

Создание исходных файлов для прикладной программы

6

Что описывает эта глава?

файла в текстовом редакторе.

В качестве альтернативы созданию прикладной программы на AWL в инкрементном редакторе Вы можете создать программу на AWL и в виде исходного

Так как текстовый редактор для AWL настроен над SIMATIC Manager, который является платформой для всех приложений STEP 7, то Вы здесь узнаете, как попасть в текстовый редактор из SIMATIC Manager и где расположены созданные исходные файлы в структуре проекта.

Обзор главы

В разделе	Вы узнаете	на стр.
6.1	Создание прикладных программ	6–2
6.2	Программирование в текстовом редакторе	6–6
6.3	Вставка шаблонов блоков, блоков и исходных файлов	6–7
6.4	Применение символической адресации	6–8
6.5	Сохранение, проверка консистентности и компиляция исходных файлов	6–9

6.1. Создание прикладных программ

Прикладная программа

Прикладная программа, которая должна исполняться на S7-CPU, состоит в основном из блоков. Кроме того, она содержит дополнительную информацию, как, например, данные для конфигурации системы или о включении системы в сеть. В зависимости от обстоятельств применения Вы должны создать для прикладной программы следующие блоки:

- организационные блоки (OB)
- функциональные блоки (FB)
- функции (FC)
- блоки данных (DB).

Наряду с этим Вы можете для упрощения работы создавать определенные Вами типы данных (UDT), который Вы можете использовать как собственный тип данных или как шаблон для создания DB.

Некоторые часто используемые блоки, как системные функциональные блоки (SFB) и системные функции (SFC), встроены в CPU. Другие блоки (например, для IEC-функций или блоков регулирования) загружаются как пакет. Эти не Вами запрограммированные блоки Вы можете очень просто вставить в свою прикладную программу.

Указание

Какие SFB и SFC встроены в Ваш CPU, Вы можете отобразить на экране в режиме online командой меню **Zielsystem** → **Baugruppenzustand** (Контроллер → Состояние модуля).

Текстовый редактор для AWL

Для программирования блоков в исходных файлах Вам вместе с базовым программным пакетом предоставляется в распоряжение текстовый редактор с компилятором. Тем самым Вы имеете возможность вводить свою программу на AWL в виде исходного файла, а затем в один прием скомпилировать ее в блоки. Во время ввода проверка синтаксиса не производится. Создание прикладной программы через исходный файл имеет следующие преимущества:

- Вы можете в одном исходном файле запрограммировать несколько блоков.
- Исходный файл можно сохранять с синтаксическими ошибками. Это невозможно при разработке блоков в инкрементном редакторе AWL. С другой стороны, Вы обращаете внимание на синтаксические ошибки только при проверке консистентности и компиляции исходного файла.
- Вы можете создавать исходный файл другими редакторами, импортировать их в SIMATIC Manager и затем преобразовать их в блоки.

Исходный файл

Исходный файл может содержать код для всей прикладной программы или же только для одного или нескольких блоков. Для возможности преобразования исходного файла в блоки должны быть приняты во внимание некоторые правила относительно структуры и синтаксиса.

Исходный код для каждого блока имеет типовую структуру. Отдельные блоки, описания переменных, операторная часть и сети обозначаются ключевыми словами.

Запуск из Manager

Текстовый редактор запускается из SIMATIC Manager. Предпосылкой является создание в нем проекта с программой S7. Вы можете создавать программу независимо или независимо от аппаратного обеспечения. Для этого Вы вставляете программу S7 (S7-Programm) непосредственно под проектом или редактируете программу, поставленную в соответствие программируемому модулю. Сама программа, среди прочего, может содержать контейнеры для прикладной программы (блоки), исходные тексты или планы.

С помощью текстового редактора редактируются исключительно исходные файлы, которые затем компилируются в блоки, сохраняемые в контейнере прикладной программы.

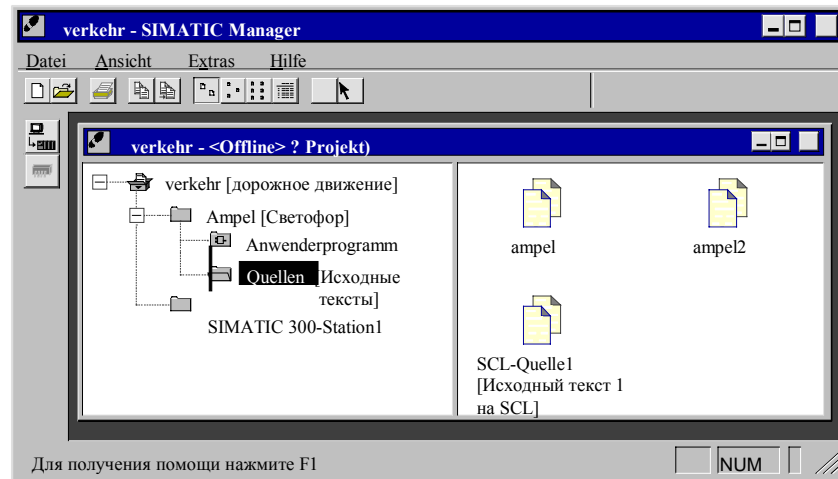


Рис. 6-1. Структура проекта с исходными файлами

Генерация исходного файла

Когда Вы впервые хотите создать новый исходный файл, Вы должны в первую очередь сгенерировать в SIMATIC Manager пустой исходный файл, через который Вы откроете текстовый редактор. Когда текстовый редактор открыт, Вы можете в нем создавать другие исходные файлы.

- В SIMATIC Manager пометьте контейнер Quellen (Исходные тексты) и вставьте файл командой меню **Einfügen** ► **AWL-Quelle** (Вставка ► Исходный текст на AWL). В правой части окна проекта появляется новый файл с предустановленным именем.
- В самом текстовом редакторе Вы можете создать новый файл просто через команду меню **Datei** ► **Neu** (Файл ► Новый). В последующем диалоге введите имя нового исходного файла.

**Открытие
исходного файла
öffnen** (Редактирование ▶

Исходный файл открывается в SIMATIC Manager двойным щелчком на его символе. Этого можно достичь также через команду меню **Bearbeiten ▶ Objekt** (Открытие объекта) или через соответствующий символ на панели инструментов.

**Создание исходного
файла из блоков**

У Вас есть также возможность преобразовать уже существующие блоки обратно в исходный файл, чтобы их там редактировать дальше. Для этого выберите в текстовом редакторе команду меню **Datei ▶ Quelldatei generieren** (Файл ▶ Генерирование исходного файла). В последующем диалоге Вы можете выбрать все блоки, из которых Вы хотите сгенерировать исходный файл.

**Редактирование
файла**

На рис. 6-2 Вы видите все шаги, необходимые для создания прикладной программы в виде исходного файла, в их взаимосвязи. При вводе кода примите во внимание необходимые ключевые слова, структуры и синтаксические правила. При редактировании Вы можете использовать обычные редакторские функции, как вырезание, копирование и т.д. Кроме того, Вы можете вставлять шаблоны блоков, другие исходные файлы или блоки.

Если Вы хотите обратиться к символам таблицы символов, Вам следует ее проверить на полноту и в случае необходимости дополнить. Исходный файл и соответствующая таблица символов перед компиляцией должны находиться в одной и той же S7-программе в SIMATIC Manager.

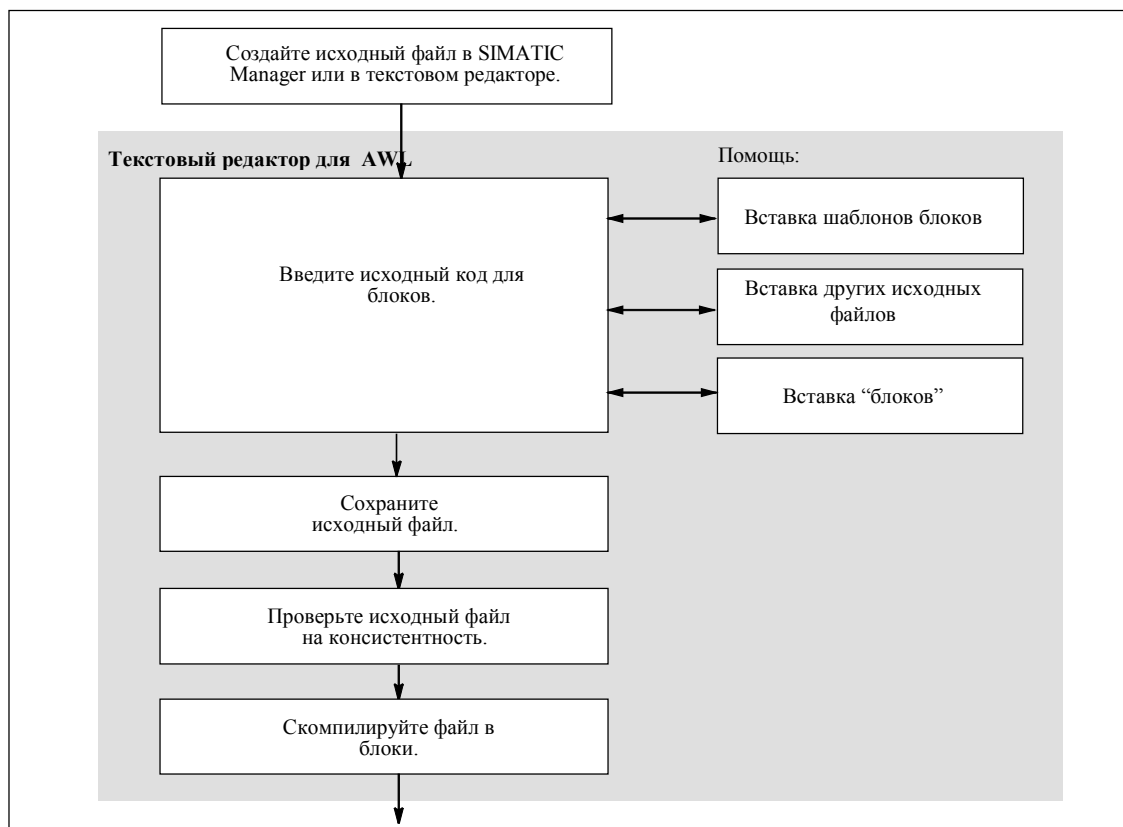


Рис. 6-2. Последовательность действий при создании исходных файлов

6.2. Программирование в текстовом редакторе

Настройка редактора

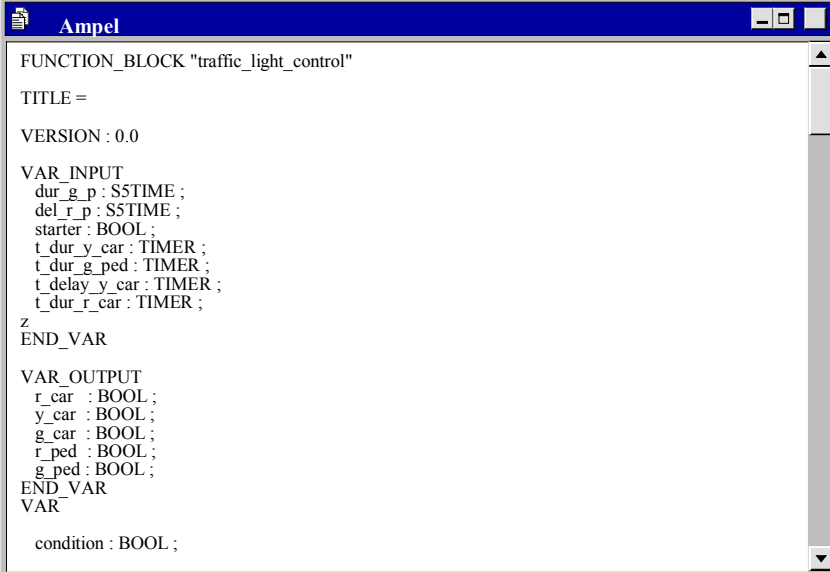
возможности удобно и в соответствии с Вашими привычками.

Перед началом программирования в текстовом редакторе Вы должны познакомиться с возможностями настройки, чтобы иметь возможность работать по

Командой меню **Extras ► Einstellungen** (Дополнительные функции ► Настройка) откройте регистровый диалог. На вкладке “Editor” (“Редактор”) Вы можете сделать предварительную установку для шрифта (вид, стиль и величина) в исходном файле. Цвет, которым выделяются строки команд, измените на вкладке “KOP”.

Исходный файл в текстовом редакторе

На рис. 6-3 показан исходный файл в текстовом редакторе:



```
FUNCTION_BLOCK "traffic_light_control"

TITLE =

VERSION : 0.0

VAR_INPUT
    dur_g_p : S5TIME ;
    del_r_p : S5TIME ;
    starter : BOOL ;
    t_dur_y_car : TIMER ;
    t_dur_g_ped : TIMER ;
    t_delay_y_car : TIMER ;
    t_dur_r_car : TIMER ;
END_VAR

VAR_OUTPUT
    r_car : BOOL ;
    y_car : BOOL ;
    g_car : BOOL ;
    r_ped : BOOL ;
    g_ped : BOOL ;
END_VAR

VAR
    condition : BOOL ;
```

Рис. 6-3. Текстовый редактор для AWL с исходным файлом

6.3. Вставка шаблонов блоков, блоков и исходных файлов

Вставка шаблонов Для упрощения программирования в редактор встроены шаблоны блоков **блоков** для OB, FB, FC, DB, экземпляров DB, DB из UDT и UDT. Шаблон блока содержит все необходимые ключевые слова в требуемом порядке. Предварительно установленные значения для необязательных данных, которые Вы не хотите использовать, просто сотрите. Благодаря шаблонам блоков облегчается ввод и соблюдение синтаксиса и структуры.

Чтобы вставить шаблон блока в исходный файл, выберите команду меню **Einfügen ▶ Bausteinvorlage ▶ OB/FB/FC/DB/IDB/DB aus UDT/UDT** (Вставка ▶ Шаблон блока ▶ OB/FB/FC/DB/IDB/DB из UDT/UDT).

Вставка блоков Вы можете вставить в свой исходный файл соответствующий исходный код из уже готовых блоков. Для этого выберите команду меню **Einfügen ▶ Objekt ▶ Baustein** (Вставка ▶ Объект ▶ Блок). В последующем диалоге выберите блоки, код которых Вы хотели бы вставить в виде текста.

Из выбранных блоков неявно генерируется соответствующий исходный файл. Его содержимое вставляется в редактируемый в данный момент файл вслед за позицией курсора.

Вставка исходных файлов В свой исходный файл Вы можете вставить содержимое любых других исходных файлов. Для этого выберите команду меню **Einfügen ▶ Objekt ▶ Datei** (Вставка ▶ Объект ▶ Файл) и выберите в следующем диалоговом окне файл для вставки.

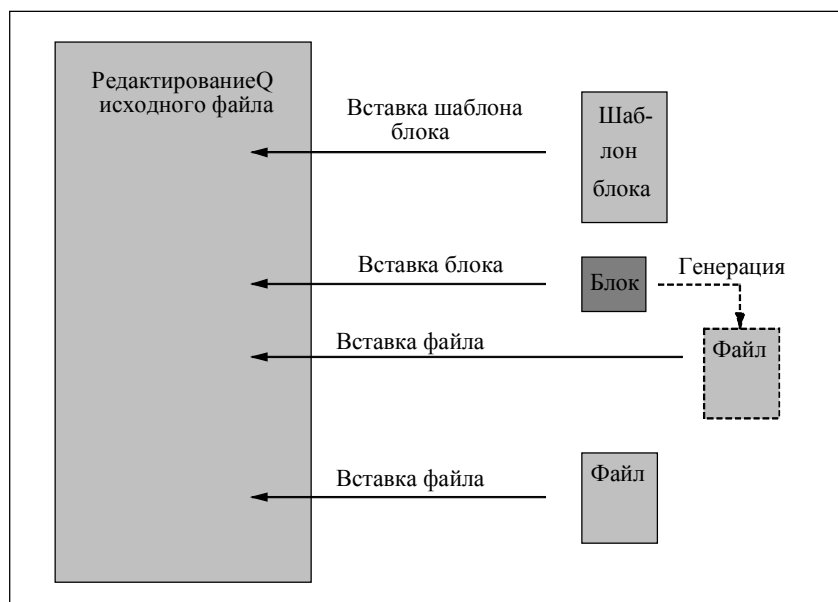


Рис. 6-4. Вставка других исходных файлов, блоков и шаблонов блоков

6.4. Применение символической адресации

Обзор

В языке программирования AWL адреса, параметры и имена блоков можно указывать абсолютно или символически. Переключение между двумя видами представления адресации производится командой меню **Ansicht ► Symbolische Darstellung** (Вид ► Символическое представление). Если символика выключена, то адресоваться можно только абсолютно; если она включена, то возможны оба вида адресации.

Указание

Если Вы хотите применить глобальные символы, то перед компиляцией Вы должны их внести в таблицу символов. Таблица символов должна находиться в той же S7-программе, что и контейнер с исходным файлом.

Представление

В большинстве случаев Вам не требуется обозначать, идет ли речь о локальных в блоке или глобальных символах. Если символ никак особо не отмечен, то он в первую очередь интерпретируется как локальная переменная. Однако, если возможна путаница, то Вы можете различить символы следующим образом:

- Символы из **таблицы символов** представляются в кавычках `".."`.
- Символы из таблицы описания переменных блока предваряются знаком `"#"`.

Символические имена из таблицы символов необходимо заключать в кавычки тогда,

- когда имя неоднозначно (одинаковые имена для локальных переменных и переменных из таблицы символов) или
- когда имя идентично зарезервированному ключевому слову (см. приложение C). Однако, от применения зарезервированных ключевых слов в качестве символов Вам следует по возможности отказаться.

Указание

Если Вы при включенной символике генерируете исходный файл из блока, то символы однозначно помечаются кавычками или знаком `"#"`.

6.5. Сохранение, проверка консистентности и компиляция исходных файлов

Сохранение Исходный файл в текстовом редакторе Вы в любое время можете **исходного файла** сохранить в текущем состоянии. Программа при этом не компилируется. Ошибки в коде также сохраняются.

Предпосылка для проверки консистентности и компиляции Для возможности проверки Вашего исходного файла на консистентность или его преобразования в блоки, он должен находиться в структуре проекта в контейнере исходных файлов выбранной S7-программы. Только так компилятор может в случае необходимости установить взаимосвязь, например, с уже имеющимися блоками или с таблицей символов.

Проверка Командой меню **Datei ► Konsistenz prüfen** (Файл ► Проверка **консистентности** консистентности) Вы можете в любое время проверить синтаксис и консистентность исходного файла, не инициируя при этом генерацию блоков.

При этом проверяется, среди прочего, синтаксис, символика и существование вызываемых блоков. Вы получаете протокол компиляции, в котором указывается имя компилировавшегося файла, количество скомпилированных строк, количество ошибок и предупреждений.

Устранение ошибок Если имеются ошибки и/или предупреждения, то они перечисляются под исходным файлом во втором подокне с указанием причины ошибки. Если Вы пометите сообщение об ошибке, то в исходном файле будет показано соответствующее место ошибки. Эта связь сообщения об ошибке и места ошибки дает возможность быстрее устранить ошибку.

Исправление ошибок и изменения Вы можете выполнять в режиме замены. Переключение между режимами вставки и замены производится с помощью клавиши INSERT.

Компиляция исходного файла Командой меню **Datei ► Übersetzen** (Файл ► Компилировать) Вы компилируете исходный файл в блоки.

После компиляции появляется протокол компиляции. Ошибки отображаются, как при проверке консистентности. Если в исходном файле запрограммировано несколько блоков, то компилируются и сохраняются только блоки, не содержащие ошибок.

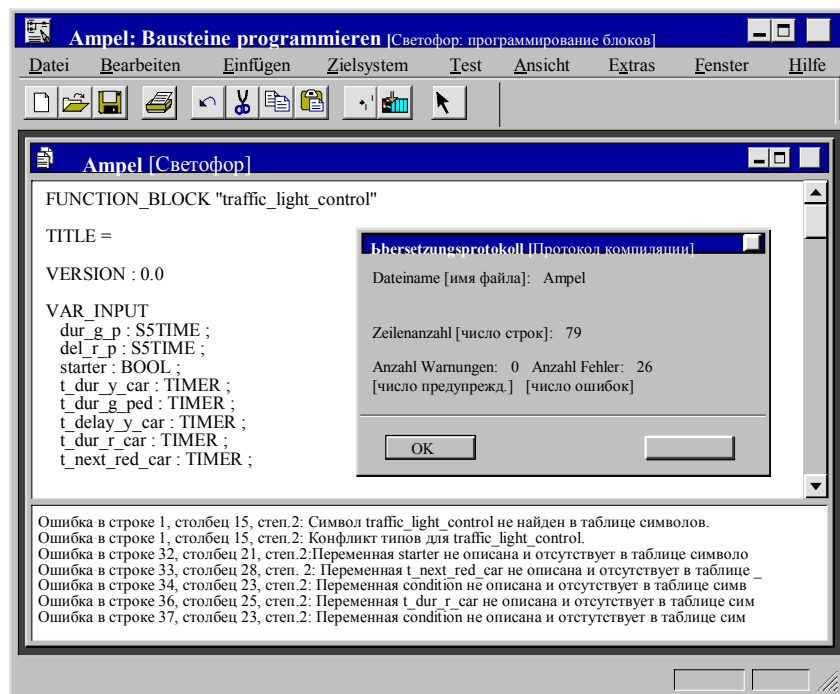


Рис. 6-5. Проверка консистентности и компиляция исходных файлов