Глава 2. Битовые операции, операции сравнения и арифметические операции

Битовые операции

Битовые операции находятся в папке Bit Logic дерева инструкций Instruction Tree. Содержание этой папки определяется выбранным редактором программ [1]. На рис. 2-1 оно приведено для редактора контактных планов LAD, а на рис. 2-2 – для редактора функциональных блоков FBD. Соответствующие команды в редакторе списка команд STL приводятся в тексте главы.

Команды опроса:

Опрос на состояние логической 1 – как отдельная операция отображается только в редакторе контактных планов LAD в виде нормально разомкнутого контакта (рис. 2-1). Данный контакт считается замкнутым, когда значение бита с указанным адресом равно 1.

Опрос на состояние логического 0 – как отдельная операция отображается только в редакторе контактных планов LAD в виде нормально замкнутого контакта (рис.2-1) – замкнут, когда значение бита с указанным адресом равно 0.

В редакторе функциональных блоков FBD команды, соответствующие нормально открытым контактам, представлены (рис. 2-2) блоками AND (логическое И) и OR (ИЛИ). Каждому контакту в LAD соответствует вход блока в FBD: прямой для нормально разомкнутого или инверсный для нормально замкнутого контакта. Соответственно основам алгебры логики [3] блоку логического умножения AND соответствует последовательное, а блоку логического сложения OR – параллельное соединение контактов. Результат логической операции присваивается биту с адресом, указанном на выходе блока. В LAD такому выходу соответствует катушка, в STL – операция присваивания (см. ниже). Количество входов блоков AND (И) и OR (ИЛИ) в строке программы может быть не более 7. Результаты логических операций записываются в вершину стека, сдвигая его содержимое (рис. 2-3).

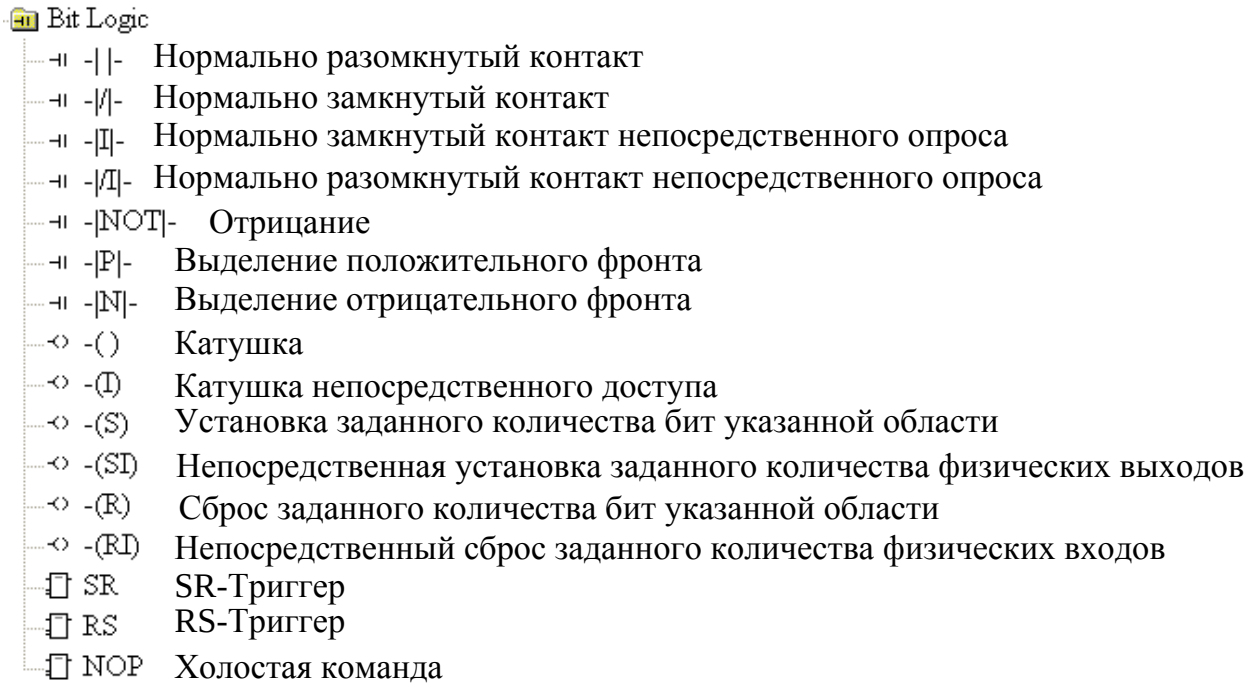


Рис. 2-1. Битовые логические операции в редакторе LAD

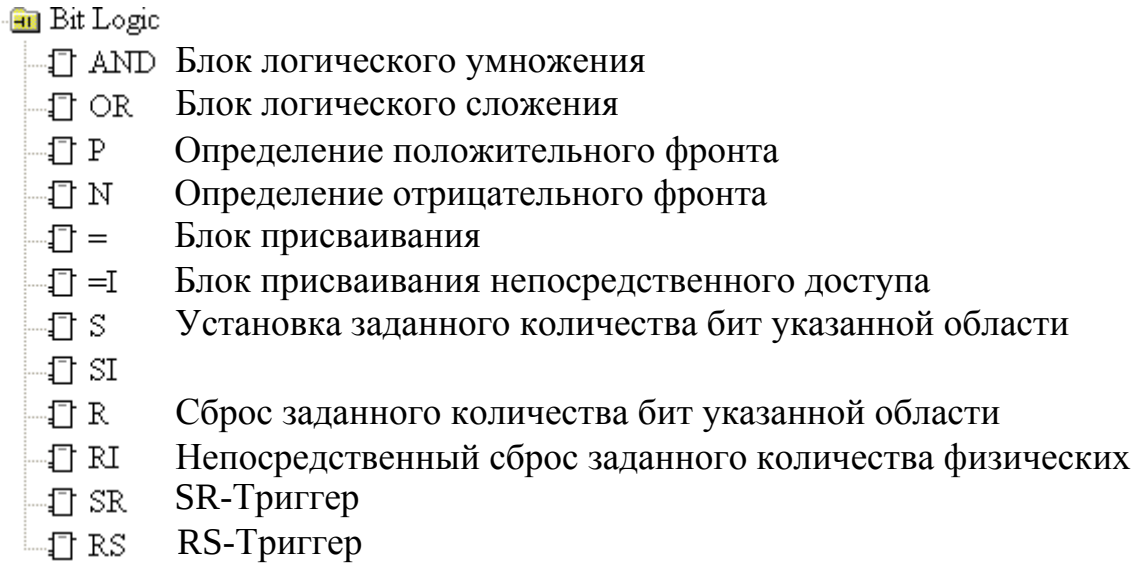


Рис. 2-2. Битовые логические операции в редакторе FBD

В STL нормально открытый/закрытый контакт представляется командами:

- LD / LDN - загружает значение бита/инверсии с указанным адресом в вершину стека (рис. 2-3,б);
- A / AN – логическое умножение значения бита/инверсии с указанным адресом и значения в вершине стека (рис. 2-3,в);

- O / ON - логическое сложение значения бита/инверсии с указанным адресом и значения в вершине стека (рис. 2-3,в).

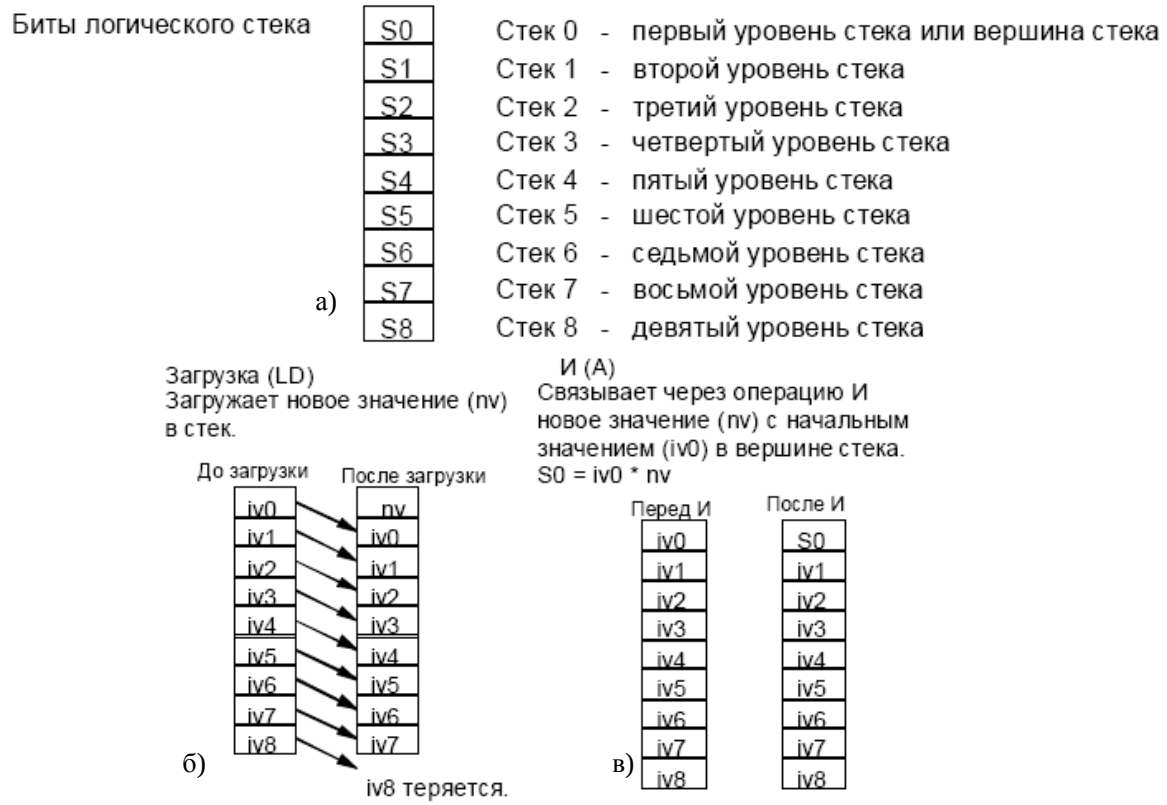
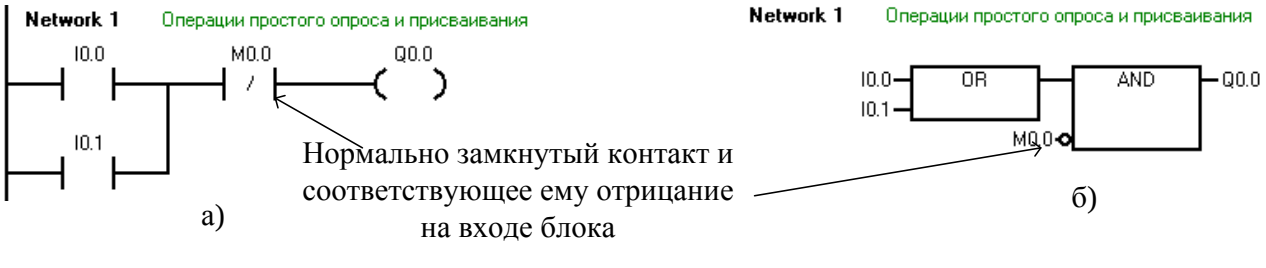


Рис. 2-3. Логический стек:
а – структура; б – обновление при загрузке (первичном опросе);
в – обновление при выполнении логической операции

Пример применения команд опроса и присваивания приведен на рис. 2-4. Здесь реализована логическая функция $Q0.0 = I0.0 \vee I0.1 \& \overline{M0.0}$




```
Network 1    Операции простого опроса и присваивания
LD     I0.0                  //Первичный опрос
O      I0.1                  //Сложение результата первичного опроса и бита
AN     M0.0                  //Умножение результата сложения и бита
=      Q0.0                  //Присваивание
```

в)

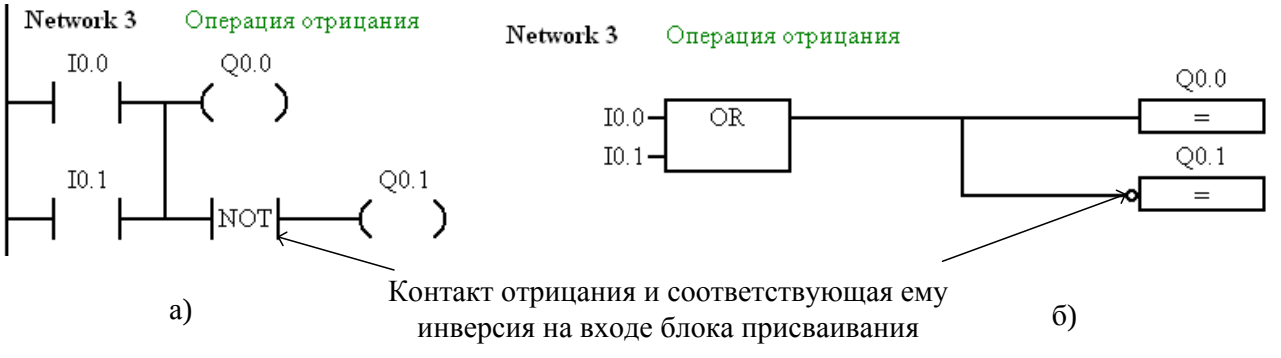
Рис. 2-4. Пример программы с простыми операциями опроса и присваивания:
а – в редакторе LAD; б – в редакторе FBD; в – в редакторе STL

Команды непосредственного опроса (рис. 2-1 и 2-2) работают только с адресами физических цифровых входов I #.#. Они загружают состояние цифрового входа с заданным адресом в логический стек непосредственно во время выполнения команды. Область памяти I при этом не обновляется.

Операция отрицания NOT – изменяет состояние сигнала на противоположное.

- В редакторе LAD команда NOT изображается как контакт (рис. 2-1).
- В FBD команда NOT обозначается отрицанием на входе блока (рис. 2-5).
- В STL команда NOT изменяет значение в вершине стека с 0 на 1 или с 1 на 0.

Пример применения команды NOT приведен на рис. 2-5. Здесь осуществляются логические операции ИЛИ и ИЛИ-НЕ: $Q0.0 = I0.0 \vee I0.1$ и $Q0.0 = \overline{I0.0 \vee I0.1}$



а) б)

Network 3 Операция отрицания

```
LD      I0.0          //Первичный опрос
O       I0.1          //Сложение результата первичного опроса и бита
=       Q0.0          //Присваивание значения суммы выходу Q0.0
NOT                   //Инверсия суммы
=       Q0.1          //Присваивание инверсии суммы выходу Q0.1
```

в)

Рис. 2-5. Пример программы с операцией отрицания:
а – в редакторе LAD; б – в редакторе FBD; в – в редакторе STL

Выделение фронта сигнала:

В редакторе LAD контакт Положительный/Отрицательный фронт замкнут в течение одного цикла при каждом появлении положительного/отрицательного фронта указанного бита.

В FBD эти команды представлены блоками P/N.

В STL эта команда обозначается как EU/ED - при обнаружении перехода значения в вершине стека (с 0 на 1)/ (с 1 на 0) значение в вершине стека устанавливается в 1; в противном случае оно сбрасывается в 0.

Присваивание – осуществляет присваивание результата логической операции биту с указанным адресом. Операция работает со всеми областями памяти, к которым возможна адресация в формате бита.

- В редакторе LAD данная операция представлена катушкой с указанием адреса.
- В редакторе FBD - это выход блока AND или OR.
- В редакторе STL представлена знаком равенства с указанием адреса бита, в который копируется содержимое вершины логического стека.

Команда **Непосредственный выход** присваивает физическому выходу с указанным адресом значение результата логической операции непосредственно во время выполнения команды, а не в заключительной части МЦ [1, 3], как при обычном присваивании. Одновременно обновляется соответствующий бит в области отображения информации на выходах.

Команды Установка и Сброс (рис. 2-6) – устанавливают в 1 или сбрасывают в 0 указанное в команде количество разрядов (N=1… 255), начиная с определенного в команде бита.

- В редакторе LAD данные операции представлены катушками со значками «S» и «R» соответственно (рис. 2-1) с указанием адреса начала области (верхний параметр) и количеством бит в области, с которой работают данные команды (нижний параметр).
- В редакторе FBD - это блоки S и R (рис. 2-2). Пример на рис. 2-6, в.
- В редакторе STL - это команды S и R с двумя параметрами: начальный адрес области памяти в формате бита и длина изменяемой области в битах (рис. 2-6, в).

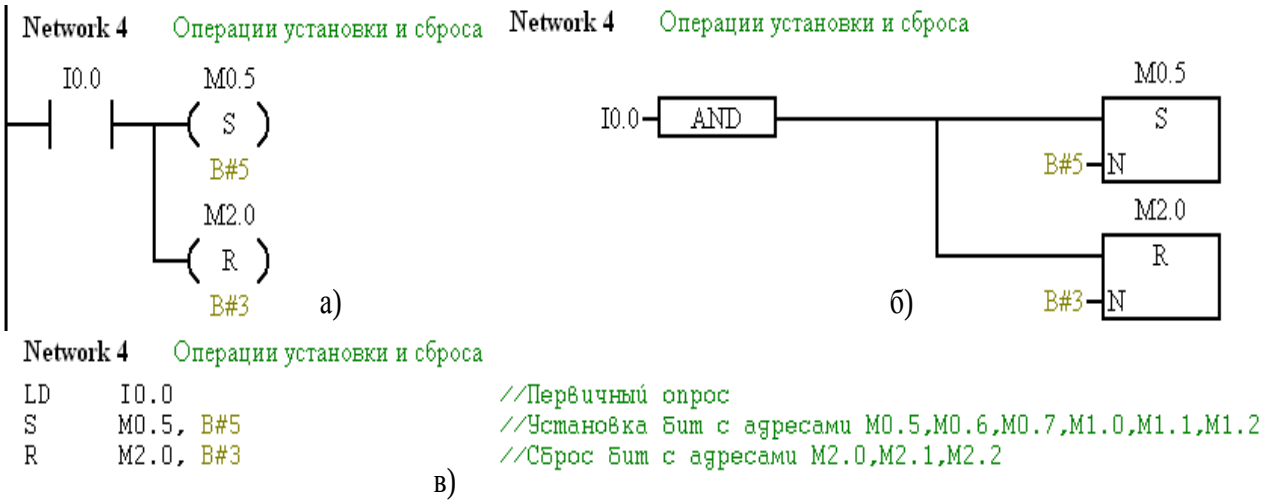


Рис. 2-6. Пример программы с операциями установки и сброса:
а – в редакторе LAD; б – в редакторе FBD; в – в редакторе STL

Команды **Непосредственная установка, непосредственный сброс** - работают только с физическими входами. Непосредственно при их исполнении указанное количество физических выходов (от 1 до 128), начиная с определенного в команде адреса, устанавливаются или сбрасываются соответственно команде. Кроме того, при исполнении данных команд новые значения записываются и в соответствующие адреса ячейки области отображения информации на выходах.

Команда **Пустая операция NOP** не оказывает влияния на исполнение программы пользователя. В LAD – это блок с именем NOP и операндом N, в FBD эта команда отсутствует. В STL она записывается как мнемоника NOP N, где операнд N – это число от 0 до 255. Используется при отладке проекта для создания резерва строк программы.

Операции сравнения

Операции сравнения имеют формат, представленный на рис. 2-7,а. Здесь

@ - обозначение символа сравнения из ряда: = = (равно), <, >, <= (меньше или равно), >= (больше или равно), <> (не равно);

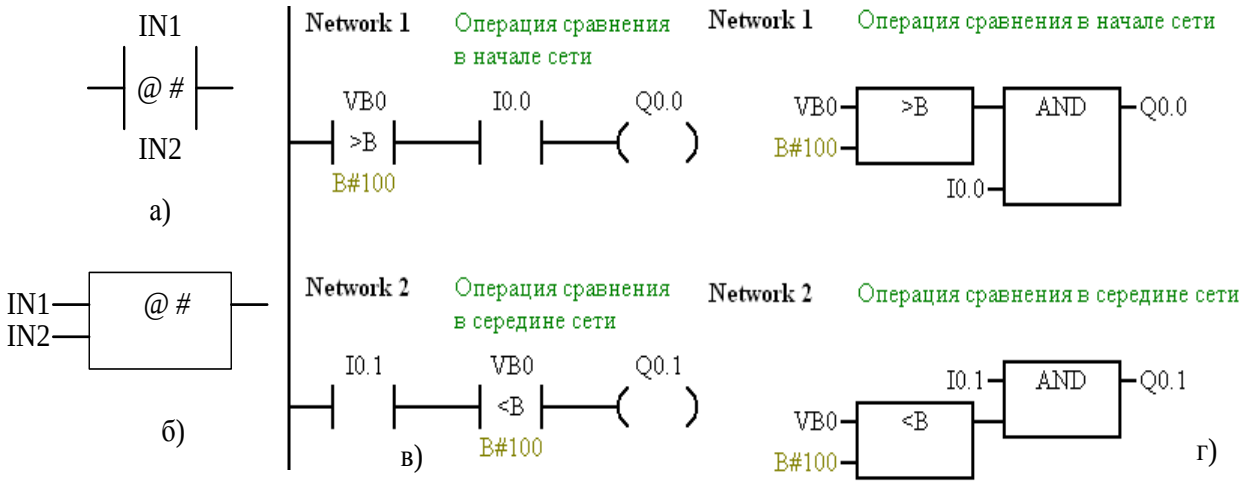
- обозначение формата представления данных IN1 и IN2 из ряда:

- B – формат байта без учета знаков чисел;
- I – формат целого 16-разрядного числа с учетом знака;
- D – формат целого 32-разрядного числа с учетом знака;

- R – формат действительного числа с учетом его знака.

В редакторе LAD эти команды представлены как контакты (рис. 2-7, а), замыкающиеся в том случае, когда сравнение истинно. Например, Network 1 на рис. 2-7, в читается следующим образом: если содержимое области VB0 больше 100, контакт операции сравнения считается замкнутым и состояние выхода Q0.0 копирует состояние контакта I0.0.

В редакторе FBD команды сравнения представляют собой блоки (рис. 2-7, г), на входы которых подаются сравниваемые операнды IN1 и IN2, а выход устанавливается в 1, если сравнение истинно и сбрасывается в 0, если оно ложно. Так, Network 1 на рис. 2-7, г: если содержимое области переменных VB0 больше 100, на выходе блока сравнения будет 1, она логически умножается на содержимое бита I0.0; на выход Q0.0 записывается результат этого умножения.




```
Network 1   Операция сравнения в начале сети
LDB>    VB0, B#100              //Загрузка в стек результата сравнения
A       I0.0                    //
=       Q0.0                    //

Network 2   Операция сравнения в середине сети
LD      I0.1                    //
AB<     VB0, B#100              //Логическое умножение вершины стека
                                // и результата сравнения
=       Q0.1                    //
```

д)

Рис. 2-7. Операция сравнения:
а – формат в редакторе LAD; б - формат в редакторе FBD;
в – пример программы в редакторе LAD;
г – пример программы в редакторе FBD; д – в редакторе списка команд STL

В редакторе STL, если сравнение истинно, то в вершину стека загружается 1.

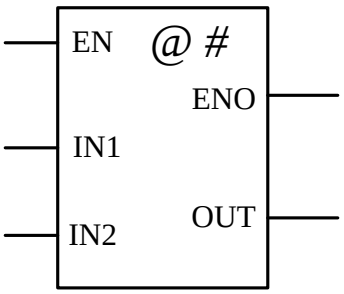
Арифметические операции

Эти операции находятся в папках Integer Math (Целочисленная математика, таблица 2-1) и Floating-Point Math (Математика с плавающей точкой).

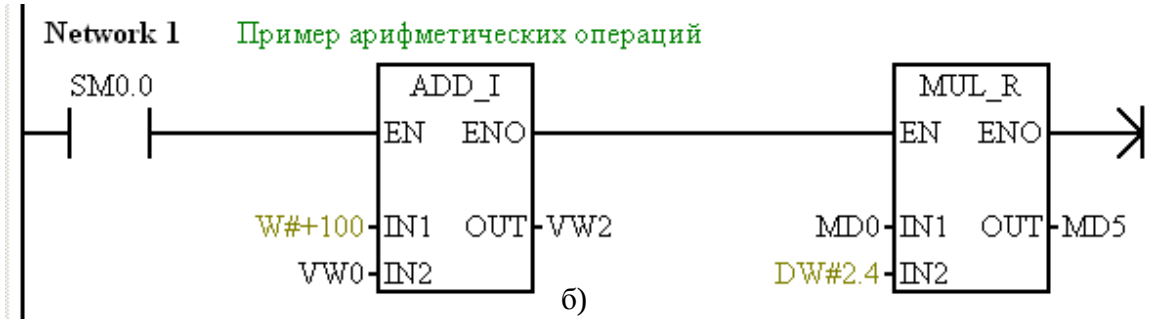
Общий вид большинства блоков этих папок показан на рис.2-8, а. Здесь @ - обозначение математических операций; # - формат представления данных. IN1 и IN2 – исходные данные для вычисления, в которых указываются операнды в виде констант или адреса размещения операндов в памяти в соответствующем формате. OUT – адрес приёмника результата вычислений.

Таблица 2-1. Операции Целочисленной математики

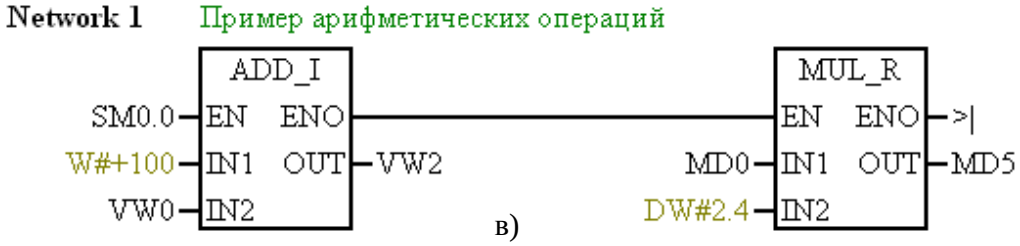
Мнемоника команд	Действие	Формат (количество разрядов) исходных данных	Формат (количество разрядов) результата
ADD_I и SUB_I	сложение и вычитание	16	16
ADD_DI и SUB_DI	сложение и вычитание	32	32
MUL	умножение	16	32
DIV	деление	16	32 (в двух СБ остаток деления, а в двух МБ - частное)
MUL_I	умножение	16	16
MUL_DI	умножение	32	32
DIV_I	деление	16	16-разрядное частное без сохранения остатка
DIV_DI	деление	32	32- разрядное частное без сохранения остатка
INC_B и DEC_B	увеличение и уменьшение на 1	8	8
INC_W и DEC_W	увеличение и уменьшение на 1	Слово со знаком	Слово со знаком
INC_DW и DEC_DW	увеличение и уменьшение на 1	Двойное слово со знаком	Двойное слово со знаком



а)



б)



в)

```
Network 1      Пример арифметических операций
LD      SM0.0                  //Опрос разрешения на операцию
MOVW    W#+100, VW2            //Первый операнд размещается
                               //по адресу приемника результата
AENO                           //Проверка корректности работы блока
                               //т.е. опрос бита ENO
+I      VW0, VW2               //Сложение с размещением результата
                               //по адресу 2 слагаемого
AENO                           //Проверка корректности работы блока
                               //т.е. опрос бита ENO
MOVR    MD0, MD5               //Первый операнд размещается
                               //по адресу приемника результата
*R      DW#2.4, MD5            //Умножение с размещением результата
                               //по адресу 2 сомножителя
```

г)

Рис. 2-8. Арифметические операции:
а - общий формат; б – пример программы в редакторе LAD;
в –в редакторе FBD; г – пример программы в редакторе STL

В папке Floating-Point Math (Математика с плавающей точкой) реализованы математические операции сложения (ADD_R), вычитания (SUB_R), умножения (MUL_R), деления (DIV_R), извлечение квадратного корня (SQRT), функция синуса (SIN), косинуса (COS), тангенса (TAN), натурального лога-

рифма (LN), получение экспоненты (EXP). Все входные и выходные параметры данных блоков имеют формат двойного слова.

Признаки арифметических операций хранятся в области специальных маркеров SMB1 (таблица П1-2 в приложении 1).

Операции преобразования

Операции преобразования находятся в папке Convert (Преобразование) дерева инструкций Instruction Tree, показанной на рис. 2-9.

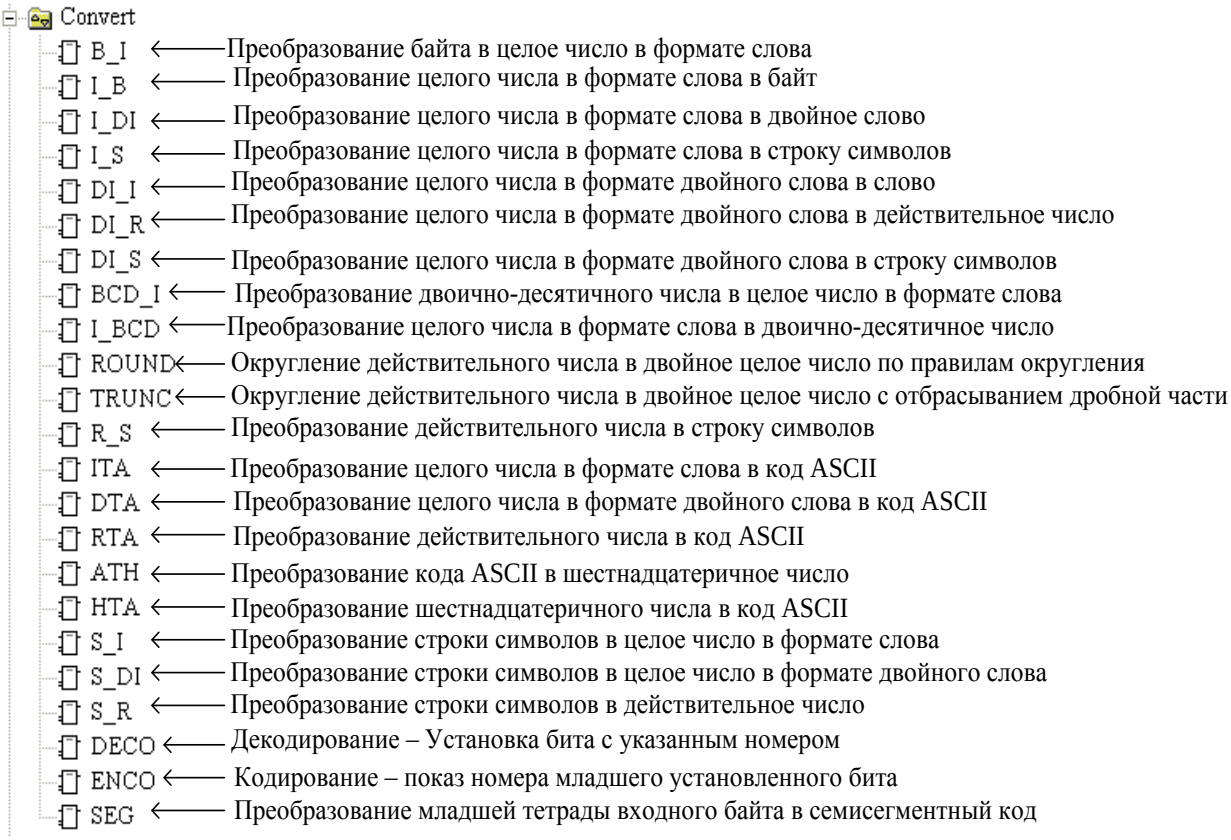


Рис. 2-9. Операции преобразования форматов в редакторах LAD и FBD

Команды данной группы работают с числами следующих форматов:

- B – формат байта без учета знака числа ;
- I - формат слова с учетом знака числа;
- DI (или D) - формат двойного слова с учетом знака числа;
- R – действительные числа;
- S – строка символов (от англ. String). Переменная данного типа состоит из нескольких (максимум 8) переменных символьного типа CHAR (от англ.

character - символ), каждая из которых занимает один байт и представляет собой отдельный символ в формате ASCII. Например, символ ’A’. Можно также записать любой печатаемый символ в одинарных кавычках;

- A – код символа в формате ASCII.

Первая программа

Формулировка задачи

Если содержимое области памяти переменных VB0 превышает 10, состояние выхода Q0.1 соответствует значению логического произведения бит с адресами I0.0 и I0.1. Если данное условие не выполняется, то состояние выхода Q0.1 соответствует значению логической суммы этих бит. Кроме того, в этом случае содержимое области VB0 увеличивается на 5.

Разработка программы

Полный текст программы в редакторе LAD приведен на рис. 2-10, в редакторе FBD – на рис. 2-11, в редакторе STL – на рис. 2-12.

Строка Network 1 реализует проверку поставленного в задании условия (VB0>10?). Если оно выполняется, контакт операции сравнения считается замкнутым и состояние выхода Q0.0 будет определяться информацией в битах I0.0 и I0.1.

Строка Network 2 будет выполнять логическое сложение бит I0.0 и I0.1 только в том случае, если VB0≤10.

Строка Network 3 реализует увеличение VB0 на 5, если число в VB0≤10. Если данное условие выполняется, происходит сначала преобразование формата байта в формат Integer. Необходимость данного преобразования связана с отсутствием арифметических операций в формате байта. Далее следует собственно сложение, а затем – обратное преобразование из формата Integer в формат байта.

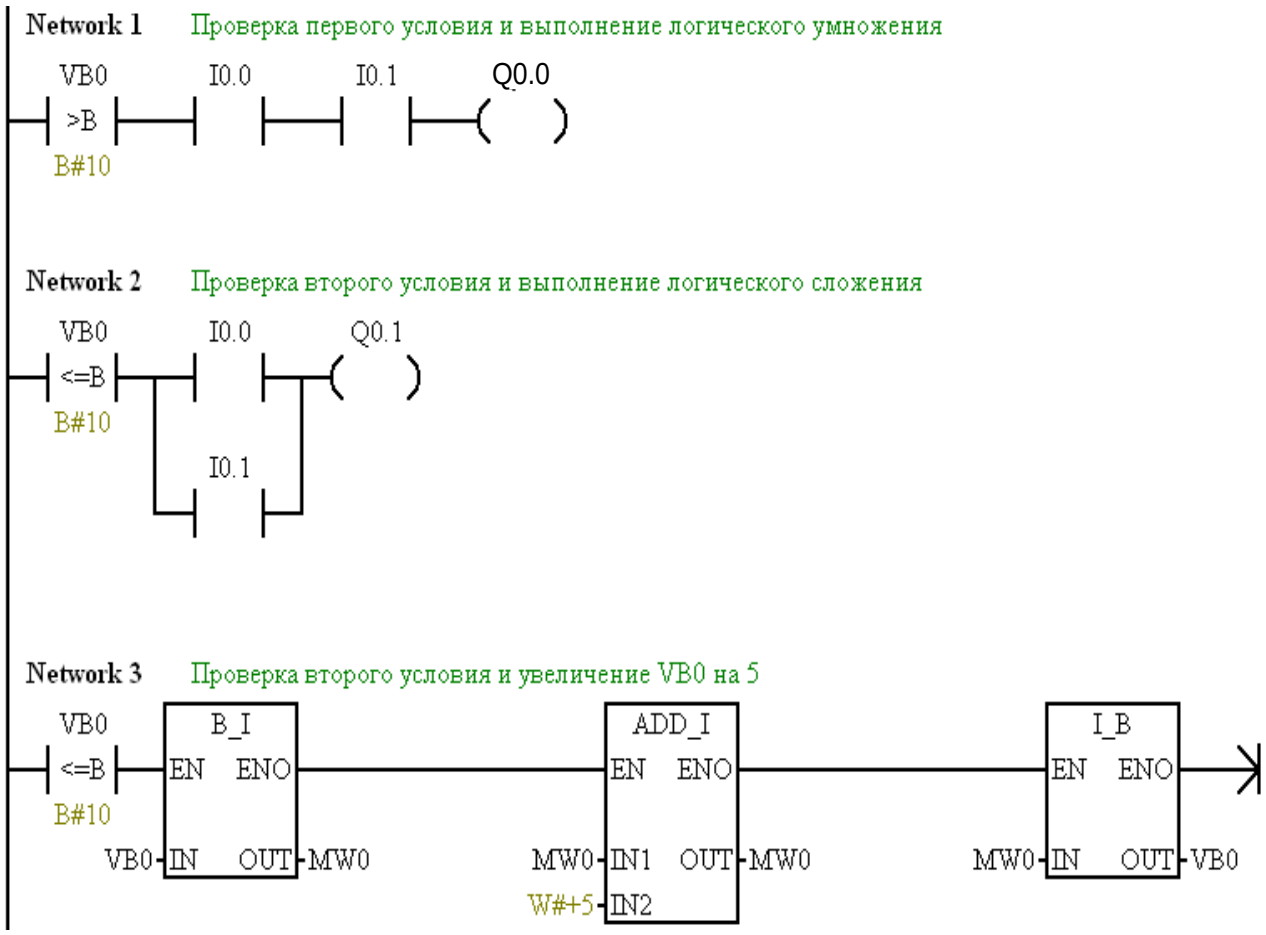


Рис. 2-10. Первая программа в редакторе контактных планов

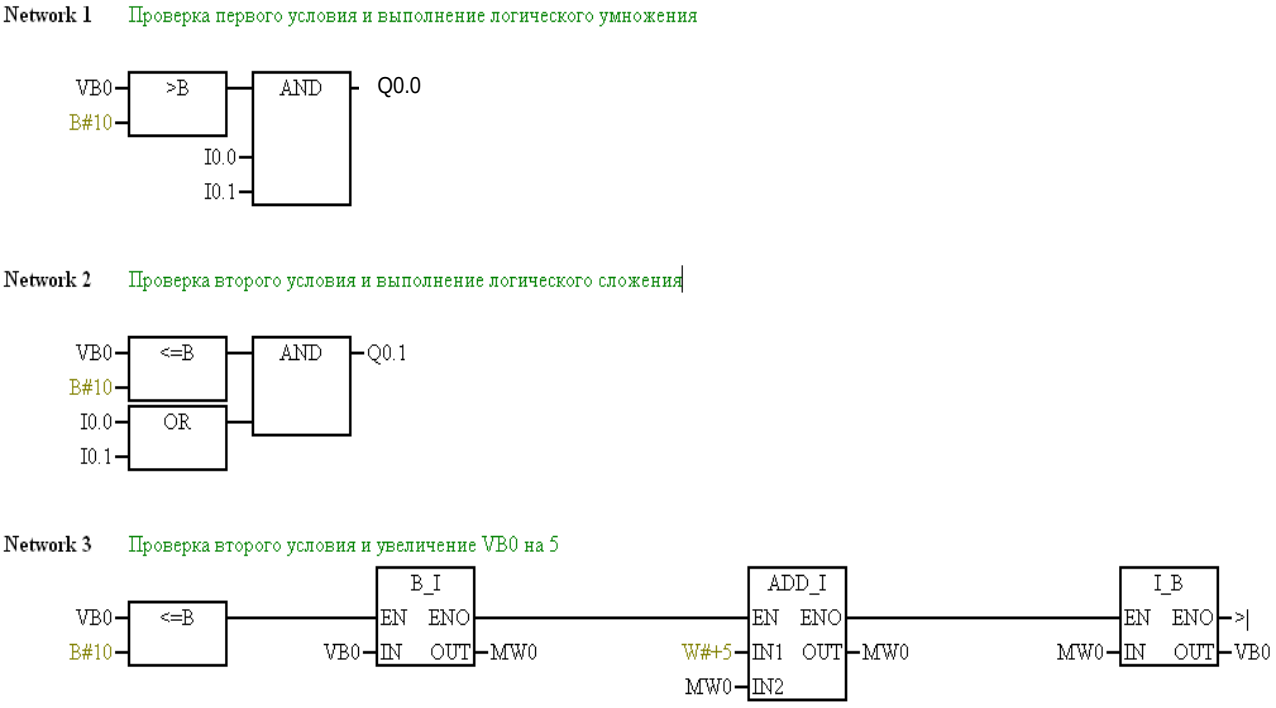


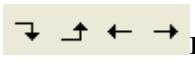
Рис. 2-11. Первая программа в редакторе функциональных блоков.

```
Network 1    Проверка первого условия и выполнение логического умножения
LDB>    VB0, B#10                  //Первичный опрос операции сравнения
A       I0.0                       //Логическое умножение стека и бита I0.0
A       I0.1                       //Логическое умножение стека и бита I0.1
=       Q0.0                       //Присвоение результата умножения биту Q0.0

Network 2    Проверка второго условия и выполнение логического сложения
LDB<=   VB0, B#10                  //Первичный опрос операции сравнения
LD      I0.0                       //Первичный опрос бита I0.0
O       I0.1                       //Логическое сложение стека и бита I0.1
ALD                                //Логическое умножение стека и результата сложения
=       Q0.1                       //Присвоение результата умножения биту Q0.0

Network 3    Проверка второго условия и увеличение VB0 на 5
LDB<=   VB0, B#10                  //Первичный опрос операции сравнения
BTI     VB0, MW0                   //Преобразование формата байта в формат слова
AENO                               //Проверка корректности выполнения блока преобразования форматов
+I      W#+5, MW0                  //Арифметическое сложение
AENO                               //Проверка корректности выполнения блока сложения
ITB     MW0, VB0                   //Преобразование формата слова в формат байта
```

Рис. 2-12. Первая программа в редакторе списка команд.

Ввод в программатор

Для набора текста программы в редакторе LAD необходимо выполнить последовательность действий, представленную на рис. 2-13.

При первоначальном вызове программной среды STEP 7-Micro/WIN по умолчанию выбирается Program Block в редакторе LAD. В поле дерева инструкций необходимо раскрывать интересующие папки. Так, для Network1 и Network2, таковыми являются папки Compare (Сравнение) и Bit Logic (Битовые операции). Для появления полного списка команд нужная папка открывается однократным щелчком ЛКМ по значку ⊞ рядом с обозначением закрытой папки. Однократным щелчком ЛКМ нужная команда выбирается в развернувшемся списке, при этом её изображение становится инверсным. Двойной щелчок ЛКМ по выбранному изображению вызывает копирование команды в рабочее поле, обведенное прямоугольником. Одновременно появляются поля операндов, отмеченные вопросительными знаками ?????. Для третьей сети Network3 помимо команды сравнения используются команды из папок Convert (Преобразование) и Integer Math (Целочисленная математика).Связи в строке программы для соединений, отличающихся от простого последовательного, осуществляются с помощью кнопок  панели инструкций Instruction.

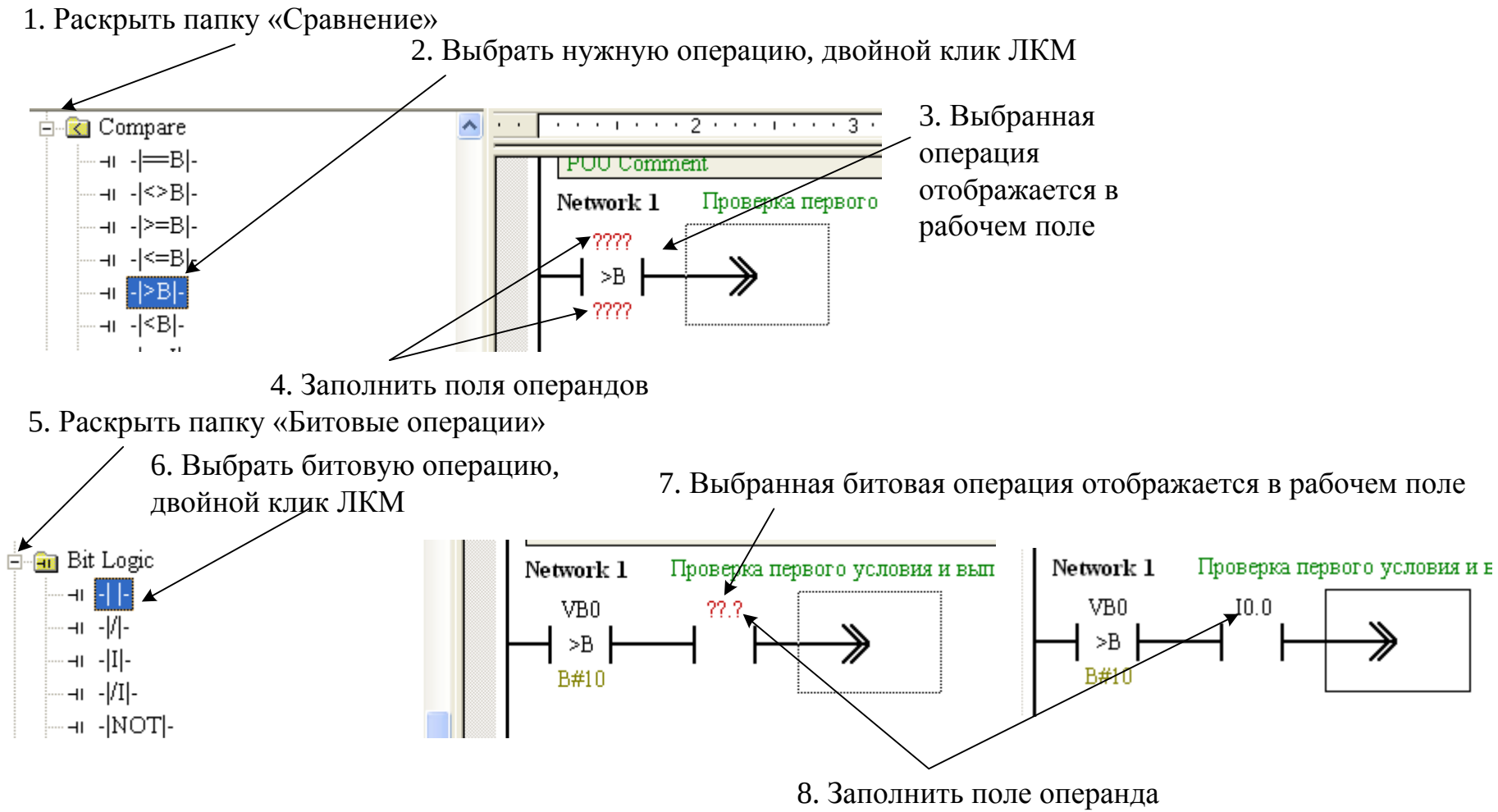


Рис. 2-13. Начало работы по набору первой программы в редакторе контактных планов

Комментарии к программе

В редакторе контактных планов можно ввести комментарии к программе в целом и к каждой сети в отдельности. Для отображения этих компонентов текста на экране необходимо, чтобы кнопки (Отображение комментария к программе) и (Отображение комментариев к сетям) на панели инструментов Common (Общая) были в нажатом состоянии.

Введение комментариев внутри каждой сети возможно только в редакторе списка команд (рис. 2-14). Переход из одного редактора в другой осуществляется выбором его через меню View (Вид).

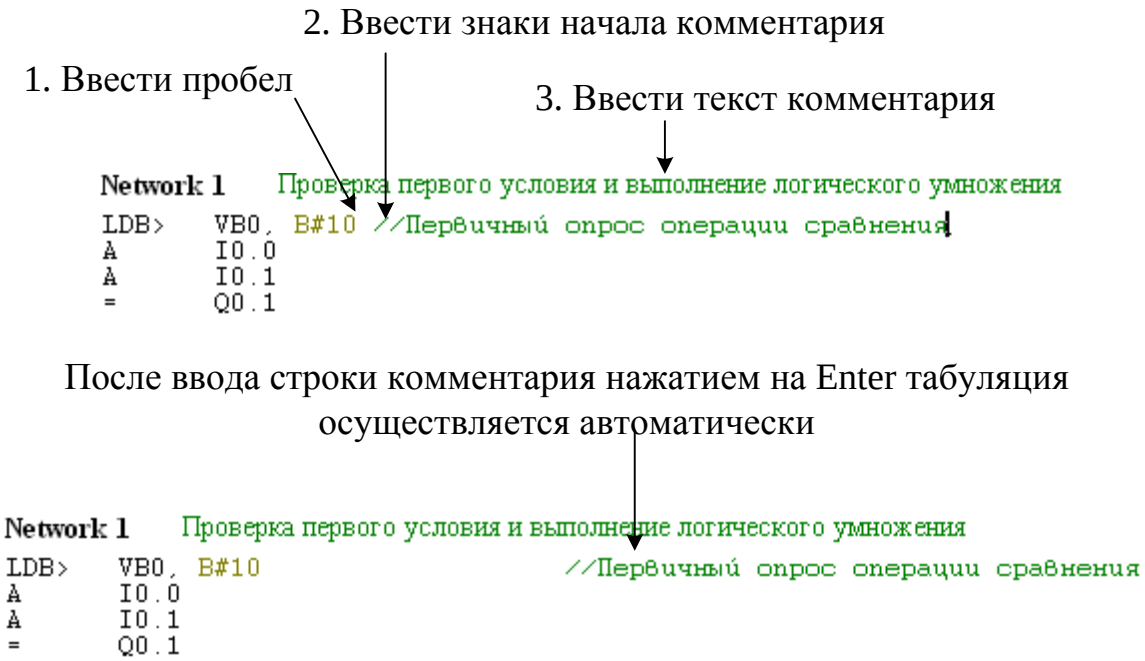
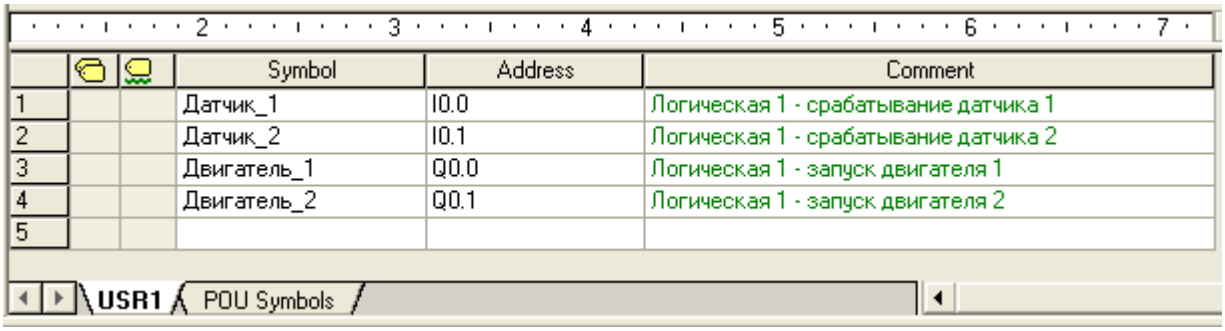


Рис. 2-14. Ввод комментариев в редакторе списка команд

Применение символьной адресации

При символьной адресации прямым адресам присваиваются символьные имена, выбор которых определяется функциональным назначением в программе. Например, адресу в области отображения информации на входах может соответствовать символьное имя «Датчик ….», адрес в области отображения информации на выходах – «Пуск двигателя» и т.п. При этом текст программы управления технологическим процессом становится более наглядным. Символьные имена задаются пользователем в таблице символьных имен, вызывае-

мой с навигационной панели Navigator Bar щелчком ЛКМ по иконке  Symbol Table (Таблица Символьных Имен). Таблица таких имён для рассматриваемого примера приведена на рис. 2-15.

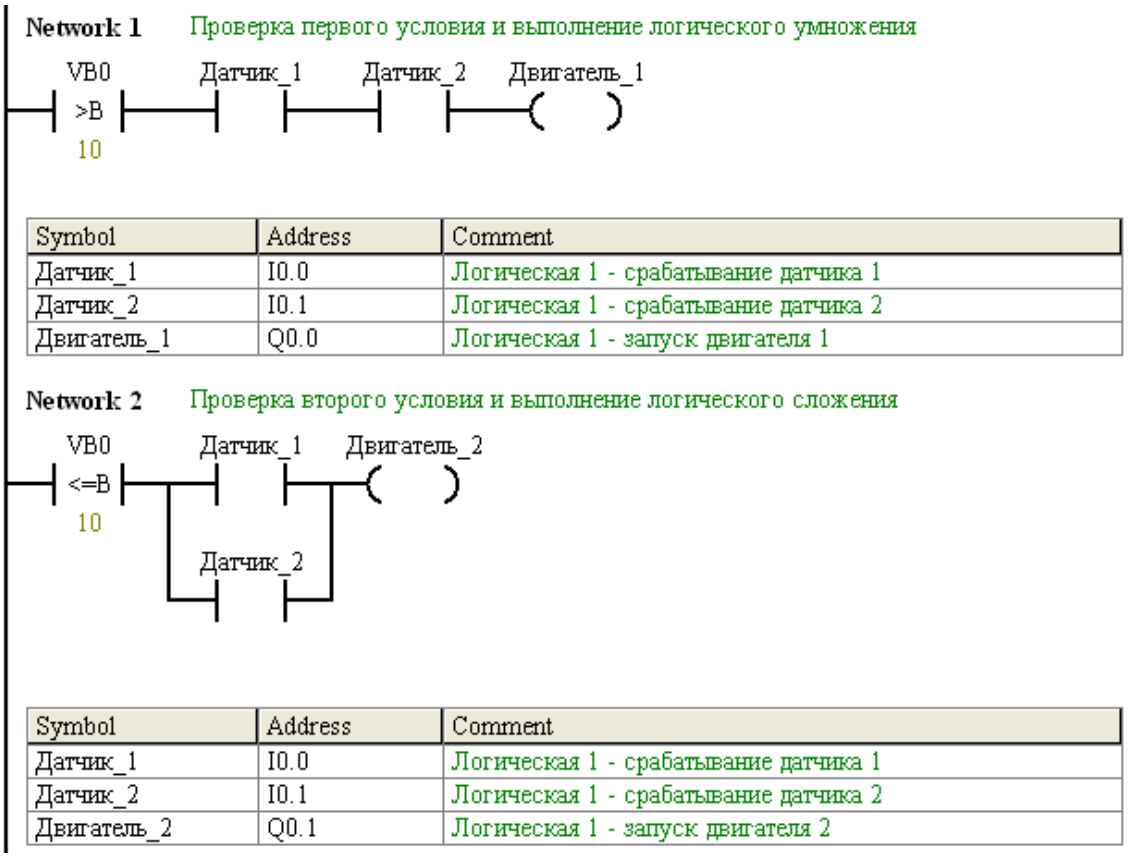


	Symbol	Address	Comment
1	Датчик_1	I0.0	Логическая 1 - срабатывание датчика 1
2	Датчик_2	I0.1	Логическая 1 - срабатывание датчика 2
3	Двигатель_1	Q0.0	Логическая 1 - запуск двигателя 1
4	Двигатель_2	Q0.1	Логическая 1 - запуск двигателя 2
5			

USR1 | POU Symbols

Рис. 2-15. Таблица символьных имен для Первой программы

Отображение символьных адресов в программе инициируется установкой соответствующих флажков в меню View (Вид). Изменение текста программы в сетях Network 1 и 2 рассматриваемого примера при применении символьной адресации приведено на рис. 2-16…2.18 (сравните с соответствующими рисунками 2-10 … 2-12).

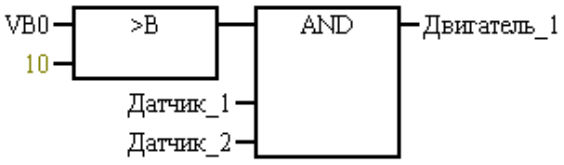


Symbol	Address	Comment
Датчик_1	I0.0	Логическая 1 - срабатывание датчика 1
Датчик_2	I0.1	Логическая 1 - срабатывание датчика 2
Двигатель_1	Q0.0	Логическая 1 - запуск двигателя 1

Symbol	Address	Comment
Датчик_1	I0.0	Логическая 1 - срабатывание датчика 1
Датчик_2	I0.1	Логическая 1 - срабатывание датчика 2
Двигатель_2	Q0.1	Логическая 1 - запуск двигателя 2

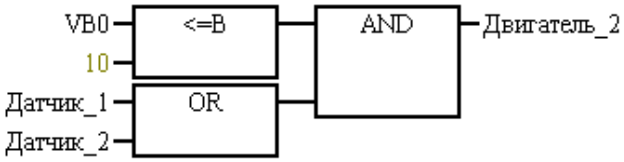
Рис. 2-16. Применение символьной адресации в редакторе LAD

Network 1 Проверка первого условия и выполнение логического умножения



Symbol	Address	Comment
Датчик_1	I0.0	Логическая 1 - срабатывание датчика 1
Датчик_2	I0.1	Логическая 1 - срабатывание датчика 2
Двигатель_1	Q0.0	Логическая 1 - запуск двигателя 1

Network 2 Проверка второго условия и выполнение логического сложения



Symbol	Address	Comment
Датчик_1	I0.0	Логическая 1 - срабатывание датчика 1
Датчик_2	I0.1	Логическая 1 - срабатывание датчика 2
Двигатель_2	Q0.1	Логическая 1 - запуск двигателя 2

Рис. 2-17. Применение символьной адресации в редакторе FBD

Network 1 Проверка первого условия и выполнение логического умножения

```
LDB>    VB0, 10                 //Первичный опрос операции сравнения
A       Датчик_1                //Первичный опрос бита I0.0
A       Датчик_2                //Логическое умножение стека и бита I0.1
=       Двигатель_1             //Присвоение результата умножения биту Q0.0
```

Symbol	Address	Comment
Датчик_1	I0.0	Логическая 1 - срабатывание датчика 1
Датчик_2	I0.1	Логическая 1 - срабатывание датчика 2
Двигатель_1	Q0.0	Логическая 1 - запуск двигателя 1

Network 2 Проверка второго условия и выполнение логического сложения

```
LDB<=   VB0, 10                 //Первичный опрос операции сравнения
LD      Датчик_1                //Первичный опрос бита I0.0
O       Датчик_2                //Логическое сложение стека и бита I0.1
ALD                             //Логическое умножение стека и результата сложения
=       Двигатель_2             //Присвоение результата умножения биту Q0.1
```

Symbol	Address	Comment
Датчик_1	I0.0	Логическая 1 - срабатывание датчика 1
Датчик_2	I0.1	Логическая 1 - срабатывание датчика 2
Двигатель_2	Q0.1	Логическая 1 - запуск двигателя 2

Рис. 2-18. Применение символьной адресации в редакторе списка команд

Отладка программы

Задание исходных данных

В соответствии с заданием необходим анализ содержимого адреса VB0. Оно может быть задано с помощью блока данных (приложение 2). Его вызов осуществляется нажатием на иконку Data Block  на навигационной панели проекта. Для примера возьмем VB0=25 (рис. 2-19).

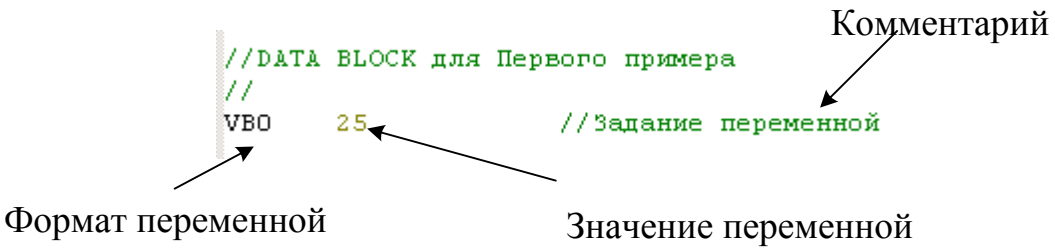


Рис. 2-19. Задание переменной в блоке данных

Компиляция программы

Режим компиляции вызывается одним из способов:

- выбором меню PLC (ПЛК) > Compile All (Компиляция всех компонентов);
- нажатием на кнопку Compile All  панели инструментов Standard.

Применение этой кнопки (а не Compile  на той же панели) обусловлено наличием нескольких компонентов программы пользователя, вызываемых кликом ЛКМ на соответствующие иконки на навигационной панели Navigator Bar:

- программный блок Program Block  ;
- таблица символьных имен Symbol Table  ;
- блок данных Data Block  .

При ошибочной компиляции хотя бы в одном из редакторов вместо ошибочной строки программы появляется красная надпись Invalid с номером сети. При отсутствии ошибок после успешной компиляции в информационном поле внизу экрана появляется соответствующее сообщение (рис. 2-20).

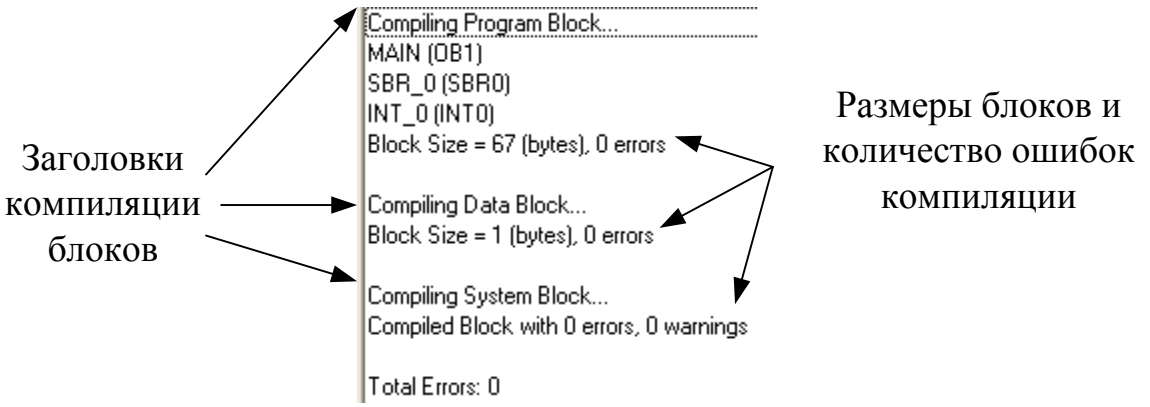


Рис. 2-20. Сообщение об успешной компиляции в поле состояния программы

Загрузка программы в CPU

Успешно скомпилированная программа может быть загружена в CPU:

- либо выбором меню File (Файл) > Download (передача «вниз» из PC, являющегося устройством высшего уровня, в CPU;
- нажатием кнопки Download на инструментальной панели Standard.

При успешной загрузке выводится соответствующее информационное сообщение. Если ранее в CPU была загружена программа, то после команды Download выводится диалоговое окно, требующее подтверждения замены программы пользователя в CPU. После подтверждения пользователем намерения заменить программу осуществляется её загрузка, а по её окончании загрузки на экран выводится соответствующее информационное сообщение.

Запуск программы

Загруженная в CPU программа запускается одним из способов:

- программатором, т.е. из программной среды STEP7-Micro/WIN (переключатель режимов под крышкой на передней панели CPU должен находиться в среднем положении TERM – управление от терминала):
  - выбором меню PLC (ПЛК) > RUN (Пуск);
  - нажатием кнопки RUN на инструментальной панели Debug (Отладка);
- перестановкой переключателя режимов под крышкой на передней панели CPU в положение RUN.

Визуализация результатов

Контроль за выполнением программы осуществляется одним из способов:

- Через таблицу состояний Status Chart (приложение 3). В примере нужно контролировать состояние двух входов – I0.0 и I0.1, двух выходов – Q0.0 и Q0.1, а также содержимое области памяти переменных VB0. Перечень этих областей памяти вносится пользователем в адресное поле Address таблицы (её вид представлен в приложении 3 на рис. 3-2).
- Введением режима Program Status (приложение 4).

Вопросы для самопроверки

1. Перечислите форматы, с которыми работают блоки арифметических операций, операции сравнения.
2. Назовите допустимый диапазон параметров для формата B, I, R.
3. Назовите особенности графических редакторов FBD и LAD.
4. Назовите особенности текстового редактора STL.
5. Какие графические элементы применяются в редакторах FBD и LAD?
6. Каким образом в редакторе FBD осуществляется инвертирование сигналов?
7. Какова последовательность работы с проектом?
8. Каковы составные части проекта пользователя и их функциональное назначение?
9. Назовите способы задания исходных данных.
10. Перечислите способы контроля выполнения программы CPU.
11. Каковы особенности режима Program Status в различных редакторах?
12. Перечислите режимы работы CPU.
13. Какие способы запуска программы Вы знаете?