	**:= P#M 120.0 Byte 20**	

A.5.4.3 OB1

Раздел описаний

В отличие от раздела описаний OB1, заданного по умолчанию, описываются следующие временные локальные переменные:

- Структура для всей информации о событии запуска (STARTINFO)
- Структура для времени (T_STMP) внутри структуры STARTINFO

Имя переменной	Тип данных	Описание	Комментарий
STARTINFO	STRUCT	TEMP	Информация о запуске для OB1
E_ID	WORD	TEMP	ID события:
PC_NO	BYTE	TEMP	Класс приоритета
OB_NO	BYTE	TEMP	Номер OB
D_ID 1	BYTE	TEMP	Данные ID 1
D_ID 2	BYTE	TEMP	Данные ID 2
CUR_CYC	INT	TEMP	Текущее время цикла
MIN_CYC	INT	TEMP	Минимальное время цикла
MAX_CYC	INT	TEMP	Максимальное время цикла
T_STMP	STRUCT	TEMP	Структура для подробностей суточного времени (временной ярлык)
YEAR	BYTE	TEMP	
MONTH	BYTE	TEMP	
DAY	BYTE	TEMP	
HOUR	BYTE	TEMP	
MINUTES	BYTE	TEMP	
SECONDS	BYTE	TEMP	
MSEC_WDAY	WORD	TEMP	
	END_STRUCT	TEMP	
	END_STRUCT	TEMP	

Раздел кода

Введите в раздел кода OB1 следующую программу пользователя на AWL:

```
AWL (OB1)                                          Объяснение
Network 1
          L     #STARTINFO.T_STMP.SECONDS          Считать секунды из информации о событии
          T     MW 10                              запуска
          L     #STARTINFO.T_STMP.MSEC_WDAY        Считать миллисекунды и день недели из
          T     MW 12                              информации о событии запуска
          L     MD 10                              Удалить день недели и записать обратно
          SRD   4                                  миллисекунды (теперь в BCD-коде в MW 12)
          T     MD 10
Network 2:                                         Положительный фронт на входе I 0.0?
          A     I 0.0
          FP    M 1.0
          =     M 1.1
Network 3:                                         Если да, то запустить прерывание с задержкой
          A     M 1.1                              (время запуска прерывания с задержкой,
          JNB   m001                               присвоенное параметру SIGN)
          CALL SFC 32
           OB_NO  := 20
           DTME   := T#10S
           SIGN   := MW 12
           RET_VAL:= MW 100
m001:     NOP   0

Network 4:                                         Запрос состояния прерывания с задержкой
          CALL SFC 34                              (SFC QRY_DINT)
           OB_NO  := 20
           RET_VAL:= MW 102
           STATUS := MW 4
Network 5:                                         Положительный фронт на входе I 0.1?
          A     I 0.1
          FP    M 1.3
          =     M 1.4

Network 6:                                         ...и прерывание с задержкой запущено (бит 2
          A     M 1.4                              STATUS [состояния] прерывания с задержкой)?
          A     M 5.2                              Тогда отменить прерывание с задержкой
          JNB   m002
          CALL SFC 33
           OB_NO  := 20
           RET_VAL:= MW 104                        Сбросить выход Q 4.0 через вход I 0.2
m002:     NOP   0
          A     I 0.2
          R     Q 4.0
```

A.5.4.4 Пример маскирования и демаскирования синхронных ошибок

Следующий пример программы пользователя показывает, как следует маскировать и демаскировать синхронные ошибки. При использовании SFC36 "MSK_FLT" в программируемом фильтре ошибок маскируются следующие ошибки:

- Ошибка длины области при чтении
- Ошибка длины области при записи

Вторым вызовом SFC36 "MSK_FLT" можно замаскировать также область доступа:

- Ошибка доступа для ввода/вывода при записи

Замаскированные синхронные ошибки запрашиваются с помощью SFC38 "READ_ERR". "Ошибка доступа для ввода/вывода при записи" демаскируется с помощью SFC37 "DMSK_FLT.

Раздел кода

Ниже вы найдете OB1, в котором запрограммирован пример программы пользователя в форме списка операторов.

AWL (Network 1)				Объяснение
	AN	M 255.0		Бит не сохраняемой памяти M 255.0 (только при первом прогоне = 0)
	JNB	m001		
	CALL	SFC 36		SFC36 MSK_FLT (маскирование синхронных ошибок) Бит 2 = Бит 3 = 1 (BLFL и BLFS маскируются)
	PRGFLT_SET_MASK		:=DW#16#C	Все биты=0 (ошибки доступа не маскируются)
	ACCFLT_SET_MASK		:=DW#16#0	Возвращаемое значение
	RET_VAL		:=MW 100	Выходной фильтр текущей программной ошибки в MD10
	PRGFLT_MASKED		:=MD 10	Выходной фильтр текущей ошибки доступа в MD14
	ACCFLT_MASKED		:=MD 14	
m001:	A	BR		Установить M255.0, если маскирование успешно.
	S	M 255.0		

AWL (Network 2)				Объяснение
	CALL	SFC 36		SFC36 MSK_FLT (маскирование синхронных ошибок)
	PRGFLT_SET_MASK		:=DW#16#0	Все биты=0 (дальнейшие программные ошибки не маскируются
	ACCFLT_SET_MASK		:=DW#16#8	Бит 3 = 1 (ошибки доступа для записи маскируются)
	RET_VAL		:=MW 102	Возвращаемое значение
	PRGFLT_MASKED		:=MD 20	Выходной фильтр текущей программной ошибки в MD20
	ACCFLT_MASKED		:=MD 24	Выходной фильтр текущей ошибки доступа в MD24

AWL (Network 3)			Объяснение
AN	M 27.3		Завершить блок, если ошибка доступа для записи (бит 3 в ACCFLT_MASKED) не замаскирована
BEC			

AWL (Network 4)			Объяснение
L	B#16#0		
T	PQB 16		Доступ для записи (при значении 0)в PQB 16

AWL (Network 5)			Объяснение
CALL	SFC 38		SFC38 READ_ERR (запрос синхронных ошибок)
PRGFLT_QUERY		:=DW#16#0	Все биты=0 (программные ошибки не запрашиваются)
ACCFLT_QUERY		:=DW#16#8	Бит 3 = 1 (ошибка доступа для записи запрашивается)
RET_VAL		:=MW 104	Возвращаемое значение
PRGFLT_CLR		:=MD 30	Выходной фильтр текущей программной ошибки в MD30
ACCFLT_CLR		:=MD 34	Выходной фильтр текущей ошибки доступа в MD34
A	BR		Ошибок не происходило и обнаружена ошибка доступа для записи
A	M 37.3		Инвертировать RLO
NOT			M 0.0=1, если присутствует PQB 16
=	M 0.0		

AWL (Network 6)			Объяснение
L	B#16#0		
T	PQB 17		Доступ для записи (при значении 0) в PQB 17

AWL (Network 7)			Объяснение
CALL	SFC 38		SFC38 READ_ERR (запрос синхронных ошибок)
PRGFLT_QUERY		:=DW#16#0	Все биты=0 (программные ошибки не запрашиваются)
ACCFLT_QUERY		:=DW#16#8	Бит 3 = 1 (ошибка доступа для записи запрашивается)
RET_VAL		:=MW 104	Возвращаемое значение
PRGFLT_CLR		:=MD 30	Выходной фильтр текущей программной ошибки в MD30
ACCFLT_CLR		:=MD 34	Выходной фильтр текущей ошибки доступа в MD34
A	BR		Ошибок не происходило и обнаружена ошибка доступа для записи
A	M 37.3		Инвертировать RLO
NOT			M 0.1=1, если присутствует PQB 17
=	M 0.1		

AWL (Network 8)			Объяснение
L	B#16#0		
T	PQB 18		Доступ для записи (при значении 0) в PQB 18

AWL (Network 9)			Объяснение
CALL	SFC 38		SFC38 READ_ERR (запрос синхронных ошибок)
PRGFLT_QUERY		:=DW#16#0	Все биты=0 (программные ошибки не запрашиваются)
ACCFLT_QUERY		:=DW#16#8	Бит 3 = 1 (ошибка доступа для записи запрашивается)
RET_VAL		:=MW 104	Возвращаемое значение
PRGFLT_CLR		:=MD 30	Выходной фильтр текущей программной ошибки в MD30
ACCFLT_CLR		:=MD 34	Выходной фильтр текущей ошибки доступа в MD34
A	BR		Ошибок не происходило и обнаружена ошибка доступа для записи
A	M 37.3		Инвертировать RLO
NOT			M 0.2=1, если присутствует PQB 18
=	M 0.2		

AWL (Network 10)		Объяснение
L	B#16#0	
T	PQB 19	Доступ для записи (при значении 0) в PQB 19

AWL (Network 11)			Объяснение
CALL	SFC 38		SFC38 READ_ERR (запрос синхронных ошибок)
PRGFLT_QUERY		:=DW#16#0	Все биты=0 (программные ошибки не запрашиваются)
ACCFLT_QUERY		:=DW#16#8	Бит 3 = 1 (ошибка доступа для записи запрашивается)
RET_VAL		:=MW 104	Возвращаемое значение
PRGFLT_CLR		:=MD 30	Выходной фильтр текущей программной ошибки в MD30
ACCFLT_CLR		:=MD 34	Выходной фильтр текущей ошибки доступа в MD34
A	BR		Ошибок не происходило и обнаружена ошибка доступа для записи
A	M 37.3		Инвертировать RLO
NOT			M 0.3=1, если присутствует PQB 19
=	M 0.3		

AWL (Network 12)			Объяснение
CALL	SFC 37		SFC37 DMSK_FLT (демаскирование синхронных ошибок)
PRGFLT_RESET_MASK		:=DW#16#0	Все биты=0 (дальнейшие программные ошибки не демаскируются)
ACCFLT_RESET_MASK		:=DW#16#8	Бит 3 = 1 (ошибка доступа для записи демаскируется)
RET_VAL		:=MW 102	Возвращаемое значение
PRGFLT_MASKED		:=MD 20	Выходной фильтр текущей программной ошибки в MD20
ACCFLT_MASKED		:=MD 24	Выходной фильтр текущей ошибки доступа в MD24

AWL (Network 13)		Объяснение
A	M 27.3	Завершить блок, если ошибка доступа для записи (бит 3 в ACCFLT_MASKED) не демаскирована
BEC		

AWL (Network 14)			Объяснение
	A	M 0.0	
	JNB	m002	
	L	IB 0	Преобразовать IB0 в PQB 16, если он есть
	T	PQB 16	
m002:	NOP	0	

AWL (Network 15)			Объяснение
	A	M 0.1	
	JNB	m003	
	L	IB 1	Преобразовать IB1 в PQB 17, если он есть
	T	PQB 17	
m003:	NOP	0	

AWL (Network 16)			Объяснение
	A	M 0.2	
	JNB	m004	
	L	IB 2	Преобразовать IB2 в PQB 18, если он есть
	T	PQB 18	
m004:	NOP	0	

AWL (Network 17)			Объяснение
	A	M 0.3	
	JNB	m005	
	L	IB 3	Преобразовать IB3 в PQB 19, если он есть
	T	PQB 19	
m005:	NOP	0	

A.5.4.5 Пример блокировки и разблокировки прерываний и асинхронных ошибок (SFC39 и SFC40)

В этом примере программы пользователя предполагается раздел программы, выполнение которого не может прерываться прерываниями. Для этого раздела программы вызовы OB35 (прерывание по времени) блокируются при помощи SFC 39 "DIS_IRT", а позже снова разблокируются при помощи SFC 40 "EN_IRT".

SFC39 и SFC40 вызываются в OB1:

AWL (OB1)		Объяснение
A	M 0.0	Раздел программы, который может
S	M 90.1	прерываться без проблем:
A	M 0.1	
S	M 90.0	
:		
:		Раздел программы, который не должен прерываться прерываниями:
CALL	SFC 39	Блокировать и сбрасывать прерывания
MODE	:=B#16#2	Режим 2: блокировать отдельные OB
OB_NO	:=35	прерываний
RET_VAL	:=MW 100	Блокировать OB35
:		
:		
L	PIW 100	
T	MW 200	
L	MW 90	
T	MW 92	
:		
:		
CALL	SFC 40	
MODE	:=B#16#2	Разблокировать прерывания
OB_NO	:=35	Режим 2: разблокировать отдельные OB
RET_VAL	:=MW 102	прерываний
		Разблокировать OB35
A	M 10.0	Раздел программы, который может
S	M 190.1	прерываться без проблем:
A	M 10.1	
S	M 190.0	
:		
:		

A.5.4.6 Пример задержанной обработки прерываний и асинхронных ошибок (SFC41 и SFC42)

В этом примере программы пользователя предполагается раздел программы, выполнение которого не может прерываться прерываниями. Для этого раздела программы прерывания задерживаются при помощи SFC41 "DIS_AIRT", а позже снова разблокируются при помощи SFC42 "EN_AIRT".

SFC41 и SFC42 вызываются в OB1:

AWL (OB1)			Объяснение
	A	M 0.0	Раздел программы, который может
	S	M 90.1	прерываться без проблем:
	A	M 0.1	
	S	M 90.0	
	:		
	:		
	CALL	SFC 41	Раздел программы, который не должен
	RET_VAL	:=MW 100	прерываться прерываниями:
	L	PIW 100	Блокировать и задерживать прерывания
	T	MW 200	
	L	MW 90	
	T	MW 92	
	:		
	:		
	:		
	CALL	SFC 42	Разблокировать прерывания
	RET_VAL	:=MW 102	Число установленных блокировок прерываний
	L	MW 100	находится в возвращаемом значении
			Число установленных блокировок прерываний
	DEC	1	находится в возвращаемом значении
	L	MW 102	Это число после разблокировки прерываний
			должно иметь такое же значение, как перед
	<>I		блокировкой прерываний (здесь "0")
	JC	err	
			Раздел программы, который может
			прерываться без проблем:
	A	M 10.0	
	S	M 190.1	
	A	M 10.1	
	S	M 190.0	
	:		
	:		Число установленных блокировок прерываний
	BEU		отображается
err:	L	MW 102	
	T	QW 12	

A.6 Доступ к области данных процесса и области периферийных данных

A.6.1 Доступ к области данных процесса

CPU может обращаться к входам и выходам центральных и децентрализованных модулей цифрового ввода/вывода либо косвенно, используя таблицы образа процесса, либо непосредственно через заднюю или P-шину.

CPU обращается к входам и выходам центральных и децентрализованных модулей аналогового ввода/вывода непосредственно через заднюю или P-шину.

Адресация модулей

Вы назначаете модулям адреса, используемые в вашей программе, конфигурируя модули с помощью STEP 7 следующим образом:

- Для центральных модулей ввода/вывода: компоновка стойки и назначение модулей слотам в конфигурационной таблице.
- Для станций с децентрализованной периферией (PROFIBUS-DP): компоновка slave-устройств DP в конфигурационной таблице ”мастер-системы" с адресом PROFIBUS и назначение модулей слотам.

При конфигурировании модулей больше не требуется устанавливать адреса на отдельных модулях с помощью переключателей. В качестве результата конфигурирования устройство программирования передает в CPU данные, позволяющие CPU распознавать назначенные ему модули.

Адресация периферийных входов/выходов

Для входов и выходов имеются отдельные области адресов. Это означает, что адрес периферийной области должен включать не только тип доступа – байт или слово, но также и идентификатор I для входов и идентификатор Q для выходов.

Следующая таблица показывает доступные области периферийных адресов.

Область адресов	Доступ через единицы следующего размера	Запись в S7 (IEC)
Периферийная область: входы	Периферийный входной байт Периферийное входное слово Периферийное входное двойное слово	PIB PIW PID
Периферийная область: выходы	Периферийный выходной байт Периферийное выходное слово Периферийное выходное двойное слово	PQB PQW PQD

Чтобы выяснить, какие области адресов возможны в отдельных модулях, обратитесь к следующим руководствам:

- Руководство «S7-300 Programmable Controller, Hardware and Installation [Программируемый контроллер S7-300: Аппаратные средства и монтаж]»
- Справочное руководство «S7-300, M7-300 Programmable Controllers, Module Specifications [Программируемые контроллеры S7-300, M7-300: Спецификации модулей]»
- Справочное руководство «S7-400, M7-400 Programmable Controllers, Module Specifications [Программируемые контроллеры S7-400, M7-400: Спецификации модулей]»

Начальный адрес модуля

Начальный адрес модуля – это адрес младшего байта модуля. Он представляет начальный адрес области пользовательских данных модуля и во многих случаях используется для представления всего модуля.

Например, начальный адрес модуля в аппаратных прерываниях, диагностических прерываниях, в прерываниях при установке/снятии модулей и в прерываниях при сбое источника питания вводится в стартовую информацию соответствующего организационного блока и используется для идентификации модуля, который инициализировал прерывание.

A.6.2 Доступ к области периферийных данных

Область периферийных данных можно разбить на следующие части:

- данные пользователя и
- данные диагностики и параметров.

Обе части имеют область входов (можно только считывать) и область выходов (можно только записывать).

Данные пользователя

Данные пользователя адресуются указанием адреса байта (для модулей цифровых сигналов) или адреса слова (для модулей аналоговых сигналов) в области входов или выходов. К данным пользователя можно обращаться при помощи команд загрузки или передачи, коммуникационных функций (доступ через интерфейс оператора) или посредством передачи образа процесса. Данными пользователя могут быть любые из следующих:

- цифровые и аналоговые сигналы ввода/вывода сигнальных модулей
- управляющая информация и информация о состоянии из функциональных модулей
- информация для соединений точка-точка и шинных соединений из коммуникационных модулей (только S7-300)

При передаче данных пользователя можно добиться непротиворечивости максимум 4 байтов (за исключением стандартных slave-устройств DP, см. раздел «Настройка рабочего режима»). Если вы используете оператор "transfer double word [передать двойное слово] ", то передаются четыре смежных и неизмененных (непротиворечивых) байта. Если вы используете четыре отдельные команды "transfer input byte [передать входной байт]", то OB аппаратного прерывания мог бы быть вставлен между этими командами и передать данные по тому же самому адресу так, что содержимое 4 исходных байтов изменилось бы прежде, чем все они были бы переданы.

Данные диагностики и параметров

Данные диагностики и параметров модуля не могут адресоваться индивидуально и всегда передаются в форме законченных наборов данных. Это означает, что всегда передаются непротиворечивые данные диагностики и параметров.

К данным диагностики и параметров обращаются, используя начальный адрес модуля и номер набора данных (DS). Наборы данных разделены на входные и выходные наборы данных. Входные наборы данных можно только читать, в выходные наборы данных можно только записывать. Вы можете обращаться к наборам данных, используя системные или коммуникационные функции (интерфейс пользователя). Следующая таблица показывает отношения между наборами данных и данными диагностики и параметров.

Данные	Описание
Данные диагностики	Если модули способны к диагностике, то вы получаете данные диагностики модуля, читая наборы данных 0 и 1
Данные параметров	Если модули являются конфигурируемыми, то вы передаете параметры модулю, записывая наборы данных 0 и 1

Доступ к наборам данных

Вы можете использовать информацию в наборах данных модуля, чтобы переназначать параметры модулей с перестраиваемой конфигурацией и читать диагностическую информацию из модулей, способных к диагностике.

Следующая таблица показывает, какие системные функции вы можете использовать для обращения к наборам данных.

SFC	Назначение
Назначение параметров модулям	
SFC55 WR_PARM	Передает модифицируемые параметры (набор данных 1) адресованному сигнальному модулю
SFC56 WR_DPARM	Передает параметры (набор данных 0 или 1) из SDB с номерами от 100 до 129 адресованному сигнальному модулю
SFC57 PARM_MOD	Передает параметры (набор данных 0 или 1) из SDB с номерами от 100 до 129 адресованному сигнальному модулю
SFC58 WR_REC	Передает любой набор данных адресованному сигнальному модулю
Считывание диагностической информации	
SFC59 RD_REC	Читает данные диагностики

Адресация модулей S5

Вы можете адресовать модули S5 следующим образом:

- Подключая S7-400 к стойкам расширения SIMATIC S5 с помощью интерфейсных модулей IM 463-2
- Вставляя некоторые модули S5 в корпус адаптера в центральной стойке S7-400

Как адресовать модули S5 при помощи SIMATIC S7, объясняется в руководстве "S7-400, M7-400 Programmable Controllers, Hardware and Installation [Программируемые контроллеры S7-400, M7-400: Аппаратные средства и монтаж]" и в описании, поставляемом с корпусом адаптера.

A.7 Настройка рабочего режима

A.7.1 Настройка рабочего режима

Эта глава объясняет, как вы можете изменять некоторые свойства программируемых контроллеров S7-300 и S7-400, регулируя системные параметры или используя системные функции (SFC).

Вы найдете подробную информацию о параметрах модулей в оперативной справке STEP 7 и в следующих руководствах

- Руководство "S7-300 Programmable Controller, Hardware and Installation [Программируемый контроллер S7-300: Аппаратные средства и монтаж]"
- Справочное руководство "S7-300, M7-300 Programmable Controllers, Module Specifications [Программируемые контроллеры S7-300, M7-300: Спецификации модулей]"
- Справочное руководство "S7-400, M7-400 Programmable Controllers, Module Specifications [Программируемые контроллеры S7-400, M7-400: Спецификации модулей]"

Вы найдете все, что вам необходимо знать об SFC, в справочном руководстве "Системное программное обеспечение для S7-300 и S7-400: Системные и стандартные функции".

Адресация стандартных Slave-устройств DP

Если вы хотите обмениваться данными длиной более 4 байтов со стандартными slave-устройствами DP, то вы должны использовать для такого обмена данными специальные SFC.

SFC	Назначение
Назначение параметров модулям	
SFC15 DPWR_DAT	Передает любой набор данных адресованному сигнальному модулю
Считывание диагностической информации	
SFC13 DPNRM_DG	Читает диагностическую информацию (асинхронный доступ для чтения)
SFC14 DPRD_DAT	Читает непротиворечивые данные диагностики (длиной 3 байта или более 4 байтов)

Когда поступает кадр диагностики DP, в CPU передается диагностическое прерывание с 4 байтами данных диагностики. Вы можете считывать эти 4 байта, используя SFC13 DPNRM_DG. Всю диагностическую информацию DP можно считать при помощи SFC14 DPRD_DAT, указав диагностический адрес стандартного slave-устройства DP.

A.7.2 Изменение режима и характеристик модулей

Настройки по умолчанию

- При поставке все конфигурируемые модули программируемого контроллера S7 имеют настройки по умолчанию, подходящие для стандартных приложений. С этими значениями по умолчанию вы можете использовать модули сразу, не выполняя каких-либо настроек. Значения по умолчанию объясняются в описаниях модулей в следующих руководствах:
- Руководство "S7-300 Programmable Controller, Hardware and Installation [Программируемый контроллер S7-300: Аппаратные средства и монтаж]"
- Справочное руководство "S7-300, M7-300 Programmable Controllers, Module Specifications [Программируемые контроллеры S7-300, M7-300: Спецификации модулей]"
- Справочное руководство "S7-400, M7-400 Programmable Controllers, Module Specifications [Программируемые контроллеры S7-400, M7-400: Спецификации модулей]"

Каким модулям вы можете назначать параметры?

Вы можете изменять поведение и свойства модулей так, чтобы адаптировать их к вашим требованиям и ситуации на вашей установке. Конфигурируемыми модулями являются CPU, FM, CP и некоторые модули аналогового ввода/вывода и модули цифрового ввода.

Есть конфигурируемые модули с резервным батарейным питанием и без него.

Модули без резервного батарейного питания после любого выключения питания должны вновь снабжаться данными. Параметры этих модулей хранятся в сохраняемой области памяти CPU (косвенное назначение параметров посредством CPU).

Настройка и загрузка параметров

Вы устанавливаете параметры модулей, используя STEP 7. Когда вы сохраняете параметры, STEP 7 создает объект ”System Data Blocks [Системные блоки данных]", который загружается в CPU программой пользователя и передается модулям при запуске CPU.

Какие параметры можно настраивать?

Параметры модулей делятся на блоки параметров. Какие блоки параметров доступны и на каких CPU, объясняется в руководстве "S7-300 Programmable Controller, Hardware and Installation [Программируемый контроллер S7-300: Аппаратные средства и монтаж]" и в справочном руководстве "S7-400, M7-400 Programmable Controllers, Module Specifications [Программируемые контроллеры S7-400, M7-400: Спецификации модулей]".

Примеры блоков параметров:

- Режим запуска
- Цикл
- MPI
- Диагностика
- Сохраняемые данные
- Тактовые меркеры
- Обработка прерываний
- Встроенные входы/выходы (только для S7-300)
- Уровень защиты
- Локальные данные
- Часы реального времени
- Асинхронные ошибки

Назначение параметров с помощью SFC

В дополнение к назначению параметров с помощью STEP 7, вы можете также включать в программу S7 системные функции для изменения параметров модулей. Следующая таблица показывает, какими SFC какие параметры модулей передаются.

SFC	Назначение
SFC55 WR_PARM	Передает модифицируемые параметры (набор данных 1) адресованному сигнальному модулю
SFC56 WR_DPARM	Передает параметры (набор данных 0 или 1) из соответствующих SDB адресованному сигнальному модулю
SFC57 PARM_MOD	Передает все параметры (наборы данных 0 и 1) из соответствующих SDB адресованному сигнальному модулю
SFC58 WR_REC	Передает любой набор данных адресованному сигнальному модулю

Системные функции подробно описаны в справочном руководстве "Системное программное обеспечение для S7-300 и S7-400: Системные и стандартные функции".

Какие параметры модулей можно динамически изменять, объясняется в следующих руководствах:

- Руководство "S7-300 Programmable Controller, Hardware and Installation [Программируемый контроллер S7-300: Аппаратные средства и монтаж]"
- Справочное руководство "S7-300, M7-300 Programmable Controllers, Module Specifications [Программируемые контроллеры S7-300, M7-300: Спецификации модулей]"
- Справочное руководство "S7-400, M7-400 Programmable Controllers, Module Specifications [Программируемые контроллеры S7-400, M7-400: Спецификации модулей]"

A.7.3 Использование функций часов

Все CPU S7-300/S7-400 оборудованы часами (часы реального времени или программные часы). Часы можно использовать в программируемом контроллере и как главные часы [master-часы], и как подчиненные часы [slave-часы] с внешней синхронизацией. Часы требуются для прерываний по времени и счетчиков рабочего времени.

Формат времени

Часы всегда показывают время (минимальная разрешающая способность 1 секунда), дату и день недели. В некоторых CPU возможно также указание миллисекунд (обратитесь к руководству "S7-300 Programmable Controller, Hardware and Installation [Программируемый контроллер S7-300: Аппаратные средства и монтаж]" и к справочному руководству "S7-400, M7-400 Programmable Controllers Module Specifications [Программируемые контроллеры S7-400, M7-400: Спецификации модулей]").

Установка и чтение времени

Вы устанавливаете время и дату для часов CPU, вызывая SFC0 SET_CLK в программе пользователя или используя пункт меню для запуска часов в устройстве программирования. Используя SFC1 READ_CLK или пункт меню в устройстве программирования, вы можете читать текущую дату и время в CPU.

Задание параметров часов

Если в сети существует более одного модуля, оборудованного часами, то вы должны с помощью STEP 7 установить параметры, чтобы указать, какой CPU функционирует в качестве главного и какой в качестве подчиненного при синхронизации времени. При установке этих параметров вы также решаете, синхронизируется ли время через коммуникационную шину или через многоточечный интерфейс, и выбираете интервалы, через которые время автоматически синхронизируется.

Синхронизация времени

Чтобы гарантировать, что время является одинаковым во всех модулях сети, подчиненные часы синхронизируются системной программой через регулярные (выбираемые) интервалы времени. Вы можете передавать дату и время из главных часов подчиненным часам, используя системную функцию SFC48 SFC_RTCB.

Использование счетчика рабочего времени

Счетчик рабочего времени подсчитывает часы работы подключенного оборудования или общее количество часов работы CPU.

В состоянии STOP счетчик рабочего времени останавливается. Его счетное значение сохраняется даже после сброса памяти. Во время «теплого» рестарта, счетчик рабочего времени должен перезапускаться программой пользователя; во время «горячего» рестарта, он продолжает работу автоматически, если уже был запущен.

Вы можете устанавливать счетчик рабочего времени на начальное значение, используя SFC2 SET_RTM. Вы можете запускать или останавливать счетчик рабочего времени с помощью SFC3 CTRL_RTM. Вы можете считывать текущее общее количество часов работы и состояние счетчика ("остановился" или ”считает") с помощью SFC4 READ_RTM.

CPU может иметь до восьми счетчиков рабочего времени. Нумерация начинается с 0.

A.7.4 Использование тактовых сигналов и таймеров

Тактовые сигналы

Тактовые сигналы – это байт памяти, который периодически изменяет свое двоичное состояние с отношением паузы к импульсу 1:1. Вы выбираете, какой байт памяти используется в CPU, когда вы с помощью STEP 7 назначаете параметры для тактовых сигналов.

Применение

Вы можете использовать байты тактовых меркеров в программе пользователя, например, для активизации проблесковых ламп или запуска периодических действий (например, измерение фактического значения).

Возможные частоты

Каждому биту байта тактовых сигналов назначена частота. Следующая таблица показывает это назначение:

Бит байта тактовых сигналов	7	6	5	4	3	2	1	0
Длительность периода (с)	2.0	1.6	1.0	0.8	0.5	0.4	0.2	0.1
Частота (Гц)	0.5	0.625	1	1.25	2	2.5	5	10

Примечание

Байты тактовых сигналов не синхронны с циклом CPU, другими словами, в длинных циклах состояние байта тактовых сигналов может изменяться несколько раз.

Таймеры

Таймеры – это область в системной памяти. В программе пользователя вы определяете функцию таймера (например, таймер с задержкой включения). Число доступных таймеров зависит от CPU.

Примечание

- Если вы используете в Вашей программе пользователя больше таймеров, чем позволяет CPU, то формируется сигнал синхронной ошибки и запускается OB121.
- В S7-300 (за исключением CPU 318) таймеры могут одновременно запускаться и обновляться только в OB1 и OB100; во всех других OB таймеры могут только запускаться.