
Содержание главы 6

6	<u>Функции передачи</u>	4
6.1	<u>Общие замечания</u>	4
6.2	<u>Блочный элемент MOVE</u>	6
6.2.1	<u>Обработка блочного элемента MOVE</u>	6
6.2.2	<u>Перемещение операндов</u>	9
6.2.3	<u>Передача констант</u>	11
6.3	<u>Системные функции для передачи данных</u>	12
6.3.1	<u>Указатель типа ANY</u>	12
6.3.2	<u>Копирование области данных</u>	13
6.3.3	<u>Непрерывное копирование области данных</u>	14
6.3.4	<u>Заполнение области данных</u>	14

6 Функции передачи

Языки программирования LAD и FBD предоставляют следующие move-функции (функции пересылки, перемещения данных):

- Блочный элемент MOVE
Копирует операнды и переменные простых типов данных;
- SFC 20 BLKMOV
Копирует область данных;
- SFC 21 FILL
Заполняет область данных;
- SFC 81 UBLKMOV
Непрерывное копирование области данных.

SFC – это системные функции из стандартной библиотеки *Standard Library* программы *System Function Blocks* (Системные функциональные блоки).

6.1 Общие замечания

Функции пересылки данных используются для копирования информации между системной памятью (system memory), пользовательской памятью (user memory) и областью пользовательских данных модулей (рисунок 6.1). Информация передается через внутренний регистр CPU, который функционирует как промежуточное запоминающее устройство (промежуточная память).

Регистр называется аккумулятором 1 (accumulator 1). Перемещение информации из памяти в аккумулятор 1 называется «загрузка» («loading»), а перемещение из аккумулятора 1 в память называется передачей или пересылкой («transferring»). Блочный элемент MOVE содержит оба пути передачи. Он перемещает информацию со входа IN в аккумулятор 1 (загружает) и сразу после этого – из аккумулятора 1 в операнд на выходе (передает).

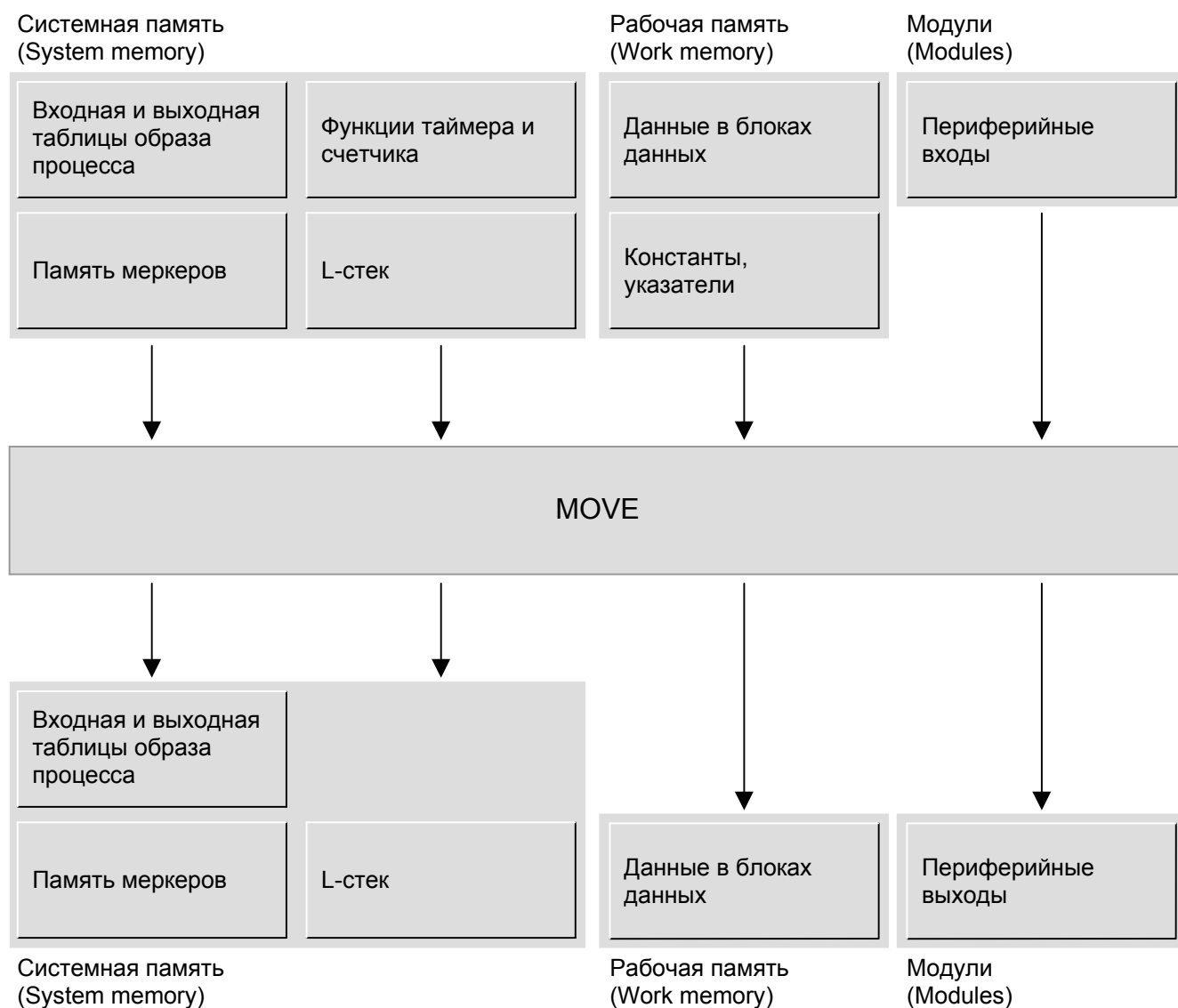


Рисунок 6.1 Области памяти для загрузки (Loading) и передачи (Transferring)

6.2 Блочный элемент MOVE

6.2.1 Обработка блочного элемента MOVE

Представление

В дополнение к разрешающему входу (enable input) EN и разрешающему выходу ENO блочный элемент MOVE имеет вход IN и выход OUT. На входе IN и выходе OUT вы можете задействовать все цифровые операнды и цифровые переменные простых типов данных (за исключением BOOL – логического типа). Переменные у входа IN и выхода OUT могут иметь разные типы данных.

Блочный элемент MOVE



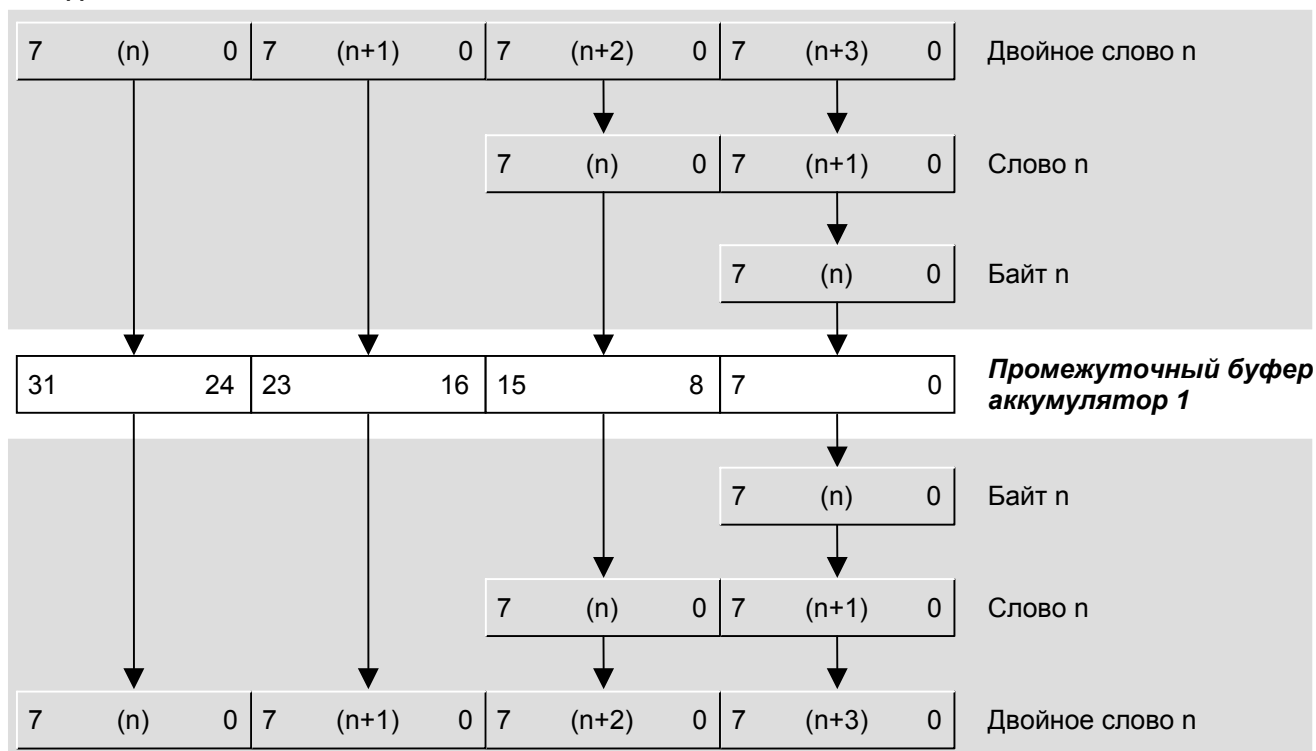
Содержимое и назначение битов в форматах данных подробно обсуждены в параграфе 3.5 «Переменные, константы и типы данных».

Блочный элемент MOVE, который можно использовать в пошаговом программировании, вы найдете в каталоге программных элементов (Program Element Catalog), выбрав команду меню View → Catalog (Вид → Каталог) [Ctrl + K] или Insert → Program Elements (Вставка → Программные элементы) в разделе «Shift» («Сдвиг»).

Различные размеры операндов

Размеры операндов (байт, слово, двойное слово) на входе и выходе блочного элемента MOVE могут быть разными. Если операнд на входе «меньше», чем на выходе, он передается в выходной операнд с правым выравниванием (значение передаваемого операнда заполняет правые биты результата) и заполнением нулями левой части. Если входной операнд «больше» выходного операнда, то передается только правая часть операнда, соответствующая выходному операнду.

Рисунок 6.2 объясняет этот факт. Байт или слово на входе загружается с правым выравниванием в аккумулятор 1, а оставшаяся часть заполняется нулями. Байт или слово на выходе OUT извлекается из аккумулятора 1 с правым выравниванием.

Вход IN**Рисунок 6.2** Передача операндов различных размеров**Функция**

Блочный элемент MOVE перемещает информацию операнда на входе IN в операнд на выходе OUT. Блочный элемент MOVE только тогда совершает передачу информации, когда разрешающий вход (enable input) равен «1» или не используется, и когда главное реле управления (master control relay, MCR) не активировано.

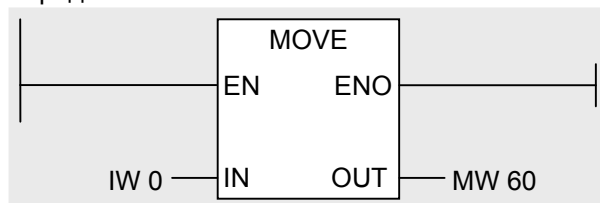
Если EN = «1» и MCR активировано, то на выход OUT записывается нуль. При наличии «0» на разрешающем входе операнд на выходе OUT остается неизменным. MOVE об ошибке не сообщает.

ЕСЛИ EN == “1” или не используется			ИНАЧЕ
ТО			
ENO := “1”			
ЕСЛИ MCR активно			
ТО	ИНАЧЕ		
OUT := 0	OUT := 1		
			ENO := “0”

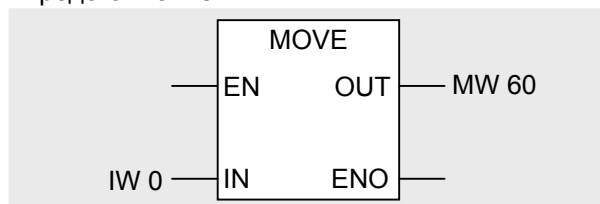
Пример

Содержимое входного слова (input word) IW 0 передается в слово памяти (memory word) MW 60.

Представление LAD:



Представление FBD:



Блочный элемент MOVE в цепи (LAD)

Перед входом EN и после выхода ENO вы можете выстроить контакты последовательно или параллельно.

Блочный элемент MOVE должен быть помещен только на ветвь, соединенную непосредственно с левой направляющей (питающей шиной). Эта ветвь также может иметь контакты перед входом EN, и это не должна быть верхняя ветвь. С помощью прямого соединения с левой направляющей (питающей шиной) вы можете подключить блочные элементы MOVE параллельно. При параллельном соединении блочных элементов вам необходима катушка, чтобы завершить цепь. Если вы не предусмотрели оценку ошибки, назначьте катушке «фиктивный» операнд, например, бит временных локальных данных.

Вы можете соединить блочные элементы MOVE последовательно. При этом выход ENO предшествующего элемента соединяется с входом EN следующего.

Если вы сконпоновали несколько блочных элементов MOVE в одной цепи (параллельно у левой направляющей, или питающей шины, и далее продолжив последовательно), то блочные элементы самой верхней ветви обрабатываются первыми слева направо, затем элементы параллельной ветви слева направо и так далее.

Примеры функций передачи вы можете найти на дискете, прилагаемой к книге (библиотека «LAD_Book» в функциональном блоке FB 106 программы «Basic Functions», «Основные функции»).

Блочный элемент MOVE в логической схеме (FBD)

Если вы хотите обработать блочный элемент MOVE в зависимости от определенных условий, вы можете запрограммировать бинарные логические схемы перед входом EN. Вы можете соединить выход ENO с двоичными входами других функций; к примеру, можно скомпоновать блочные элементы MOVE последовательно, где выход ENO предыдущего элемента соединен с входом EN последующего блочного элемента.

EN и ENO не должны быть жестко смонтированы.

Примеры функций передачи вы можете найти на дискете, прилагаемой к книге, в библиотеке «FBD_Book», FB 106 в программе «Basic Functions» («Основные функции»).

6.2.2 Перемещение операндов

Кроме упомянутых в данной главе операндов, вы можете также пересылать значения таймеров и счетчиков (см. главу 7 «Таймеры» и главу 8 «Счетчики»). Параграф 18.2 «Функции блока для блоков данных» рассказывает об использовании операндов данных.

Передача входов

IB n	Передача входного байта
IW n	Передача входного слова
ID n	Передача входного двойного слова

В некоторых CPU загрузка из и передача во входную таблицу образа процесса допустимы только для входных байтов, которые также доступны как входной модуль (см. параграф 1.5.2 «Образ процесса»).

Передача выходов

QB n	Передача выходного байта
QW n	Передача выходного слова
QD n	Передача выходного двойного слова

В некоторых CPU загрузка из выходной таблицы образа процесса и передача в нее допустимы только для выходных байтов, которые также доступны как выходной модуль (см. параграф 1.5.2 «Образ процесса»).

Передача из I/O (периферийных входов/выходов)

PIB n	Загрузка байта периферийного входа
PIW n	Загрузка слова периферийного входа
PID n	Загрузка двойного слова периферийного входа
PQB n	Передача байта в периферийный выход
PQW n	Передача слова в периферийный выход
PQD n	Передача двойного слова в периферийный выход

При передаче данных в области периферийных входов/выходов (в I/O-области) вы можете получить доступ к различным операндам в зависимости от направления передачи. Вы должны определить I/O-входы (входы PI) при входе IN блочного элемента MOVE и I/O-выходы (выходы PQ) при выходе OUT.

Когда происходит передача из I/O (периферийного входа/выхода) (операция загрузки), доступ к входным модулям осуществляется как к периферийным входам (peripheral inputs, PI). Адресоваться могут только доступные модули. Обратите внимание, что прямая загрузка из I/O-модулей (модулей периферийных входов/выходов) может пересылать значение, отличное от загружаемого из входов модуля с теми же адресами. Пока сигнальное состояние входов соответствует значениям в начале программного цикла (когда CPU обновил образ процесса), значения, загруженные непосредственно из I/O-модулей, являются текущими значениями.

Периферийные выходы (peripheral outputs, PQ) используются для передач в I/O. Доступ возможен только к тем адресам, по которым расположены I/O-модули. Передача в I/O-модули, имеющие таблицу образа процесса, одновременно обновляет эту выходную таблицу образа процесса, поэтому между одинаково адресуемыми выходами и периферийными выходами различий нет.

Передача меркеров

MB n	Передача байта меркерной памяти
MW n	Передача слова меркерной памяти
MD n	Передача двойного слова меркерной памяти

Передача из и в область адресов памяти меркеров всегда допустима, так как вся память меркеров расположена в CPU. Заметьте, что у CPU разных версий размеры области памяти меркеров различны.

Передача временных локальных данных

LB n	Передача байта локальных данных
LW n	Передача слова локальных данных
LD n	Передача двойного слова локальных данных

Перемещение из и в L-стек всегда разрешено. За информацией по этому вопросу обратитесь к параграфу 18.1.5 «Временные локальные данные».

6.2.3 Передача констант

Определить константы можно только на входе IN блочного элемента MOVE.

Передача констант простых типов данных

Фиксированное значение, или константа, может быть переслано в операнд. Иначе говоря, эта константа может быть передана в нескольких различных форматах. В параграфе 3.5.4 «Простые типы данных» вы найдете обзор различных имеющихся форматов. Все константы, которые могут быть переданы с помощью блочного элемента MOVE, принадлежат простым типам данных. Примеры:

B#16#F1	Передача двузначного шестнадцатеричного числа
– 1000	Передача целого числа (типа INT)
5.0	Передача вещественного числа (типа REAL)
S5T#2s	Передача таймера S5
TOD#8:30	Передача времени суток

Передача указателей

Указатели (pointers) – это особый вид констант, используемый для вычисления адресов в стандартных блоках. Вы можете использовать блочный элемент MOVE для сохранения этих указателей в операндах.

P#1.0	Передача внутреннего указателя области (area-internal pointer)
P#M2.1	Передача указателя пересечения областей (area-crossing pointer)

6.3 Системные функции для передачи данных

Для передачи данных доступны следующие системные функции:

- SFC 20 BLKMOV
Копирует область данных;
- SFC 21 FILL
Заполняет область данных;
- SFC 81 UBLKMOV
Непрерывное копирование области данных.

Эти системные функции имеют два параметра типа ANY (таблица 6.1). В принципе, вы можете назначить этим параметрам любой операнд, любую переменную или любую абсолютно адресуемую область. Если вы используете переменную сложного типа данных, то это может быть только «вся» переменная (целиком); компоненты переменной (отдельный массив или структурные компоненты, например) недопустимы. Для определения абсолютно адресуемой области используйте указатель типа ANY (Любой).

Таблица 6.1 Параметры для SFC 20, 21 и 81

SFC	Параметр	Объявление	Тип данных	Содержимое, характеристика
20	SRCBLK	INPUT	ANY	Исходная область, из которой копируются данные
	RET_VAL	OUTPUT	INT	Информация об ошибке
	DSTBLK	OUTPUT	ANY	Область назначения, в которую копируются данные
21	BVAL	INPUT	ANY	Копируемая исходная область
	RET_VAL	OUTPUT	INT	Информация об ошибке
	BLK	OUTPUT	ANY	Область назначения, в которую копируется исходная область (включая несколько копий)
81	SRCBLK	INPUT	ANY	Исходная область, из которой копируются данные
	RET_VAL	OUTPUT	INT	Информация об ошибке
	DSTBLK	OUTPUT	ANY	Область назначения, в которую копируются данные

6.3.1 Указатель типа ANY

Когда требуется определить область операнда, адресуемую абсолютно, в качестве параметра блока типа ANY, используется указатель данного типа. Общий формат ANY-указателя следующий:

R#[DataBlock]Operand Type Quantity (R#[БлокДанных]Операнд Тип Количество)

Примеры:

P#M16.0 BYTE 8	Область из 8 байт, начинающаяся с MB 16
P#DB11.DBX30.0 INT 12	Область из 12 слов в DB 11, начинающаяся с DBB 30
P#I18.0 WORD 1	Входное слово IW 18
P#I1.0 BOOL 1	Вход I 1.0

Обратите внимание, что адрес операнда в указателе типа ANY всегда должен быть адресом бита.

Определить постоянный ANY-указатель имеет смысл, когда вы хотите получить доступ к области данных, для которой переменных не объявлено. В принципе, ANY-параметру вы можете назначить переменные или операнды. Например, 'P#I1.0 BOOL 1' идентично 'I 1.0' или соответствующему символическому адресу.

6.3.2 Копирование области данных

Системная функция **SFC 20 BLKMOV** копирует содержимое исходной области (параметр SLCBLK) в область назначения (параметр DSTBLK) в направлении роста адресов (пошагового увеличения).

Могут быть назначены следующие фактические параметры:

- Любые переменные из областей операндов для входов (I), выходов (Q), меркеров (M) и блоков данных (переменные из глобальных блоков данных и из экземплярных блоков данных);
- Переменные из временных локальных данных (использование типа данных ANY определяется особыми обстоятельствами);
- Абсолютно адресуемые области данных, которые требуют определения ANY-указателя.

SFC 20 нельзя использовать для копирования таймеров или счетчиков, для копирования информации из или в модули (область операнда P) или копирования системных блоков данных (блоков SDB).

В случае входов и выходов определенная область копируется независимо от того, указывают фактически адреса на модули входов или выходов или нет. Вы также можете определить переменную или область из блока данных в загрузочной памяти (блок данных, запрограммированный с ключевым словом UNLINKED, Несвязанный) в качестве исходной области.

Исходная область и область назначения не могут перекрываться. Если исходная область и область назначения имеют разные размеры, передача происходит, исходя из длины наименьшей из двух областей.

Пример (рисунки 6.3а, 6.3б, сеть 4, Network 4): начиная с байта памяти (меркеров) MB 64, 16 байт копируются в блок данных DB 124, начиная от DBB 0.

6.3.3 Непрерывное копирование области данных

Системная функция **SFC 81 UBLKMOV** копирует содержимое исходной области (параметр SRCBLK) в область назначения (параметр DSTBLK) в направлении возрастания адресов (пошагового увеличения). Функция копирования непрерывна, создает возможность увеличенных временных интервалов откликов на прерывания. Скопировано может быть максимально 512 байт.

Могут быть назначены следующие фактические параметры:

- Любые переменные из областей операндов для входов (I), выходов (Q), меркеров (M) и блоков данных (переменные из глобальных блоков данных и из экземплярных блоков данных);
- Переменные из временных локальных данных (использование типа данных ANY определяется особыми обстоятельствами);
- Абсолютно адресуемые области данных, которые требуют определения ANY-указателя.

SFC 81 не может использоваться для копирования таймеров или счетчиков, для копирования информации из или в модули (область операнда P) или для копирования системных блоков данных (блоков SDB) или блоков данных в загрузочной памяти (блок данных, запрограммированный с ключевым словом UNLINKED, Несвязанный).

В случае входов и выходов определенная область копируется независимо от того, указывают их адреса на модули входов или выходов или нет.

Исходная область и область назначения не могут перекрываться. Если исходная область и область назначения имеют разные размеры, передача происходит, исходя из длины наименьшей из двух областей.

6.3.4 Заполнение области данных

Системная функция **SFC 21 FILL** копирует установленное значение (исходную область) в область памяти (область назначения) с целью полной перезаписи области назначения. Передача производится в направлении возрастания адресов (пошагового увеличения). Могут быть назначены следующие фактические параметры:

- Любые переменные из областей операндов для входов (I), выходов (Q), меркеров (M) и блоков данных (переменные из глобальных блоков данных и из экземплярных блоков данных);

- Абсолютно адресуемые области данных, которые требуют определения ANY-указателя;
- Переменные из временных локальных данных (использование типа данных ANY определяется особыми обстоятельствами).

SFC 21 не может использоваться для копирования таймеров или счетчиков, для копирования информации из или в модули (область операнда P) или системных блоков данных (блоков SDB).

В случае входов и выходов определенная область копируется независимо от того, указывают фактически адреса на модули входов или выходов или нет.

Исходная область и область назначения не могут перекрываться. Область назначения всегда полностью перезаписывается, даже когда исходная область больше области назначения, или когда длина области назначения не кратна целому числу размеров исходных областей.

Пример (рисунки 6.3а, 6.3б, сеть 5, Network 5): содержимое байта памяти (меркеров) MB 80 копируется 16 раз в блок данных DB 124, начиная с DBB 16.

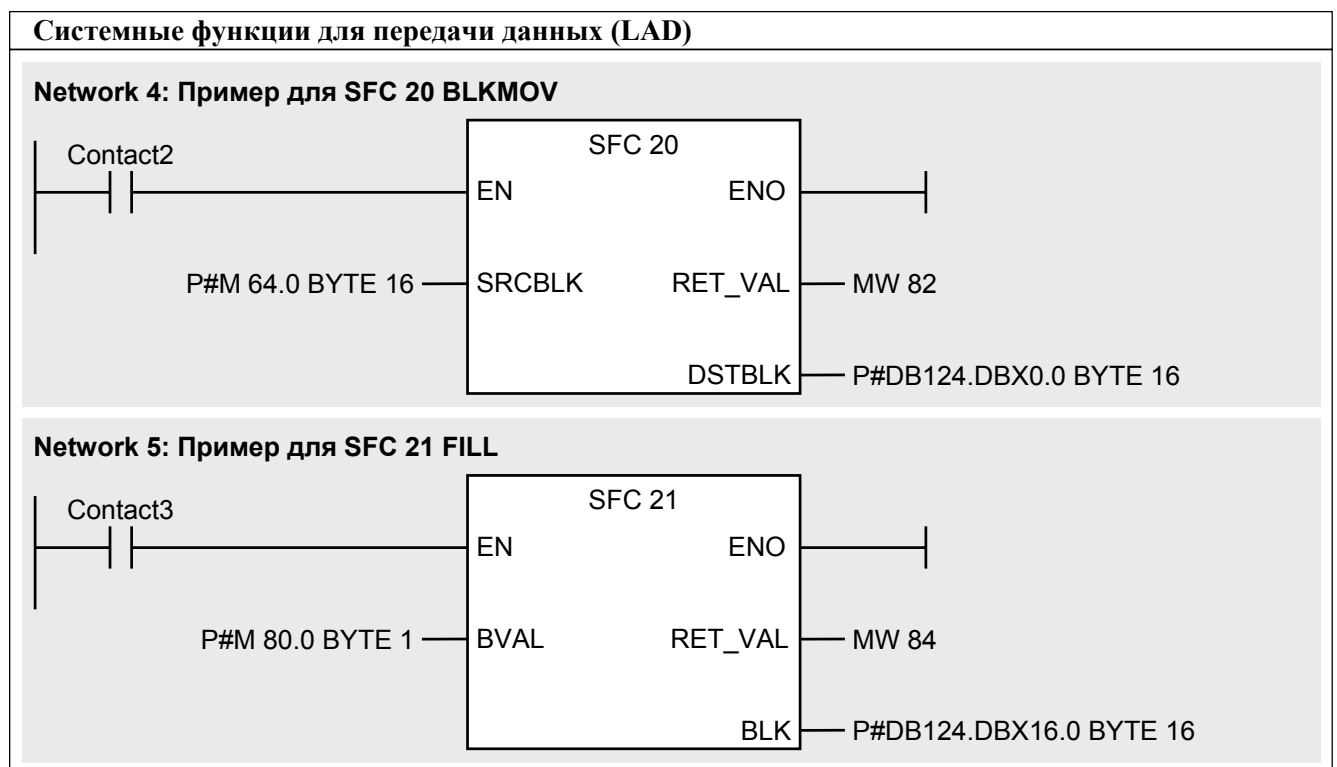


Рисунок 6.3а Примеры для SFC 20 BLKMOV и SFC 21 FILL

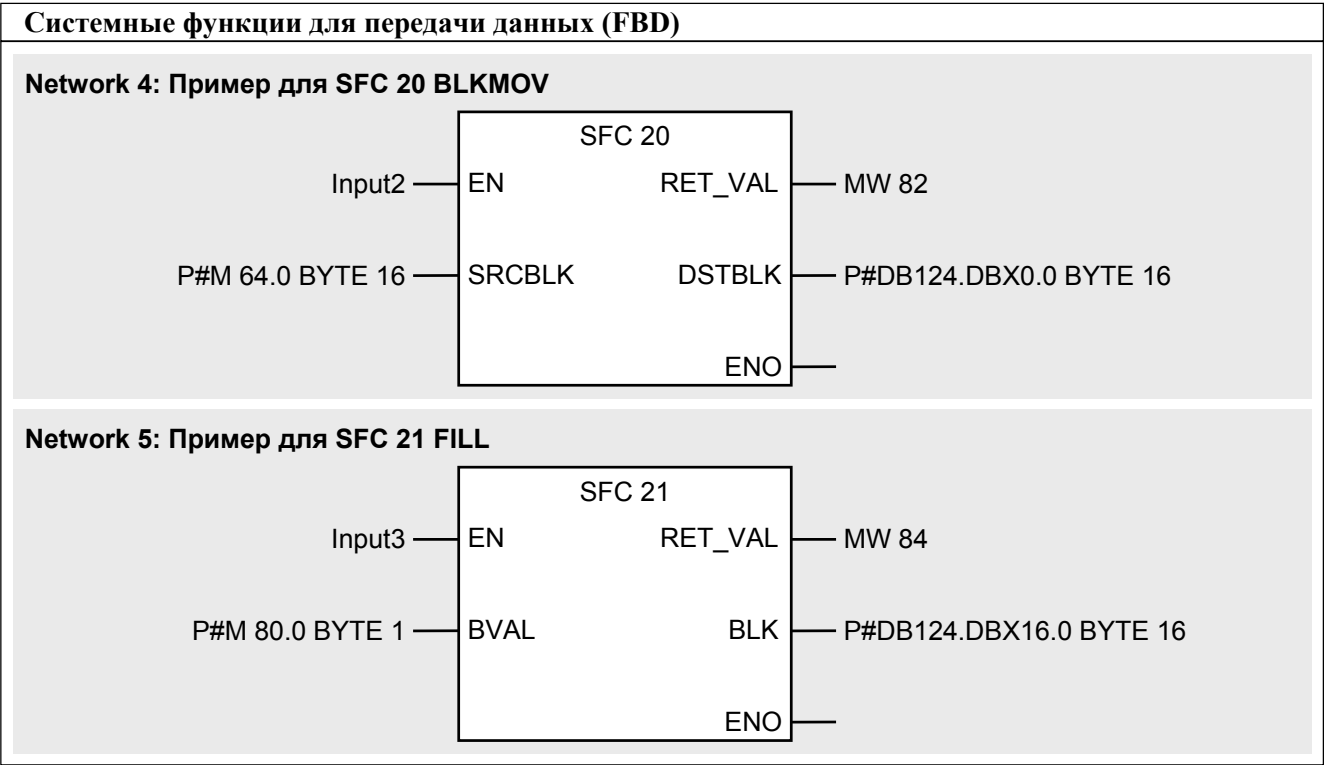


Рисунок 6.36 Примеры для SFC 20 BLKMOV и SFC 21 FILL